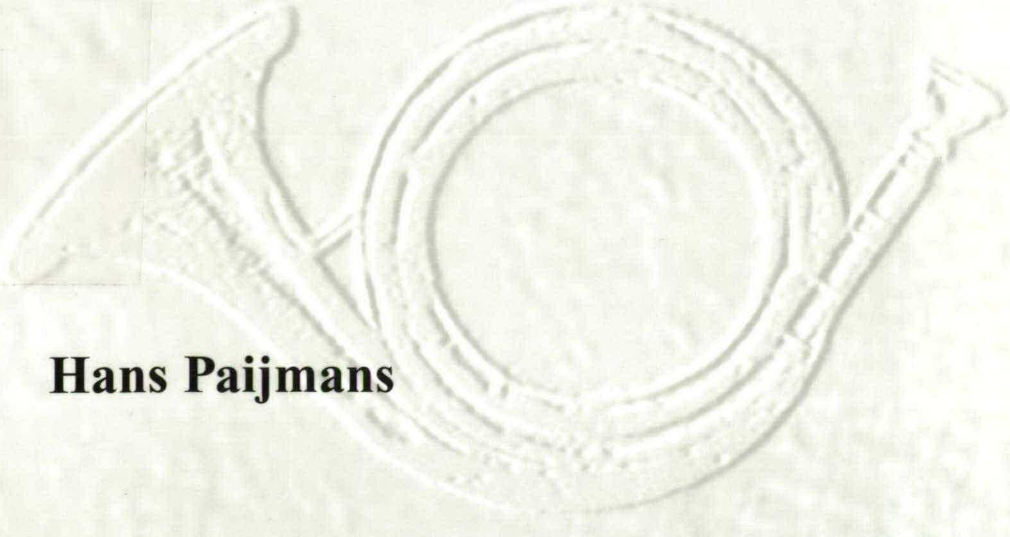
Explorations

in the

Document Vector Model

of

Information Retrieval



Hans Paijmans

Explorations in the Document Vector Model of Information Retrieval

Hans Paijmans

Explorations in the Document Vector Model of Information Retrieval

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Katholieke Universiteit Brabant, op gezag van de rector magnificus,
prof. dr. F.A. van der Duyn Schouten, in het openbaar te verdedigen ten
overstaan van een door het college voor promoties aangewezen commissie in
de aula van de Universiteit op dinsdag 14 september 1999 om 11:15 uur

door

Josephus Johannes Paijmans

geboren op 5 april 1948 te Oisterwijk.

Promotor: prof. dr. H. Bunt
Copromotor: prof. dr. W. Daelemans

ISBN 90-361-00-240
NUGI 601

Voor mijn ouders

His progress through life was hampered by his tremendous sense of his own ignorance, a disability which affects all too few people.
Terry Pratchett, Maskerade...

Contents

1 Introduction **3**
1.1 The document vector model as basis for a taxonomy 3
1.2 Expository text and other text types 6
1.3 Inside the document 8
1.3.1 Text models for information retrieval 8
1.3.2 Structures in texts 11
1.3.3 Topicality 12
1.4 IR and database management systems 14
1.4.1 Information retrieval as a classification task 16
1.4.2 Classification and categorization 17
1.4.3 The selection of features 18
1.4.4 The Blair/Maron experiment 20
1.4.5 The IR paradoxes 20

2 IR in historical perspective **23**
2.1 Short history of IR 23
2.1.1 The Assyrian and Hellenistic libraries 23
2.1.2 The Romans and early christians 26
2.1.3 Medieval libraries 28
2.1.4 Dewey's revolution 29
2.1.5 Other library systems 30
2.2 The index language model 33

3 Automated information retrieval **37**
3.1 The onslaught of mechanization 37
3.1.1 The computer and the library 38
3.2 Indexing by computer 41
3.3 Measuring performance 42
3.3.1 Precision and recall 43
3.3.2 Specificity and exhaustivity 48
3.4 Computing a baseline 48
3.5 Intermezzo: artificial intelligence 51
3.5.1 Frames and scripts 52
3.5.2 Bottom-up and top-down modelling 52
3.5.3 Rule-based information retrieval 58
3.5.4 Neural networks 60

3.6 Conclusions 61

4 Vector-based models 63

4.1 The general model of IR 63
4.2 The document–IL translation 65
4.3 The Boolean model 66
4.3.1 Frequency–based weights 68
4.3.2 Differences between the two methods 75
4.3.3 Feature reduction models 76
4.4 Vectors and similarity functions 76
4.4.1 Topical similarity 77
4.4.2 The vector space model 79
4.4.3 Similarity functions 80
4.5 The cluster model 82
4.6 The probabilistic models 83
4.7 The relevance feedback model 85
4.7.1 Genetic algorithms 86
4.8 Conclusions 87

5 Comparing IR-systems 91

5.1 Introduction 91
5.2 Background 92
5.2.1 Derived and assigned indexing 93
5.2.2 Pre- and postcoordination 94
5.3 The systems 95
5.3.1 TOPIC or RUBRIC 95
5.3.2 CLARIT 96
5.4 First discussion 98
5.5 Methodology 100
5.5.1 Selection and preparation of the databases 101
5.5.2 Collecting the data 103
5.5.3 Virtual docreps 104
5.6 Conclusions and suggestions 105
5.7 APPENDIX 108
5.7.1 Document140 108

6 Gravity wells of meaning 111

6.1 Introduction 111
6.2 Organization of experiments 113
6.3 Weighting schemes 115
6.3.1 Frequency-based term-weighting schemes 116
6.3.2 The term discrimination value 116
6.3.3 Differences between the two frequency-based methods . . 117
6.3.4 Logical document structure 119
6.3.5 The first-mention theory 119
6.3.6 Cue-words or 'weak semantics' 119

6.3.7 Word categories 120
6.4 Methodology 120
6.4.1 The Pearson-test 121
6.4.2 The T-test on word stems 121
6.4.3 T-tests on sentences 122
6.4.4 Word categories and word weights 123
6.5 Conclusions 124
6.6 Acknowledgments 126
6.7 Appendix: list of articles. 126

7 Local Dictionaries 129
7.1 Introduction 129
7.1.1 The index language 130
7.2 Information retrieval as a classification task 132
7.2.1 Classification and categorization 132
7.2.2 Relevance feedback 132
7.2.3 Surface properties of text as features 133
7.2.4 Selection and reduction 133
7.2.5 Class representations 134
7.2.6 Similarity functions 135
7.3 Methodology and experiments 136
7.3.1 Evaluation issues 136
7.3.2 A baseline 139
7.3.3 The database 142
7.3.4 The experiments 143
7.4 Conclusions 147
7.4.1 The local dictionaries 147
7.4.2 The strategies 148
7.4.3 Further work 148

8 Author Recognition 151
8.1 Introduction 151
8.2 Authorship attribution 152
8.2.1 Some notes on terminology 152
8.2.2 Genre classification 152
8.2.3 Content analysis 153
8.2.4 Comparing authors 153
8.3 Stylistic fingerprints 155
8.4 Lexical cohesion 157
8.4.1 Text Tiling 158
8.4.2 Measures for Lexical Cohesion 159
8.5 Machine learning 159
8.5.1 IBL4 161
8.5.2 Kolmogorov* 162
8.6 Methodology and experiments 162
8.6.1 Preparation of the texts 163

8.6.2 Creation of the databases 163
8.6.3 The search space 166
8.7 Conclusions 170
8.8 Appendix 171
8.8.1 Corpus I 171
8.8.2 Corpus II 171
8.8.3 Corpus III 171

9 Conclusions **173**
9.1 Synopsis of results 175
9.1.1 Clarit-Topic 175
9.1.2 Gravity wells 177
9.1.3 Local dictionaries 178
9.1.4 Author recognition 179
9.2 Main findings 181
9.3 Future work 182

10 Samenvatting **185**
10.1 Clarit-Topic 186
10.2 Gravity wells 186
10.3 Local dictionaries 187
10.4 Auteursherkenning 187
10.5 Conclusies 188

Bibliography **188**

List of Figures

1.1 Taxonomic tree for document representations [Lancaster, 1977] . 5
1.2 Taxonomic tree for retrieval techniques [Belkin and Croft, 1987] 5
1.3 The flat text model 9
1.4 The hierarchical text model 10
1.5 The network or hypertext text model 11
1.6 Krippendorfs schema for text analysis 13
1.7 Differences between database management system and IR-system 14
1.8 Document- and keyword-vectors 16
1.9 The classical model of information retrieval 17

2.1 An archive of clay tablets 24
2.2 A document inside its 'authenticity' wrapper 25
2.3 Books on chains in a medieval library. 29
2.4 Example of ICONCLASS signatures. 32

3.1 The Peek-a-boo or 'optical coincidence' system 38
3.2 Notched edge cards and application 39
3.3 Precision and recall 44
3.4 PR-graph for Table 3.2 46
3.5 Baseline effectiveness as function of precision and recall 50
3.6 Hypergeometric distribution for 0-200 relevant documents on databases of different sizes. 51
3.7 Bottom-up and top-down in the SCISOR system 54
3.8 A question in the SCISOR system 55
3.9 Frames in the German TOPIC system 57
3.10 AIR neural network 61

4.1 Conceptual model of IR 64
4.2 Classical model. 64
4.3 The relation between the word frequency and the rank order [van Rijsbergen, 1979]. 75
4.4 Non-orthogonal vectors. 79
4.5 Positioning vectors in keyword space 80
4.6 Clustered file organization 82
4.7 The document vector model 88

5.1 The classic model of information retrieval 92
5.2 Average number of terms spanning 2-8 docreps. 103

5.3 Average and lowest frequency for terms spanning 3-8 docreps. . 105
5.4 Precision vs. recall in CLARIT (left) and TOPIC (right) 108

6.1 tf.idf weights of sentences in a single document 114
6.2 Information values of the Dv in SACJ 118

7.1 Hypergeometric distribution for 0-100 relevant records. 141

8.1 Topology for a stylometric neural network (Matthews and Merriam) 160
8.2 Average accuracy on comparing frag. 23 with all other fragments169

List of Tables

1.1 Genres from the LOB corpus 7
1.2 Genres as used by Pieper 7
1.3 Text properties as used by Pieper 8
1.4 Text properties as used by Biber (slightly abridged) 9

2.1 Farradane's operators 31
2.2 Part of a thesaurus 33

3.1 Various manipulations of precoordinate indices 40
3.2 Recall-precision after judging n documents in linear order (Salton/McGill:1983, p. 166) 45
3.3 A topic in the RUBRIC system (indentation added) 59

4.1 Words with lowest and highest Poisson-overestimation from the Reuter corpus. 70
4.2 Words with lowest and highest discrimination value computed for different word weights 72
4.3 Lowest and highest (average) *atc* weights, sorted from low to high 74
4.4 Keywords in three kinds of spaces 78

5.1 A topic from TOPIC system (indent added) 97
5.2 Result of CLARIT indexing 98
5.3 Original classification of document 140 101
5.4 Final document representations in TOPIC and CLARIT 102
5.5 Test results for TOPIC 106
5.6 Test results for CLARIT 107

6.1 Descriptives for SACJ. 115
6.2 Correlation coefficients for SACJ. 120
6.3 T-tests for SACJ. 122
6.4 T-tests for average sentence-weights (SACJ). 123
6.5 Word categories and average weights for SACJ 124

7.1 Precision and recall for classification with C4.5 138
7.2 Breakeven points and accuracy with associated precision and recall for VSM 138
7.3 Base line performance of local dictionaries with cosine 140

7.4 Nine highest scoring features for gold according to weighting method . . . 143
7.5 Breakeven points for Reuters English Data . . . 145
7.6 Breakeven points for using a GA-produced query. . . . 146
7.7 Results for classification on SVD with cosine . . . 147
7.8 Variations on the computation of the tf.idf . . . 149

8.1 Classification results on stemmed and unstemmed articles (from Littin, 1995). . . . 153
8.2 Confusion tables from K* (class E) . . . 165
8.3 Variations . . . 166
8.4 Differences between the experimental databases classified by IB4. Asterisks indicate the .99 confidence level (T-test). . . . 167
8.5 Cross-table of three fragments by Austen and two by Brontë, showing averages from a ten-fold classification test. Method: E. 167
8.6 Results of direct classification in K* of two fragments by Austen and one by Brontë after training on Pride and Wuth1 . . . 168
8.7 Tenfold, KS, IB4 federalist. . . . 169

9.1 Classification results on stemmed and unstemmed articles (from Littin, 1995, abridged). . . . 174
9.2 Cross-table of three fragments by Austen and two by Brontë, showing averages from a ten-fold classification test. . . . 180
9.3 Highest scoring features for documents from the class 'gold' according to three different weighting methods . . . 182

Dankwoord

Het is op de Katholieke Universiteit Brabant de gewoonte dat de verdediging van een proefschrift wordt geopend met een op een gebed lijkende formule. De kenners stellen vast dat het hier gaat om een politiek correcte versie van het traditionele katholieke gebed "Veni, Spiritus Sanctus, et illumine sensus et corda nostra. Sedes Sapientiae, ora pro nobis"[1].

Bij het aanvragen van de verdediging heb ik het verzoek gedaan om deze verwaterde Nederlandse formule achterwege te laten en daarvoor de orginele Latijnse tekst te nemen, zoals die tijdens mijn middelbare schooltijd op het Gymnasium van de Paters Augustijnen te Eindhoven voor en na elke les werd gebezigd. Het is voorbarig om uit dit verzoek conclusies te trekken aangaande mijn religieuze overtuiging; ik wilde hiermee echter mijn dank en respect betuigen voor wat zij mij hebben bijgebracht: de liefde voor de klassieke en klassiek-christelijke cultuur en tradities. Het kwam dan ook als een teleurstelling voor mij dat voor deze aan tradities en symbolen verder toch zo rijke zitting, de opening met het "Veni, Spiritus Sanctus..." door de Rector Magnificus niet kon worden toegestaan. Mijn dank aan de Paters Augustijnen en met hen aan allen die vroeger door onderwijs en onderricht hebben bijgedragen aan het feit dat ik nu tot de verdediging van mijn proefschrift ben toegelaten moet dus op deze, iets prozaïscher manier tot uitdrukking worden gebracht.

Speciale dank, hoewel niet zonder Typografische Knipoog, gaat natuurlijk in de eerste plaats uit naar mijn promotor, Harry Bunt, en mijn co-promotor, Walter Daelemans. Jarenlang hebben zij de broedende kip zo weinig mogelijk gestoord; de merkwaardige resultaten die er dan af en toe onderuit kwamen werden echter met zorg opgekweekt tot artikelen die de toets der wetenschappelijke kritiek konden doorstaan. Als ik Harry's inbreng zou moeten karakteriseren, dan is het met het Engelse gezegde "Perfection is in small matters, but perfection is no small matter" en dat ik deze *perfection* wel nooit zal bereiken is niet zijn schuld. Zijn aansporingen tot wetenschappelijke nauwkeurigheid zijn echter op vruchtbaarder bodem gevallen dan hij wellicht zelf beseft en ik hoop er nog jarenlang de vruchten van te kunnen plukken. Walter is de man die mij in de grondbeginselen van de kunstmatige intelligentie heeft ingewijd en het zou hem waarschijnlijk verbazen hoezeer mijn wereldbeeld daardoor vervolgens is beinvloed. En zijn speciale talent om een door mij met veel omhaal

[1] Kom, Heilige Geest en verlicht ons gevoel en onze harten. Zetel der Wijsheid, bid voor ons.

van woorden trots gepresenteerde theorie met een enkel zacht uitgesproken zinnetje volledig onderuit te halen zal in de komende jaren in Tilburg node worden gemist.

In de veel te lange tijd dat ik met dit proefschrift bezig ben geweest hebben mijn collega's altijd een warme belangstelling opgebracht voor de voortgang van het Magnum Opus. Dat die warme belangstelling met het verstrijken der jaren niet altijd meer even eerbiedig onder woorden werd gebracht mag hen niet worden aangerekend. Met name Elias Thysse wil ik hier noemen, maar ook Jakub, Antal, Ko, Bertjan en buiten het werkverband Roeland van Hout, Pieter Nieuwint en al degenen die met elkaar de microcosmos van de faculteit der letteren vormen. En de studenten, *bien étonnés*, maar achter mijn gemopper "dat ze er dit jaar weer helemaal niets van hebben begrepen..." leeft het besef hoezeer het dagelijkse werken met jonge, intelligente mensen een voorrecht is.

Geen deel uitmakend van onze faculteit, maar niet geheel toevallig op dezelfde verdieping residerend, bevindt zich het Infolab van de economische faculteit. Ook de mensen van het Infolab, voorop natuurlijk Jeroen Hoppenbrouwers ('Hoppie' voor ingewijden), zijn altijd meer dan behulpzaam geweest als ik weer eens problemen had met LaTeX of andere Unixvragen had; ook naar hen gaat mijn dank uit.

Besonderen Dank auch an Frau Doktor Karin Krüget-Thielmann, die Teile dieses Textes gelesen und viele hilfreiche Bemerkungen dazu gemacht hat. Furthermore I want to give special thanks an Dr. Derrick Kourie form the South African Computer Journal, whose unflagging interest in my work helped me over the last hump. And let us not forget that Finnish student who created the Linux operating system and revived the magic of computing.

Niets van dit alles was echter mogelijk geweest zonder mijn vrouw, Xandra, en mijn ouders. Het is helaas op de volgende pagina's niet zichtbaar met welke zeer praktische suggesties Xandra soms heeft bijgedragen aan de inhoud ervan, maar ik kan de lezer verzekeren dat het er heel wat waren. En het zijn natuurlijk mijn ouders geweest die nooit hebben getwijfeld aan het feit dat deze dag ooit zou komen - en dus kwam hij er!

Waalre, augustus 1999

Hans Paijmans

Chapter 1

Introduction

1.1 The document vector model as basis for a taxonomy

The core of this dissertation is formed by four published papers, concerned with different aspects of information retrieval and text classification. These papers form the chapters 5 through 8. Important issues in these papers are the comparison of IR-systems, designed according to different models of information retrieval; the identification of text passages that are rich in information; the reduction of the number of features for text classification, and the recognition of authors through the detection of patterns of lexical cohesion.

These are rather diverse problems, and each paper can indeed be read on its own. Chapter 5 is concerned with the problems that emerge when two IR-systems that follow different models are to be compared. We argue that these problems can be resolved by mapping the document representations of both systems back to one-dimensional document vectors. Chapter 6 considers the question whether passages in texts, that are particularly rich in information, can be recognized using positional information. We show that this, somewhat surprisingly, is not the case. In chapter 7 we continue research initiated by [Apté et al., 1994a] on the selection of keywords for document classification purposes. Here we find that, when a different weighting scheme is used than that proposed by Apté, the number of keywords contributing to the vector length can be drastically reduced, for instance from a hundred to sixty or even as few as twenty keywords. Finally, in chapter 8 we successfully apply the criterion of lexical cohesion to the problem of author recognition, where the feature vectors in this case do not represent documents but sentences, and the features themselves do not contain keyword weights, but information on the cohesion between subsequent sentences.

What connects these four papers is that they all apply the translation of text into feature vectors and the comparison of these vectors. The literature in information retrieval and text classification abounds with models, strategies and algorithms dealing with these issues. A multitude of techniques have been proposed, that seem not to cause substantial performance differences when applied in IR systems.

Let us take an imaginary IR- or text classification system. It probably is centered around a representation of the document as a collection of keywords, and this already poses various questions: How are the keywords selected? After they have been selected, how are they stored in the system? What weights are used, if any, to indicate their relative importance?

Even if the document representation can be created by widely different techniques, these do not in themselves pose real constraints for the application of similarity functions to compare documents, or to compare documents with a query, and this introduces another set of questions: How to compare the representations of documents? How to select the documents that are returned to the user? A Boolean system would divide the collection of documents into two classes, relevant and not relevant, and is based on a document representation where the keywords have binary weights (0 and 1). But this is not necessarily the case: the fuzzy retrieval model, based on fuzzy logic, is a submodel of the Boolean model and in this model other than binary word weights are used and the output is ranked, not divided into two classes. Probabilistic models also produce non-binary weights but they also need relevance judgments and might therefore very well be classified under the relevance feedback model. The vector space model also can use non-binary weights, but if these weights are produced through application of the probabilistic model, how should the IR-system be classified? In other words: there are too many factors influencing the behaviour of a system than that one can conveniently attach the name of a single model to its design.

This has brought us to the realization that the important models in the information retrieval literature do not adequately describe complete IR-systems, and that on the other hand the schematic descriptions of IR processes that do (see e.g. section 4.1), are too general to be of real use in the classification of IR-models. What is needed is a generic model that forms a basis for a variety of more specific models and in which the combination of choices that have been made for any particular IR-system can be described. On the basis of such a generic model a taxonomy of models can be built, in such a way that the attribution of models to implemented systems is cumulative rather than exclusive. We think that we have found a generic model in the *document vector model*, under which most other popular models in IR can be subsumed.

In 1984 [Smith and Warner, 1984] published a tentative taxonomy of document representations only. The purposes of their taxonomy were firstly, to relate new work to previous work, and secondly, to detect voids and to suggest new areas of research. We may add that a judicious arrangement of models will also help to better understand the various models, individually and in relation to each other. They present a taxonomy tree from [Lancaster, 1977] (Figure 1.1). The categories that are presented in this tree are neither mutually exclusive nor exhaustive, and at least one subdivision, for structured document representations, should be added for completeness. In 1987 [Belkin and Croft, 1987] published a taxonomy that tries to classify retrieval techniques in relation to each other (Figure 1.2). Note that in the fig-

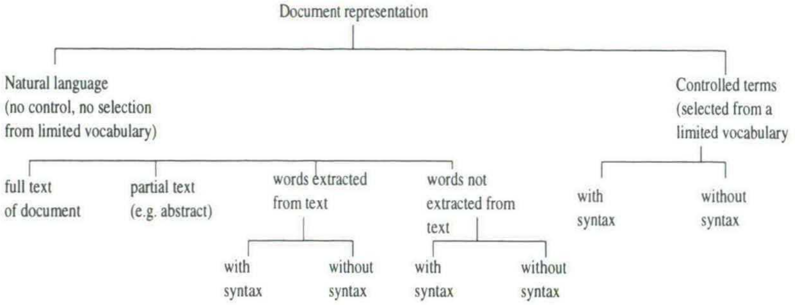


Figure 1.1: Taxonomic tree for document representations [Lancaster, 1977]

ure no reference is made to the relevance feedback model because "...Relevance feedback techniques are not considered retrieval techniques by our criteria. Rather they are used to refine the request model..." (p. 123).

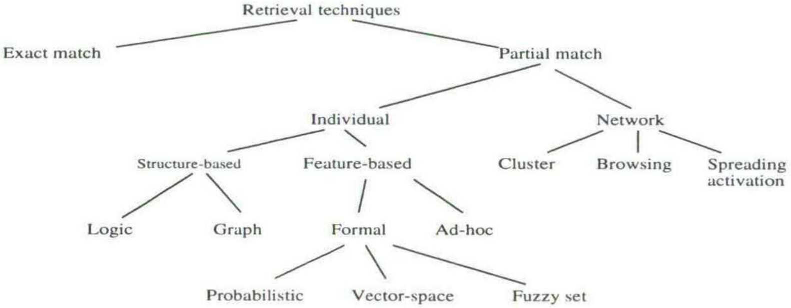


Figure 1.2: Taxonomic tree for retrieval techniques [Belkin and Croft, 1987]

In a fully satisfactory taxonomy it should be possible to describe the essential characteristics of every individual system. This means that such a taxonomy should consider the following aspects of IR-systems:

1. Which information from, or part of the document is presented to the system (the document surrogate) and how it is obtained.
2. By which metric the features that find their way into the system are selected, and in what form they are stored.
3. The form in which the features are stored, more in particular the manner in which the weights are obtained.
4. The form in which the features are presented to the similarity functions.
5. The similarity functions itself.

6. The structures that govern the subsequent processing of the results of the similarity functions (e.g. *relevance feedback*).

At the end of chapter 4 we will return to these issues and show how the document vector may act as a basis for this description.

1.2 Expository text and other text types

The main subject of this study is expository text, more in particular those properties of expository text that can be used to recognize the informational content of the document for retrieval or classification purposes. This latter activity, classification of texts, may be be extended to non-expository texts, hence the inclusion of a chapter on author recognition (chapter 8), but the main focus will be on expository texts as the vehicle of concepts and ideas.

The term 'expository text' is used by the various contributors of [Britton and Black, 1985]. Unfortunately the definitions that are given are negative, rather than a positive description of its properties. Expository text is considered to be more or less equal to non-narrative prose. [Hahn, 1990] uses this term matter-of-factly to refer to full-text technical reports, letters, memos, and magazine articles. [Rau and Jacobs, 1988] name the fact that narrative texts have a 'plot' while 'expository' texts do not need to have one, as one of the differences.

In the field of information retrieval, there has always been a tacit consensus of expository text. Indeed, we may wel use a *de facto* definition of 'expository text' as text that is the target of information retrieval activities. This is because the concepts that are 'exposed' generally are the reasons for wanting to retrieve a text in the first place. The traditional targets for information retrieval are databases of scientific texts, press archives and the archives of law offices. Text genres such as poetry or novels, that do not explictitly 'expose' concepts are not likely to be included in retrieval systems.

This last statement presupposes the existence of systems of genres and classifications of texts. As an example we give the genres that are used as classes in the LOB corpus (figure 1.1), the two columns indicating a first, intuitive interpretation of what may be considered expository text and what not.

Given that there exist differences between texts, and that some genres will sooner be targets for IR activities than other texts, we will want to have criteria to identify such genres in an input stream. Also, when a document base consists of several genres, a user may conceivably restrict the set of potential useful documents to some genres and exclude others.

Earlier research has been concerned with the problems of genres and typologies; e.g. Biber [Biber, 1989], [Biber, 1993] for English text and Pieper [Pieper, 1979] for German texts. We mention these authors, because both use statistical methods to identify the various genres in the respective languages. However, they approach their typologies from different directions. Pieper constructs a number of hypotheses on genres in the German language (Figure 1.2, again with an intuitive interpretation for expository and non-expository

expository text	non-expository text
Academic prose	General fiction
Official documents	Mystery fiction
Bibliographies	Science fiction
Press reportage	Adventure fiction
Skills and hobbies	Romantic fiction
Popular lore	Humor
Religion	

Table 1.1: Genres from the LOB corpus

non-expository text	expository text
Hörspiel	Wissenschaftliche Texte
Drama	Allgemeine Gesetzetexte
Diskussion	Zeitung: Agenturberichte
Roman-nichtdialog	Zeitung: eigene Berichte
Briefe	Zeitung: Sportberichte
Zeitung: Feuilleton	

Table 1.2: Genres as used by Pieper

text) and uses variables from morphology and syntax to study the differences between these genres, which she calls *clines*[1].

The work by Biber, on the other hand, is aimed at finding the differences between speech and writing. He uses the texts from the LOB corpus as examples of written texts, adding two more genres consisting of written personal and professional letters. The spoken texts were taken from the London-Lund corpus. He then applies factor analysis to a number of properties of the texts (see table 1.4) to create groups of texts that are maximally different on all dimensions. The resulting groups are called *registers.*

As the tables 1.3 and 1.4 show, Pieper and Biber use similar quantitative criteria, that are relatively easy to identify and measure.

Perhaps Biber may be criticized for selecting his texts from a corpus that already had a class system imposed on it: one might argue that the individual texts in the corpus were chosen to fit these classes. This would not invalidate his main thesis that there exist systematic syntactic and morphological differences between speech and writing, but it would make the results with respect to class differences within the LOB corpus less convincing. However, the outcome of his experiments as far as relevant to the identification of a genre of 'expository text' runs parallel to those of Pieper.

The results of both Pieper's and Biber's experiments show a clear cluster of three genres that display strong similarities for almost all dimensions. In the German corpus these are scientific texts, laws ('Allgemeine Gesetztexte') and newswire ('Zeitung: Agenturberichte'); in the LOB corpus these are academic prose, official documents and press (press reports, press editorials and press

[1] The word 'cline' is more often used in biology and is used by Pieper to emphasize the gradual transition of one class into another.

Wörter/Satz	Imperativformen/Satz
finite Verben/Satz	Nomina/Satz
finite Hauptsatzverben/Satz	Nomina mit vorangehender Genitivkonstr./Satz
finite Nebensatzverben/Satz	Nomina mit folgender Genitivkonstr./Satz
Präsensformen/Satz	Nomina mit der Endung -ung/Satz
Präteritumformen/Satz	Nomina mit der Endung -heit bzw. -keit/Satz
Perfekt formen/ Satz	Nomina mit der Endung-ismus/Satz
Plusquamperfektformen/Satz	Nomina mit der Endung -ion/Satz
Futur-I-Formen/Satz	Nomina mit der Endung -ik/Satz
Futur-II-Formen/Satz	attributive Adjektive/Satz
Indikativformen/Satz	Demonstrativpronomina/Satz
Konjunktiv-I-Formen/Satz	Possessivpronomina/Satz
Konjunktiv-II-Formen/Satz	Ordinalia/Satz

Table 1.3: Text properties as used by Pieper

reviews). We may therefore assume that indeed there exists such a thing as 'expository text' and that it coïncides with the traditional document bases of information retrieval.

1.3 Inside the document

The work mentioned above is concerned with the creation of classification systems in which complete texts may be positioned. A next step would be to devise methods for identifying smaller units in the text and create a classification for those units.

This is a fundamentally different proposition. Individual texts may for classification purposes be considered as items that may be directly compared, but this is not necessarily true for the parts of which individual texts are composed. On the contrary: depending on the text model that is used, many structures that are identified as text components fulfill widely different roles in the text, even if they have a similar shape.

1.3.1 Text models for information retrieval

For IR purposes three text models are recognized (by a.o. [MacLeod, 1990]):

1. The flat text model.

   In this model, as supported by STAIRS[2], a text is represented by a structure consisting of two parts: a set of attributes and the text proper. The attributes consist of bibliographical or book-keeping type data; the text itself is divided into named subcomponents: the paragraphs. Only one paragraph of each type is allowed in the document. Paragraphs are not themselves subdivided in smaller parts, except for the individual words. The division into paragraphs enables the user to limit searching

[2] IR system marketed by IBM [IBM, 1976].

past tense	THAT verb complements
perfect aspect verbs	THAT add. complements
present tense	WH clauses
place adverbials	infinitives
time adverblals	present participial clauses
first person pronouns	past participial clauses
second person pronouns	past prt. WHIZ deletions
third person pronouns	present prt. VHIZ deletions
pronoun IT	THAT relatives
demonstrative pronouns	WH relatives
lndefinite pronouns	sentence relatives
DO as pro-verb	adv. subordinator
WH questions	prepositions
nominalizations	attributive adjectives
gerunds	predicative adjectives
nouns	adverbs
agentless passives	type/token ratio
BY passives	word length
BE as main verb	conjuncts
existential THERE	downtoners
	hedges

Table 1.4: Text properties as used by Biber (slightly abridged)

and retrieval to selected parts of the document (so-called *field-control*). Documents with a similar paragraph structure can be organized in collections.

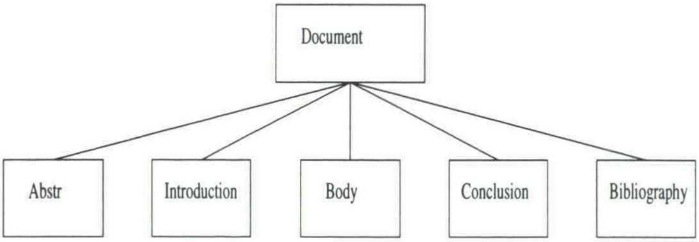


Figure 1.3: The flat text model

This model is called the 'flat' model. Most commercial IR systems use the most simple form of this model, where indexed documents are not even divided into *fields* or paragraphs, and the book-keeping type data may or may not be present.

This organization is not only suitable for natural language documents, but also for forms or even ingenious combinations of forms and natural language (see also [Paijmans and Verrijn Stuart, 1982]).

Several attempts have been made to translate the IR model into the relational database model (e.g. for Oracle [Bantzer and Toussaint, 1987], [MacLeod and Reuber, 1987]), but this approach has never become popular either in research environments or in commercial applications.

2. The hierarchical model.

   The flat model has been criticized for being insufficient for advanced retrieval activities. Mark-up languages, such as ODA or SGML, allow for a hierarchical model dat describes the typical document much more closely. Thus a document may consist of sections that in turn are divided into subsections and again divided into paragraphs or special constructs such as item lists. Many of these constructs have a title or header that is marked as such, including the document itself.

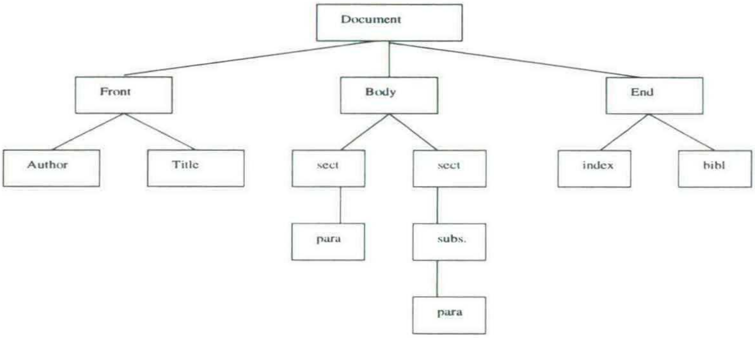


Figure 1.4: The hierarchical text model

   Although this model allows for a more precise description of documents, those descriptions rapidly become so complex that no suitable language for retrieval has been developed. One of the underlying reasons is that there is no model for matching the topicality of a section, or field, with its function, although there have been attempts to at least match an informational weight with individual parts of a text ([Kieras, 1985], [Paijmans, 1994]).

3. The network model.

   A different text model has been described by [Conklin, 1987]. This is the hypertext model, that in the eighties already had made a strong impact on computerized manuals and help systems (e.g. the help function of Microsoft Windows) and now has become the model for the World Wide Web. In a hypertext document it is possible to establish links at will between different parts of a document, or between parts of different documents, even on different computers. This makes it possible to browse documents in an order that differs from the linear order in which documents are usually read.

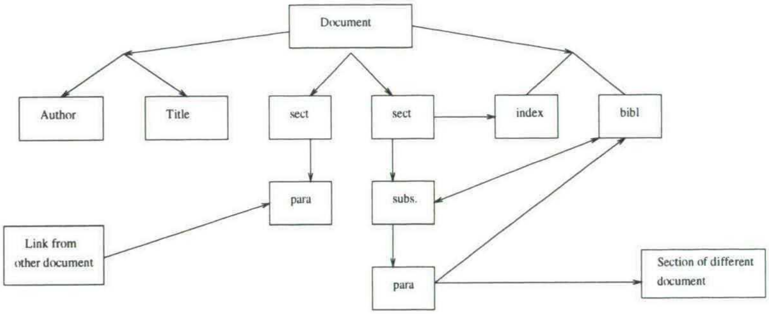


Figure 1.5: The network or hypertext text model

Non-linear reading was already done in the pre-computer era. Most factbooks are not meant to be read sequentially, but to pick sections that are relevant for a specific purpose. Teaching material often has provisions for skipping chapters or to return to them after a while. But computer technology, notably through the mouse-oriented interface, has made it easy to point at words or sentences and activate links that cause a different part of the document (or parts of a different document) to be displayed.

1.3.2 Structures in texts

Text may be seen as an aggregate of structures: from sequential structures of tokens (characters, words) and sentences, to the hierarchical or network structure of chapters and possibly volumes, and all the other structures that may be found in the typical Table of contents. Using these components, a text may be described in any of the three models mentioned above.

A text is also a collection of meaningful clusters of statements and facts that may be combined to form information or knowledge. Even when browsing through a typical corpus that is composed of sentences or pararaphs taken out of context, one may find many such facts and statements, isolated from its information/knowledge context. Nevertheless they can often be recognized as such. These smaller statements, or *propositions*, and the larger structures that can be recognized will be called *document knowledge representations* if they are made explicit and stored.

In later sections we will look into the methods of discovering the topicality or 'aboutness' of a text, but first we have to pay attention to the attempts to make the rhetorical structure of a text explicit. Much research has been done in this area, but no substantial progress seems to have been made towards automatic extraction of those representations except for very simple structures such as lists of keywords and collocations. Also, such research often seems to concentrate on the progress of dialogues, rather than on the topicality of the text. This especially true for theories such as Rhetorical Structure Theory [Mann and Thompson, 1987] or theories on goals, plans and intentions [Grosz and Sidner, 1986]. Older, but still valid are the observations of [Schank and Abelson, 1977] on frames and acts, giving birth to attempts to convert sentences into primitive acts, thus effectively paraphrasing these sentences. This usually involves the creation of larger and lower-level descriptions than the original sentences. Working in the opposite direction we mention Lehnert's 'abstraction units', that convert descriptive nets into smaller, less detailed and higher level ones [Lehnert, 1981].

These approaches all have in common that the structures they use are difficult, if at all possible, to formalize in computer programs, and that where such attempts have succeeded, they ran full tilt into the scaling problem. It was not untill well in the nineties that research returned to less ambitious schemes, such as recognizing topical boundaries in text with relatively simple means like lexical cohesion [Morris and Hirst, 1991], [Kozima, 1994]), or text tiling using frequency-based word weights [Hearst and Plaunt, 1993], [Hearst, 1993a].

This return to the quantitative approach brought together text classification and the recognition of stylistic features as used for stylometric purposes, in particular for author recognition (see chapter 8).

1.3.3 Topicality

A rather different classification of texts is that accordig to content, aptly called *content analysis*. "Content analysis is a research technique for making replicable and valid inferences from data to their context" [Krippendorf, 1980], p. 21. Note that the word 'content' in this context refers not only, or not even in the first place, to the topicality, but also to emotional, rhetoric or other categories. For instance, the German sociologist Ertel [Ertel, 1976] classified texts according to the dogmatism displayed by the author, by counting words like "always", "whenever" or "never", which indicate a dogmatic state of mind in the writer, or "often", "sometimes" and "occasionally" as indicators of a more tentative state of mind.

Krippendorf points out that messages do not have a single meaning, i.e. that there exist different contexts for the same collection of textual data. Also, meanings do not have to be shared, but they are always relative to the sender and the receiver. Mass communication of course is the traditional home of this field. In figure 1.6 content analysis is related to other data analysis activities such as laboratory experiments, field experiments and information retrieval. The axes on which these activities are measured are the *degree of obtrusiveness* (the chance that the assessment of a phenomenon influences the phenomenon), the *structuredness* of the data, and the *context sensitivity* of the data.

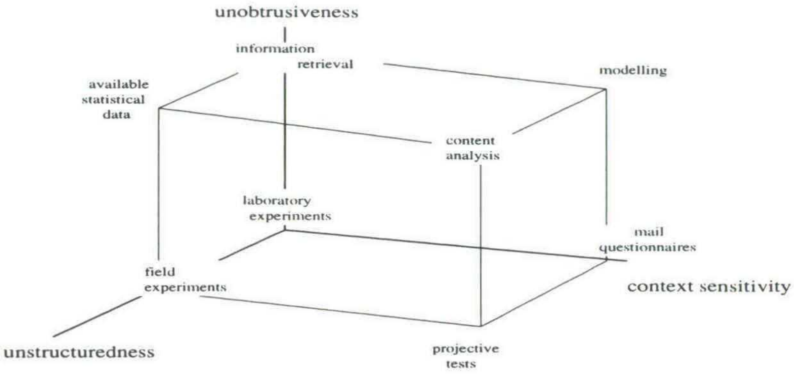


Figure 1.6: Krippendorfs schema for text analysis

At first sight the place of IR as an unobtrusive, not context sensitive, but structured activity, may be surprising. Typically the data on which IR operates is unstructured, and the context of a text or document is very relevant. However, Krippendorf argues that "retrieving information from a databank is prestructured by the formal requirement of manipulation and storage". The low score of IR on context sensitivity is brought about by the fact that the data in IR are dissociated from the symbolic meanings that the subjects (authors) involved may have had. Nevertheless, we feel that the distance between content analysis and IR is smaller than the schema suggests.

Abstracts and extracts

In chapter 4 we will define the various transformations that a document may undergo in the course of classification and retrieval in more detail; here we will give some elementary explanations.

Taking a typical IR system, we can identify the *document surrogate*, from which keywords are extracted, the *document representation*, which is the collection of keywords in the system that point to the document, and the *on-line document* that is displayed by the system at retrieval time. In the information retrieval literature the division between document surrogate, document representation and on-line document is not always clearly made; in particular some confusion exists about the exact place of abstracts in the scheme of things.

Let us first agree that abstracts and extracts are two different notions. Although both serve to represent the original document,"...so that readers may decide, quickly and accurately, whether they need to read the entire document." ([ANSI, 1979], p.1), they are generated in very different ways. An extract is a part or a collection of parts of the original, selected to represent the whole, and it consists of selected sentences from the original. The automatic construction of extracts is relatively easy. A number of methods is available by which to judge the importance of words and hence sentences [Paice, 1990].

An abstract, on the other hand, is an independent description of an 'internalization' of the original document. Although the ANSI definition describes an abstract as "...an abbreviated, accurate representation of the contents of a document.", we want to introduce the word 'internalization' in the definition, because it puts emphasis on the fact that a processing and reformulation of the document is a prerequisite for the generation of an abstract, as opposed to an extract. Examples of this internalization may be found in the work of Lebowitz [Lebowitz, 1983] or in the German TOPIC ([Hahn and Reimer, 1987], [Hahn, 1990]) that is treated in more detail in chapter 2.

1.4 IR and database management systems

In the text models described above we already mentioned the dichotomy between the bookkeeping-type data, that can be stored in any data base management system, and the content of the text proper that will ultimately have to find its way into the IR-system.

The next issue that we will address is the question of what exactly is the difference between a (relational) data base management system and an information retrieval system. There are many very good database management systems around, so why can a librarian not just take any well-established relational database management system, such as Oracle or Ingres, and work with that?

The classical data base management system (DBMS) is shown in the upper part of Figure 1.7. To make it possible to record an object we analyse the relevant properties of that object and create corresponding fields in the database record. If the object is a person, we may create the fields (or 'attributes'): *name*, *address* and *telephone number*, or for a book the fields *author*, *title* and *publisher*. Obviously the name 'Smith' as a value in the author field indicates a different role for Mr. Smith than if the name occurred as a value in the publisher field.

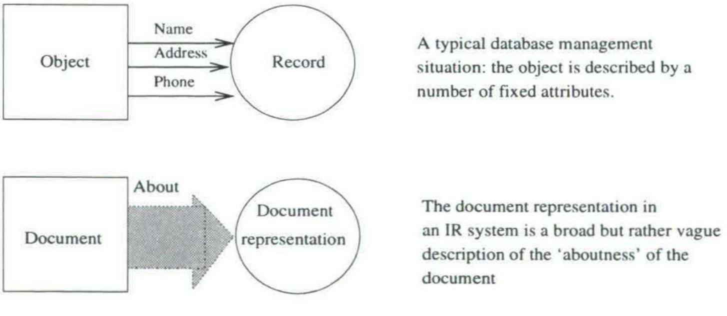

Object
Name
Address
Phone
Record
A typical database management situation: the object is described by a number of fixed attributes.
Document
About
Document representation
The document representation in an IR system is a broad but rather vague description of the 'aboutness' of the document

Figure 1.7: Differences between database management system and IR-system

In traditional, relational data base management systems, such attributes are collected in the tuples of a relation, or table, and subjected to a number of normalizations. By combining such tables, complicated objects and actions from the real world can be mapped into the file and record structure of a *relational* DBMS [Date, 1983]; when the structures are combined at the level of records rather than that of tables, the name *hierarchical* or CODASYL DBMS is preferred ([Olle, 1980]).

This translation or mapping from the object to the attributes of a record is much more difficult when the object to be mapped is the topicality of a text, such as a book or article. The relationship between form (strings of characters) and content is much less clear than in the case of e.g., digits and numbers. A single word may have widely different meanings or a single concept can be adressed by different words. The meaning of words can be influenced by context, but this context is not necessarily the direct textual context of words within a certain distance of each other.

Therefore, a marked difference exists between the datatype or datatypes of the text proper and that of the bibliographic data. The bibliographic data may easily be modeled by means of a database record with fixed attributes (or fields), but in the text there are no clear markers or fixed positions corresponding to potential attributes. Indeed, the detection of strings, called "cue strings" or "cue phrases" that might serve as such markers has been an ongoing concern in IR ([Paice, 1990]).

An obvious solution is to create a single attribute with a name like *contents* and to use it to store keywords or a short description of the contents of the document. An inverted file of these keywords can be created and retrieval then is effectuated by matching the words from the query with the words in the inverted file, creating sets of records that contain single keywords and manipulating these sets with Boolean operators, such as AND, OR and NOT. This approach to retrieval is called the *Boolean model.* In commercial IR systems it is the prevalent model (but see section 1.4.4 of this chapter for comments on its performance).

Storing several data items in a single attribute field conflicts with some of the most fundamental concepts of data base management: most models, in particular the relational model, do not allow fields to be filled with more than a single value item.

Therefore IR often works as if every keyword in the database is a separate attribute, or feature, that for every document can have the value 0 or 1, according to the occurrence of that keyword in that document (as in Figure 1.8), or other values, such as the *term frequency* of the keyword in the document. These attributes are considered to be *symmetric* or *orthogonal,* i.e. there are no dependency relations between individual keywords. The horizontal rows correspond with the documents, and in the relational database model are called 'tuples' or records.

In this model the number of attributes in the document representation can run into the tens of thousands and, moreover, such databases are *sparse,* i.e. almost empty: the 'ones' are few and far between. It is easy to see how such a

	Barcelona	Art	History	Gaudi	Dogs	Cats	Horses	Cars	Mathematics	Games	Glass	Wine
Doc.1	1	0	0	0	0	1	0	0	0	0	1	1
Doc.2	0	1	0	1	0	0	0	0	0	0	0	0
Doc.3	0	0	0	0	1	1	0	1	0	0	0	0
Doc.4	1	0	0	0	0	0	0	0	0	0	0	0
Doc.5	0	0	1	0	0	0	0	0	1	0	0	0

Keyword-vector

Document-vector

Figure 1.8: Document- and keyword-vectors

Table can be inverted in such a way that the keywords form the tuples and the documents the columns, hence the name *inverted file* that is sometimes used for an index. This model is the predominant model of information retrieval used in traditional as well as in present-day research. Other related models concentrate on the comparison of these vectors (e.g the *vector space model*), or on the methods by which the weights in the vectors are arrived at, such as the frequency based models.

Of course there are other approaches to the modelling of the document representation. For instance, in chapter 3 we will see how the contents of a document can be translated into a conceptual dependency representation and incorporated in a knowledge base. But for the purposes of this study we will consider the document representation to be a feature vector of symmetric keywords or, in some cases, of other attributes and refer to it as to the *document vector model.*

In the sections 3.2, 4 and 4.3.3 we will have more to say on the details of the computation of feature weights, the comparisons of the vectors and the measuring of the performance.

1.4.1 Information retrieval as a classification task

The function of any IR system is to extract relevant items from texts; translate them into the symbols of an *index language*; arrange these symbols so as to improve accessability and offer them to the prospective user (see Figure 1.9). Also, the query coming from the user must be translated into the same index language and similarity functions must be provided to compare documents and queries and rank the documents according to the outcome of such comparisons.

If the express goal of the system is to lead the user to the document itself, we will call this an IR system in the narrower or proper sense, also referred to as a *document retrieval system.* Sometimes the user may decide that his information need is satisfied without accessing the document itself. If it is among the goals of the system to support this, then we call this an IR system in the wider sense, or a *data retrieval system*. An example of the former is STAIRS [IBM, 1976], of the latter SCISOR [Rau et al., 1989], [Rau and Jacobs, 1990]. For a more detailed overview of IR systems and general terminology, see the end of chapter 2 and chapter 3.

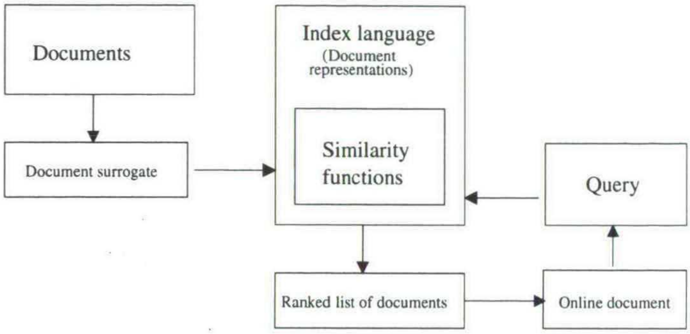


Figure 1.9: The classical model of information retrieval

Depending on the similarity function that is used, the system may return a weak ordering in *relevant – not relevant*, or it may return a strong ordering of the documents, where the estimated degree of relevance is a point on a continuous scale. In the first case, the IR system is in fact a content-based classification system that accepts a set of criteria (the query) and divides the documents in two classes: those relevant to that query and those that are not. Of course, any strong ordening can be converted into a weak one by applying a cut-off threshold or by invoking some other criterion, not the least among which is the decision by the user.

1.4.2 Classification and categorization

We already noted that a weak ranking of documents in those that are relevant to a query and those that are not, is the same as a two-class classification. Indeed, many of the tools used in text classification are similar to those in vector-based information retrieval and both rely on the comparison between document vectors and a query- or example-vector.

The difference between the two is that in IR the query is the translation of an information need, whereas in text classification the equivalent of the query is the result of the analysis of texts that belong to the target class and texts that do not. When this involves pre-existing categories the term *text categorization* is preferred.

Both information retrieval and text categorization may be considered as consisting of four main phases ([Lewis, 1992], [Belkin and Croft, 1992]):

1. Indexing: the translation of the contents of the documents into the symbols and structures of the index language, creating for every document a document representation. The factor speed in IR is of less importance than in text classification, because in the latter case large numbers may have to be processed in real time. We want to draw special attention to the *document surrogate*, which is an intermediate stage between the document and the document representation. Often so-called full text indexing in reality operates not on the documents, but on such surrogates, e.g. the abstract, or even the title (see also section 2.2).

2. Query formulation / categorizer formulation. When the user approaches the IR system, his or her[3] information need is also translated into the index language. In the case of text categorization, a similar expression is created that makes the system decide on the class a document should be assigned to: the *categorizer*. The query in IR often is an *ad hoc* occurrence, whereas a set of categorizers may be in use for a long time. This justifies more effort to be spent in expert analysis of potential categorizers, or large scale statistical analysis.

3. Comparison: the query or the categorizer is compared with the document representations and a binary or a graded similarity is computed. Categorization systems will most often require a binary decision; IR systems may allow for both.

4. Feedback / Adaptation: in an IR system the query formulation generally is initiated by a query vector that is constructed by the user. Subsequent modifications of the original query may be done explicitly by the user, adding or culling keywords. In some systems the user may also indicate a set of retrieved documents that come near his information need, and the system then updates the query towards those documents and away from undesirable ones: the *relevance feedback* model. This last model is reported to enhance performance considerably ([Rocchio, 1971], [Lewis, 1991]).

Because of this fourth phase we made special provisions in our somewhat elaborated version of the schematic classical IR model of Figure 1.9 by adding among other enhancements the *on-line document*, a special representation of the document that should enable the user to decide on its suitability for either retrieval or perusance in a relevance feedback cycle. For a more detailed discussion of the scheme and its components the reader is again referred to chapter 4.

1.4.3 The selection of features

The most important but also the most difficult phase in both information retrieval and text categorization is the indexing part, or the selection of the

[3] We will henceforth use the male form in generalizations, rather than indulge in stylistic contortions for the sake of political correctness.

features (generally keywords) that go into the index language.

This has long been the task of a human indexer, who reads the document, or at least the document surrogate, and decided on the terms that would go into the document representation. But human indexing cannot hope to keep up with the mass of documents to be indexed and even if it could, its quality is not much if any better than automated indexing.

The studies of [Cleverdon, 1984], [Lancaster, 1968], [Lancaster, 1976], Salton and many others all point to the following conclusions:

> - if two people [...] construct a thesaurus in a given subject area, only 60% of the terms may be common to both thesauruses;
> - if two experienced indexers index a given document using a given thesaurus, only 30% of the index terms may be common to the two sets of terms;
> - if two search intermediaries search the same question on the same database on the same host, only 40% of the output may be common to both searches.
> - if two scientists [...] are asked to judge the relevance of a given set of documents, the area of agreement may not exceed 60%.
>
> [Cleverdon, 1984].

On the other hand we will see below (section 1.4.4) that the systems that use automated extraction of terms from full text documents do not fare much better.

In the document vector model as described above, the features in most cases consist of word types that are taken from the document. In some cases experiments have been done with *n-grams* of characters or words [Chudacek, 1984], [Teufel and Schmidt, 1988]; in other cases single words and collocations have been used. If the system uses word types, there is a number of possible choices here that can be applied in isolation or combined:

1. The system can store all words in the keyword-document Table in the form in which they occur in the text.

2. The system can filter out function words and low-content words (often called 'stop words'; the list with such words is called the 'stop list').

3. The system can confine selection of keywords to those that occur in a list (*controlled dictionary*). This is the opposite of the filtering out of keywords.

4. The system can bring down the number of items to be indexed by applying some truncation or stemming algorithm. This causes a mapping of several morphologically related words on the same index entry.

5. The system can apply a *word weight* (e.g, the *discrimination value*, see also chapter 4). This weight may be used as a threshold that can cause keywords to be filtered out altogether.

6. The system can apply a *word-document weight* (e.g, the *tf.idf*, see also chapter 4). This weight too can be applied as a threshold that causes certain document-keyword combinations to be ignored.

1.4.4 The Blair/Maron experiment

Although commercial information retrieval systems almost exclusively use the Boolean model, in which documents are either retrieved or not, according to set membership, the scientific community has always harboured doubts as to its performance (see e.g., [Salton et al., 1983], [Fox and Koll, 1988]). Perhaps the most cited article in this respect is the report of Blair and Maron that was published in 1985 [Blair and Maron, 1985].

Blair and Maron for their experiment analysed the activities of legal staff of a large law office that used STAIRS for full-text indexing and retrieval of approximately 100,000 documents, among which the documents that were relevant for a certain law suit. The legal staff, experienced users of STAIRS, persevered every query untill they were certain that 80% of the documents that were relevant to that query were retrieved (a targeted recall ratio of 80%). Blair and Maron then used special techniques, described in detail in [Blair, 1996] to judge the actual recall ratio. The gist of their findings was that even experienced users would not obtain a recall of more than 20-40% , and worse, that they were not aware of the fact.

This should not be surprising. The retrieval of a particular document on a particular keyword is in a Boolean system the product of two probabilities. The first is the probability that de indexer or the indexing system selected that particular keyword as an indexing term for the document and the second is the probability that the user at query time selected that same keyword in his query; the product of these probabilities goes down very rapidly.

1.4.5 The IR paradoxes

Still, after thirty years of research in document representations for information retrieval (or more than a hunderd years if we include Dewey and his disciples), the fact remains that the actual document representation only has a minor effect on the performance of the complete system ([Croft, 1987], [Lewis, 1992]). The same is true for the similarity function that is used. Although Noreault, McGill and Koll in their report on a variety of ranking strategies by weighting keywords and similarity functions ([Noreault et al., 1981]) found an improvement of 20% over random ranking, they also concluded that "While some algorithms were bad, most produced very similar results. No algorithm or approach distinguished itself as being greatly superior to others". More recently Shaw, Burgin and Howell [Shaw et al., 1997a], [Shaw et al., 1997b] computed a low performance standard based on chance for a number of IR experiments, including those of TREC[4]. After comparison of the operational performance of these systems with this low performance standard, they concluded that the effectiveness of those systems often was comparable to drawing documents at random from the database. More details, and definitions of performance, are given in chapter 3.2.

Similar observations caused Lewis to formulate his two paradoxes ([Lewis, 1992]):

- The *Equal Effectiveness Paradox.* It is easy to imagine terrible text representations that would support no better than random classification of documents. It is easy to imagine an excellent text representation that happens to contain a single term indicating exactly the set of documents specified by the current user. Yet all reasonable text representations have been found to result in very similar effectiveness on the text retrieval task.

- The *Perfect Query Paradox.* Most text representations and text retrieval systems in use have the property that, given almost any subset of the database, it is possible to create a request that will be translating in a query retrieving exactly those documents. This is true because, with most text representations in use today, any document can be uniquely identified by a few terms that occur in few or no other documents.

Perhaps a third paradox should be added here:

- The *Indifferent Results Paradox.* One would think that the ultimate test for any retrieval or classification system is the similarity of the set of documents that is retrieved after a query to a predefined set that is considered to contain the correct documents for that query. In fact it is highly doubtful that such a set of correct documents does exist for non-trivial queries or rather, that a single such set exists. Also, the membership of a document in such a set is not a simple binary value, but may range over several discrete or continuous values indicating degrees of relevance. The best one can do is to state that for a particular user with a particular information need a given document may be more or less interesting.

In the last thirty years, that is almost from the beginning of automated information retrieval, the vector model has been explored in detail. The conclusion seems to be warranted that its performance is at least as good as the best manual indexing methods and better than most other models, such as the Boolean model or any of the AI based models described in chapter 3. At the same time the situation at the forefront of information retrieval can be said to

[4] TREC: Text REtrieval Conference. This organization distributes a common test collection with sets of queries for which the relevant documents are known. Hence, systems can be compared and contrasted on the same data.

be far from satisfactory, and seen in the context of Internet and World Wide Web even disastrous ([Lawrence and Giles, 1998]).

We have explored some of the territory of information retrieval, in particular that of the document vector model, looking for ways and means to improve on existing procedures. In the second chapter of this dissertation, we describe how information retrieval emerged slowly as a discipline of its own, together with books and writing until in the twentieth century mechanization was applied, culminating in the use of the computer. In chapter three this concise history of information retrieval is continued and we describe the mechanized and computerized systems, including some examples of artificial intelligence. In chapter four we describe the creation and application of document vectors: this chapter duplicates some material that already was printed in the four published papers.

The first and foremost problem was how to compare the performance of such systems and procedures. We found the document vector model to be a convenient in-between model to map other, more complicated document representations on (see chapter 5).

A major problem in the document vector model, when applied to the complete texts of documents, is the length of the document vector, which is equal to the number of different keywords in the database. This vector may well be several tens of thousands keywords long, and the filtering out of function words and the application of stemming algorithms does not make much difference. Especially in the field of text categorization several attempts have been made to reduce this great number of features. These attempts have been made in two directions: by reducing the sheer number of keywords and by condensing the keywords by means of factor analysis, or related techniques, into a smaller number of new vectors.

In chapter 7 we consider the problem of these long vectors, extending the experiments of [Apté et al., 1994a], [Apté et al., 1994b] with the called *local dictionaries*. Unfortunately, there was again no easy way to compare the results but we have developed an improved strategy to generate these local dictionaries.

A different approach is taken in chapters 6 and 8. In chapter 6 we address the consequences of parts in a document that have different functions and the possibility that some areas or passages have a higher informational content than others. In chapter 8 we leave the keyword as a feature altogether and will try to distinguish between documents on other features. This latter project moves outside the field of *expository text* that always has been the central subject of our interest, but it is included because the tools and methodology is very much like those used in other chapters.

Chapter 2

The retrieval of information from a historical perspective

2.1 Information retrieval and libraries

The discipline of information retrieval is nearly as old as the written word. With the advent of libraries and large document collections much effort was put in systems that enable people to retrieve information from the texts that are stored in such collections. In this chapter we will give a concise overview of the attempts to manage documents for retrieval purposes, from the earliest recorded book collections until the introduction of mechanization and computers.

2.1.1 The Assyrian and Hellenistic libraries

Texts may be collected and preserved in archives or libraries for several reasons. The first and probably earliest reason was the need for a place to store administrative data on food, taxes and other such resources. To this was added the safe-keeping of contracts and laws. As religion in these early communities often was a more stable factor than the worldly power of chieftains and dynasties, the earliest archives and libraries were attached to temples.

With chieftains and dynasties becoming stable institutions of power, one of the first things new administrations do is to start archives of their own, because no administration can survive without good bookkeeping. But soon it seems that the very fact of owning a library becomes a factor in the wielding of power. Emperors and despots start actively collecting texts, even if such texts have no direct bearing on the efficiency of their administration.

This marks the functional dividing point between archives and libraries; while archives share with libraries the function of collecting documents, preserving them and making them available, they employ different principles and management techniques. Archival institutions are *receiving* agencies: they do not *select* their material - their function is to preserve documents as organic bodies of documentation. They must respect the integrity of these bodies of documents and maintain as far as possible the order in which they were created

Figure 2.1: An archive of clay tablets

[Ketelaar, 1997]. Libraries, on the other hand, might be described as collecting agencies, where the *content* of the document is the criterion for the selection.

Originally, library materials were not distinguished from archival records and were preserved in the same places until the mid-15th century and the invention of printing. But it was only with the French revolution that there was for the first time a unified administration of archives that embraced all extant repositories and record-producing public agencies.

The task of preserving the integrity of archival records also posed the problem of how to prevent unauthorized changes. In Mesopotamia, from 2000 BC onwards, it was obligatory for even the smallest commercial transaction to be written down and 'signed' by both parties and the witnesses. Seals therefore were in wide use. However, it was always possible for one or both parties to add a few symbols or to moisten the (unbaked) clay and to remove some symbols. To prevent this, the Assyrians invented a system of wrapping the clay tablet with a contract in a second layer of clay and copying the contract again on the envelope [Chiera, 1960]. In case of disagreement on the authenticity of the text on the outer layer, this layer was broken and the inside could be inspected. As there was no way of changing the inside tablet without damaging the outer layer beyond repair, this offered an early but very effective way of authentication (figure 2.2).

The first positive reference to a collection of books for other than archival purposes, i.e. a library, is in a letter from an Assyrian King (possibly Assur-

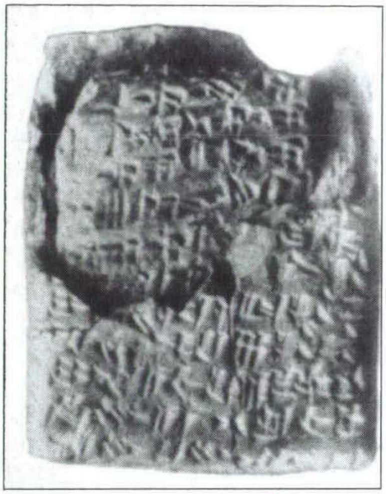


Figure 2.2: A document inside its 'authenticity' wrapper

banipal) to one of his agents, from approx. 700 BC.

> "Message from the King to Sjadoeno: I am well; I hope that you are well too. When you receive this letter, take the three men with you [names are given] and the learned men of the town of Borsippa, and collect all books that are in their houses, and all books that are kept in the temple Ezida... [follows a list of books that are considered to be of great value] ...search for valuable books that are in your archives and which do not exist in Assyria and send them to me. I wrote to the librarians and guardians ... and nobody will refuse to hand you the books. When you see a book that I did not write about, but of which you think it may be important for me, take it and send it to me." [Chiera, 1960].

From this letter we not only learn that this king was a collector of books, but also that such collections already existed and were attached to temples. The same is true for the Greeks, where both temples and private persons, such as Euripides (5th century B.C.), actively collected books. The first important institutional libraries appeared in Athens during the 4th century BC with the great schools of philosophy. The most famous collection was that of the Peripatetic school, founded by Aristotle and systematically organized by him with the express intention of facilitating scientific research.

Aristotle's library formed the basis, mainly by means of copies, of the library established at Alexandria, which grew to become the most famous library in antiquity. At its height it held 700,000 books. Most of them must have been copies, because there were not enough authors in the ancient world to produce so many titles [Sprague de Camp, 1961]. We know a little about the organization of this library, which was situated in the *Mouseion*, or 'temple of the Muses'. New books, for instance, were not introduced immediately in the library, but were first stored more or less according to provenance, waiting for subject specialists who saw to the arrangement according to content. But the most important tool that we know of in the Alexandrine library was a catalogue raisonnée created by Callimachus (305-240 BC.), the *Pinakes*, which means '(painted) tablets'. This name presumably refers to painted signs hanging over the shelves or presses in which the books were stored. We know that Callimachus used a classification system with six poetic genres and at least five prosaic areas (history, rhetoric, philosophy, medicine and law) and an inevitable section of 'miscellaneous'. In each category the books were arranged alphabetically according to author. Also for each book the *incipit* (the first line or lines of the book), the number of scrolls and the number of lines was given (for checking the accuracy of copiers). Chronological data on authors were provided and critical commentaries were supplied where needed, e.g. for works of doubtful authenticity [Witty, 1958]. This catalogue perhaps was not the first in its kind, because we know of a similar method used in the library of Assurbanipal in Niniveh, three centuries earlier [Thompson, 1968b].

The real or imagined importance of libraries as a factor in political power may be measured from the fact that the Ptolemies placed an embargo on papyrus partly to keep it from Pergamon, the library of the Attalids, the rivals of the Ptolemies. Although this embargo may have stimulated the use of parchment in the library of Pergamon, leather (abundant in this country of goats and sheep) had been used as a writing material for a long time. Nevertheless from this moment leather as substrate for writing became to be called 'parchment', after the name of that city.

2.1.2 The Romans and early christians

The Romans avidly copied much of Greek and Hellenistic civilization, and after the conquest of the East the private library became commonplace under the rich Romans. Conquering generals seized books and libraries and used them to set up libraries in Rome, and this may have contributed to the fact that such private collections in Rome were more important and used more heavily than public libraries. A number of private libraries was founded by the emperors, often as an adjunct to a temple. Another and to us rather unexpected place where to find book collections were the larger public baths, which to the Romans were not only hygienic institutions, but which also functioned as the 'club' to the English gentlemen or perhaps the green for golfers of the twentieth century: a place where political and business deals were prepared in a relaxed atmosphere.

We do not know much more about the organization of the Roman libraries than about that of their Greek predecessors. Nevertheless we may assume that general classifications of knowledge such as those of Aristotle and Porphyry, and scholarly curricula such as the *trivium* and *quadrivium*, expounded by Julius Caesar's librarian Terentius Varro, will have played a part [Boorstein, 1983].

The trivium and quadrivium together form the *artes liberales* (which in turn may be associated with the Greek *enkyklios paidea*: the 'all-round education', from which term our 16th-century encyclopedia is derived. This latter term was used in the neo-pythagorean school of Alexandria to indicate the inner cohesion of all sciences. Varro enumerates nine 'artes liberales' in his *Disciplinarum libri IX*: grammatica, rhetorica, geometria, arithmetica, astronomia, musica, medicina and architectura. The trivium contained grammatica, rhetorica and dialectica; the quadrivium the arithmetica, geometria, astronomia and the musica (the lists are not always consistent). This classification remained in use during the Roman times and the middle ages and was as a system only superseded by Dewey's Decimal Classification system in 1876 (see below, section 2.1.4).

From volume to codex

Between the time of the ancient libraries and that of the libraries of the middle ages an important invention changed the outlook of book collections: following the change from papyrus into parchment the scroll or *volumen* was replaced by the *codex*, the bound book. As a result, texts became more compact and easier to handle. Nevertheless the word 'volume' is still used in our modern libraries.

The papyrus scroll had many disadvantages. There exist Egyptian scrolls with a length of 150 feet, but the average volumen might measure some 40 feet. This sufficed for about one hundred pages, so that e.g. the Iliad or Odyssee required fifteen or twenty scrolls each (later standardized to 23 scrolls or 'books'). Compared to this, the *codex* was wonderfully convenient. Pages could be written on either side and, more importantly, every page could be accessed immediately, which was a great improvement over the tiresome scrolling and unscrolling of the volumen. This feature invited a host of reference and retrieval tools: a title page, a Table of contents, numbered pages and an index [Boorstein, 1983].

It must be realized that both judaeism and christianity (and later islam) are very much religions of 'the book'. The word 'bible' itself of course means 'book'[1] and especially in christianity emerged a tradition of writing: letters, service books, commentaries and the like. The oldest codices that have survived are from the first century AD and contain christian texts [Boorstein, 1983]. For a variety of reasons the early christians preferred the more durable *vellum* (parchment) above the brittle papyrus. In referring to their sacred writings they often made comparative studies of sources as the writings were related, and students liked to refer from one source to another.

[1] Actually the word *biblos* referred to the Phenician city that served as distribution centre of papyrus.

This reference work entailed having a comparatively large volume of writings available 'on the desktop' and increased the attractiveness of the easy turning of pages possible with a codex. In this respect it is noteworthy that Roman legal scholarship, which also required a comparison of sources, likewise showed an early preference for the codex. Finally it was the express intention of early christians to shun pagan literature by using an entirely different form of book[2].

In this light it is interesting that to this day the jews adhered to the scroll for their religious writings. Although the study of 'the Law' is all-important in the jewish religion, it was more important to emphasize the distance between the new religion of christianity and the old religion of Moses, and again the form of the book was used as a discriminator.

Popular consensus has it that the occupation of Rome by the germanic tribes of Vandals and Goths effectively put an end to all culture and science. The opposite is true; for instance, king Theodoric (480-493) was an able ruler who stimulated the growth of literacy. His 'minister of culture' was the famous Boethius, and the historian and grammarian Cassiodorus also fulfilled a high function in his administration [de Burgh, 1959]. The latter founded two monasteries in Calabria, which he called *Vivarium* from the nearby fish ponds. He made these monastic foundations a sort of academy, and his work *Institutiones divinarum et saecularum litterarum* tells us much about his ideas on the organization and use of the library. No mention is made of a classification system, but one passage suggests at least the existence of a shelf list, and the commentaries on the scriptures were marked with classification symbols for easy reference ([Thompson, 1968a]). Cassiodorus in turn influenced Saint Benedict (480-550 AD), the father of the European monastery. His rules again stress the importance of the reading and, incidentally, the copying of books.

2.1.3 Medieval libraries

The general procedure for early libraries untill well into the 19th century was to decide on some subject classification for the books and to allot rooms and shelves to those subjects, not unlike the organization suggested by Callimachus' Pinakes. Once placed, the book was identified by room, shelf and number, and this identification remained fixed until the next major reorganization of the library.

Most libraries boasted a catalogue, or at least an inventory, showing the bookcase and sometimes the shelf where each book was kept. Generally catalogues grouped the books in three divisions. First came the bible and commentaries. Writings of the church fathers and contemporary theologians followed. Finally there was a smaller section of worldly books, including at various places some classics such as Virgil and Homer, but also mathematics, medicine, astronomy, law, and historical and philosophical writings. In a few cases fairly systematic catalogues have survived; one of the most notable is that at Dover Priory, compiled in 1339, which consists of three parts [Irwin, 1968]:

[2] The CD-ROM edition of the Encyclopaedia Brittanica also offers an excellent introduction in the history of libraries and in library science in general.

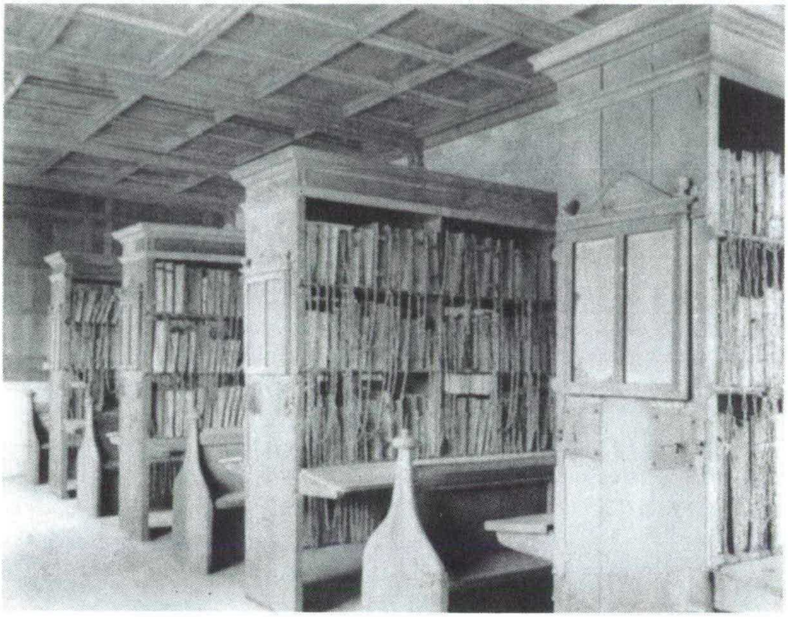


Figure 2.3: Books on chains in a medieval library.

- a list of titles with the volume, number of leaves or pages and the number of separate treatises in the book;
- a shelf list, each item in composite works being noted; with the leaf or page where it begins and the incipit
- an author list in alphabetical order of the entire collection.

Because of the labour invested in the production of the individual book, the volumes often were literally chained to the shelves. The chains were attached to the front side of the covers and this caused the books to be stored on the shelf with the spine out of sight to avoid the covers to be damaged by the chains. Attached to every bookcase was a desk, allowing the books to be read without disconnecting the chain. Such a library still is preserved in its original form at Hereford Cathedral (Figure 2.3). It was only with the invention of the printing press that books became cheap enough to be left unchained, and now the spine offered itself as a convenient place to display author and title.

2.1.4 Dewey's revolution

The organization used in antiquity and the middle ages for storing and retrieving books was thus very much centered on the shelf on which the book was placed and on the author of the individual book. The invention of the printing press and the rapid growth of literacy in Renaissance and Enlightenment made the organization of libraries and the retrieval of information ever more difficult. Nevertheless the organization of libraries remained essentially unchanged until the late 19th century, when finally the need was felt for an organization centered on the subject of the book, rather than on the author. Most of the impetus for this change came from science and technology, where the practice of working in teams in research institutions partly superseded the practice of single individuals, who worked for years to complete their research and then published the results as a book. Also, the proliferation of specialist journals that publish short papers charting the progress of teamwork has meant that the names of single authors have become somewhat less important as tools for identifying works in libraries. Catalogues that list the subjects of research are more useful to specialists in related fields around the world, who may not know researchers by name but wish to have access to their work.

In 1876 Dewey published anonymously a work titled: *A classification and subject index for cataloguing and arranging the books and pamphlets of a library* [Dewey, 1876] that was to have far-reaching effects. The real value of Dewey's work was not primarily the specific classification scheme he developed (although this is still widely used), but rather the idea of

- the introduction of *relative* as opposed to *absolute* location;
- the assignment of (decimal) numbers to books rather than shelves, thereby making the specification of detailed subjects feasible;
- the provision of a relative index on subjects.

The great difference between Dewey's system and previous systems was that the subject identity was attached to the book instead of to the shelf. Books were placed relative to each other using a linear numbering scheme, hierarchically divided into 10 classes by 10 divisions by 10 sections. At the time Dewey was criticized for having put too much detail into his scheme, but the 17th edition of his classification system, published in 1965, had 20,000 topics and was criticized for not being detailed enough.

2.1.5 Other library systems

Many European libraries today use a direct descendant of Dewey's classification schema, the Universal Decimal Classification by Otlet and Lafontaine [Foskett, 1982]. In 1894 they had sought and obtained Dewey's permission to use and extend his system to a "universal index to recorded knowledge" [Rayward, 1975]. The reason that caused them to turn to Dewey's numerical scheme was the simple fact that any alphabetic arrangement of concepts and keywords was out of the question in the multi-lingual and nationalist Europe of that time, and ironically the way to a numerical system was pointed out by an American. The main difference between Dewey's system and the UDC was the introduction of 'auxiliaries', special characters that enable the documentalist

	Increasing association			
Clarity of perception		Cognition (awareness)	Memory (temporary)	Evaluation (fixed memory)
	Recognition (concurrent)	Concurrence /#	Self-activity /*	Association /;
	Convergent thinking (not distinct)	Equivalence /=	Dimensional /+	Appurtenance /(
	Divergent thinking (distinct)	Distinctness /)	Reaction /-	Functional dependence /:

```
example: ''Books by English authors''
Authors /: Books
\#
Nationality /= English
```

Table 2.1: Farradane's operators

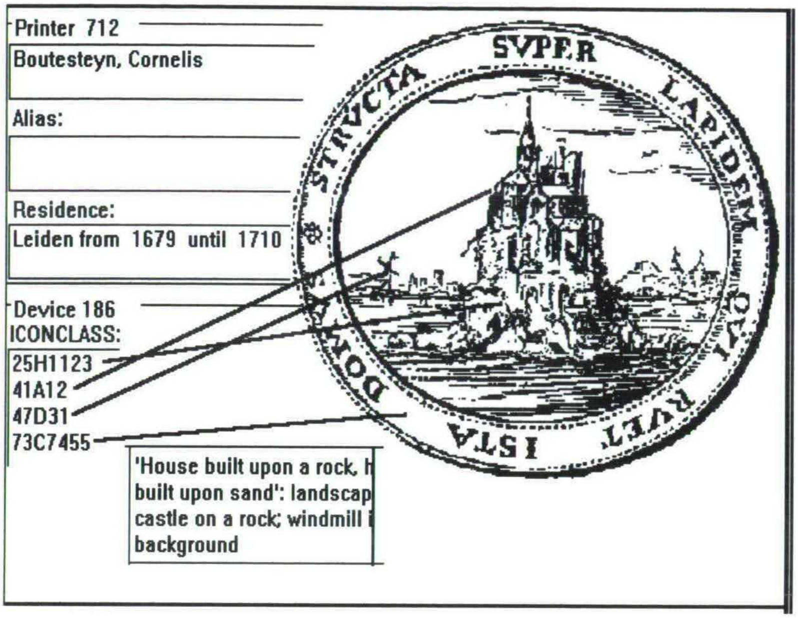
to synthesize the codes [Foskett, 1982]. For instance, if 635.965 would refer to *hot-air heating* and 697.38 to *indoor plants*, the signature 635.965:697.38 could refer to *the effect of hot-air heating on indoor plants.*

An interesting application of UDC-like signatures outside the library may be found in the ICONCLASS system of [van de Waal, 1955]. This system uses decimal numbers, 'auxiliaries' and even short strings (for proper names) to describe pictures. For an example of a picture described with ICONCLASS signatures see figure 2.4.

Other scholars devised different techniques for translating the content of documents into a more formalized language, among whom Ranganathan ([Ranganathan, 1967]). He introduced the term *facet analysis* to denote the technique of dividing a complex subject into its several parts by relating them to a set of five fundamental categories of abstract notions, which he called *personality*, *energy*, *matter*, *space*, and *time*. He employed these in his Colon Classification system (1933), which is used in some Indian libraries, but has found few followers elsewhere.

Yet another ingenious system was developed by Farradane. This system was intended to furnish a method to map the possible relations between keywords [Farradane, 1966]. Farradane based his system on the development of the learning process of children, which, he maintained, was based on developing powers of discrimination in time and space. In time, the stages of co-occurence of two concepts or ideas would be 'non-time' or 'awareness', 'temporary' and 'fixed'; in space 'concurrent', 'not-distinct' and 'distinct'. These six stages form a matrix, the points of intersection denoting nine different kinds of relations. Concepts may be joined according to these relations using special operators (see Table 2.1).

Controlled dictionaries and thesauri were developed to create some semantic order in the jungle of language, and for many libraries this still is the



Printer 712

Boutesteyn, Cornelis

Alias:

Residence:

Leiden from 1679 until 1710

Device 186

ICONCLASS:

25H1123
41A12
47D31
73C7455

'House built upon a rock, h
built upon sand': landscap
castle on a rock; windmill i
background

The picture shows a printer's device used by the 17th-century Dutch printer Boutesteyn ('sturdy stone'). The subject of the image is indexed with four ICONCLASS notations, from divisions 2, 4, and 7. The short description says: "House built upon a rock, house built upon sand": landscape with castle on rock; windmill in background.

Notation (Code)	Textual Correlate (Meaning)
25H1123	rock-formations
41A12	castle
47D31	windmill
73C7455	"house built upon a rock; house built upon sand" doctrine of Christ on love, etc. (Matthew 7:24-27; Luke 6:47-49)

Figure 2.4: Example of ICONCLASS signatures.

```
database machines
    USE   database management systems special
          purpose computers database
          management systems
    UF    data dictionaries
          database machines
          databases
          DBMS
    NT    distributed databases
          relational databases
    BT    file organization
          management information systems
    TT    computer applications
          file organization
    RT    database theory
          decision support systems
          integrated software

database theory
    RT    database management systems
          distributed databases
          programming theory
          relational databases
```

Table 2.2: Part of a thesaurus

prevalent state of affairs. A controlled dictionary is a list of words that are allowed in the index and is the opposite of the stop list (see also chapter 1). The word 'thesaurus' is sometimes used to refer to a controlled dictionary, but in general is a structured list of words in which the relationships between the words are mapped in relations like *broader terms*, *narrower terms*, *related terms* and the like (see Table 2.2)

A detailed survey of these systems and methods may be found in [Foskett, 1982].

2.2 The index language model

A general model of information retrieval thus slowly emerged that in reality dealt not so much with *information retrieval*, but with *document retrieval*, because when we use an IR system in a library, we do not retrieve information, but pointers to books, articles, *documents*. Hopefully the information that we need is somewhere in those documents, but often it is not. Therefore it is very important that IR systems have structures that enable the user to inspect the set of documents that were retrieved and to reformulate his query.

In order to allow a collection of documents to be searched, some information about the contents of the documents must have been collected in the system. This is the essence of IR: to condense in some manner the contents of an otherwise unmanagable quantity of documents and articles to a size that can be conveniently searched, but still contains sufficient information to select potentially useful documents. Such information may be as shallow as that contained in author and title, it may be an abstract or a set of keywords, or it may be a complex representation of the contents of and concepts in the document.

When the information is collected, that is hopefully relevant for the information need of a future user of the system, it has to be expressed, or translated, in a description or representation in the IR system in such a way that it is accessible to that part of the system that accepts the user's query. Therefore the repository of these representations is aptly called the *index language*, and the traditional problems that we face in information retrieval are, essentially, linguistic problems (see also [Sparck-Jones and Kay, 1973]):

- What parts of the document should be considered for analysis, and which semantic, syntactic or other features should we take into account?
- What should be the semantic 'units of description' in the index language and what syntactic features should they have?
- What (syntactic and semantic) devices are available in the index language for manipulating descriptions during a search?

Now let us again ascertain the position of the IR system proper as the link between the document and the prospective user (see Figure 1.9). On the left we see the documents, on the right the queries put forward by the user. Both have to be 'translated' into the index language. To answer a query, the translation of the query must be matched against the translations of the documents: the document representations. This matching takes place using the similarity functions that are provided by the index language (see also a more formal description of this process in chapter 4).

If we want to work with this model we should be alert to the various guises that the original document, or rather the data in the original document, may take. Including the document itself, there are at least four different forms, more or less corresponding to the three vertically aligned areas of Figure 4.2 in chapter 4):

- The document itself. However, things are never as simple as that. There are many stages in the process that take place between the moment that the document leaves the hands of its author and the moment that it starts its way through an IR system. In the meantime it may have been printed and published in a variety of forms or as part of one or more hyperdocuments. It may also have been submitted to processes such as OCR (optical character recognition), parsing, filtering or tagging long before it enters the IR system and the dividing line between the document and the next stage, the document surrogate, is never very clear.

- The *document surrogate* is the part of the document that is the input to the IR system. It may be the complete document, certain parts of it, such as the title, an abstract or the table of contents, or even a set of keywords prepared by a documentalist. In the case of scientific articles the abstract is often written by the author of the document and as such is part of it. The importance of the document surrogate will be understood by the fact that in the early IR-systems and experiments no explicit difference was made between the "full text" of the document and the surrogate. For a long time it was silently understood that the text that was indexed by "full-text" systems in reality was an abstract and only in the last fifteen years it has become routine to index the full text of a complete document.

- The *document representation* is constructed by the system from the document surrogate and it is stored as a representation of the original document in the index language. In most IR systems the document representation consists of a set of keywords.

- The *on-line document.* When a document is included in the result set of a query, a reference to it is presented to the user. This reference can be the bibliographic reference, but it can also include parts of the document representation, the document surrogate or even the complete document. That representation of the document that is presented to the user, and which may well differ substantially from the original document, the document surrogate or the document representation, is the on-line document.

A special case is the scanning of complete documents at query time by e.g. a regular expression search program (regular expressions are a kind of shorthand for describing strings). In this case the document, the document surrogate and the document representation are one and the same thing.

In the next two chapters we will consider various types of IR systems. In chapter 3 we will see how the advent of the computer substantially changed the way that information was retrieved; first by automating parts of the model presented here, but also by introducing new methods and models. In chapter 4 we will concentrate on models that describe the document representation and in chapter 4.3.3 the strategies that are used to compare these document representations with queries and with one another.

Chapter 3

Information retrieval and automation

3.1 The onslaught of mechanization

The usual way to specify subjects in non-automated libraries was and is by means of compound terms, generally noun phrases (or NPs). When single words like 'system' or 'government' don't suffice, modifiers are added until the necessary level of precision is reached, e.g. 'government information systems', 'systems analysis and programming' or 'electric welding of aluminum'. While such descriptors come naturally to humans, they have a number of drawbacks. For instance, the number of subjects may become very large as it is now possible to go in very fine detail, and this ever increasing size of the indices brings its own problems of storage and retrieval. At the same time it is no longer a simple matter to arrange multi-term items in a way that allows for easy access in manual systems; should 'electric welding of aluminum' be stored under the 'A' of aluminum, the 'E' of electric or the 'W' of welding? Therefore, in many indexing systems the individual words in compound terms are rotated and every permutation is made an entry in the alphabetical list, but this exacerbates the problem of index size.

In Table 3.1 some examples of manipulated precoordinative indices are shown, including the Keyword in Context (KWIC) arrangement. Because the concepts in the index were combined or coordinated prior to searching, this type of systems is called *pre-coordinative.*

In the period between 1930 and 1950 librarians sought a way out from under the ever increasing weight of such manipulated indices. The obvious solution is to only allow short terms, and leave it until the moment that a search is done to coordinate the terms. Such systems are called *post-coordinative systems.* The first of these systems was Taube's *Uniterm* system of 1952 [Foskett, 1982], in which for every keyword a list is kept of the documents to which it is assigned. If the user wants to perform a logical AND operation on two keywords, for example, he has to compare the lists for these two keywords and select the documents that occur in both lists. This is of course a tedious operation, and soon mechanical devices were invented to perform such operations automati-

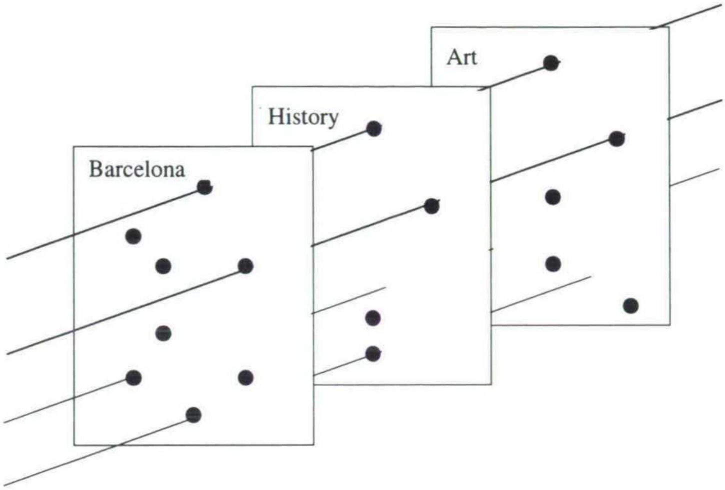


Figure 3.1: The Peek-a-boo or 'optical coincidence' system

cally, even in the pre-computer era. There exist a number of variations, notably the *optical coincidence* or *peek-a-boo* system (Figure 3.1) or the *notched-edge cards*, but the underlying principle in all cases is the application of Boolean logic, more in particular the Boolean operators AND, OR and NOT.

In a peek-a-boo system each card represents a keyword and all documents in the collection are represented by a position on a grid that is printed on the card. The documents that are relevant to that particular keyword are represented by a hole punched out on the position of that document. If, for example, somebody wants to know which documents are about Barcelona AND history AND art, he takes the three cards for these three keywords and aligns them; if there are any documents relevant for all three keywords, the positions of these documents will be clearly recognizable by the light passing through all three cards.

In the notched-edge system each card represents a document; the keywords are defined by holes along the edge. If a keyword is assigned to a document, the cardboard between the hole and the edge is cut away. Searching is performed by inserting pins through the holes that correspond with the keyword that is sought; when the pin is lifted, the cards of which the edge is cut through, will sag down (Figure 3.2). By inserting more than one pin, all kinds of Boolean operations can be performed with relative ease.

3.1.1 The computer and the library

The introduction of post-coordinative systems predated by perhaps a decade that other revolution that changed the world during the second half of this century: the birth of the computer. In the late fifties and early sixties of

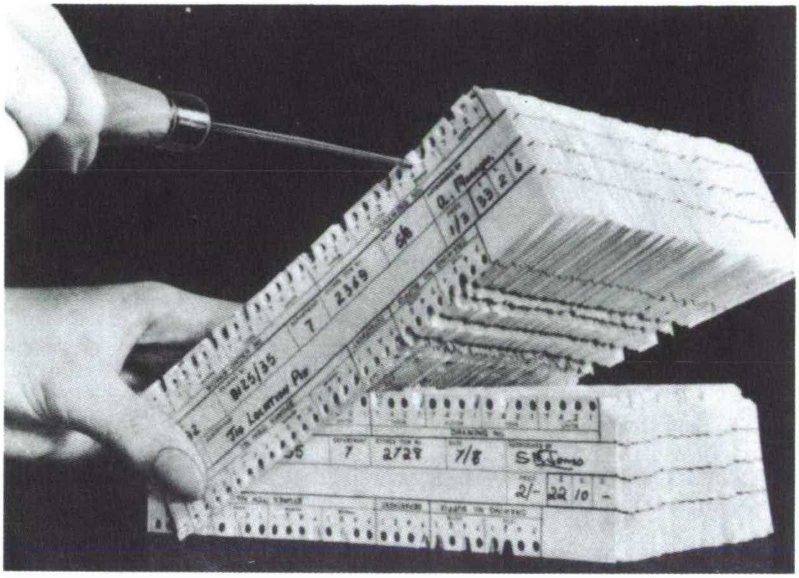


Figure 3.2: Notched edge cards and application

this century many researchers speculated on the possibilities of using computers for the storage and retrieval of books and articles, e.g. [Luhn, 1958] and [Edmundson, 1969]. Not unlike the high-strung expectations with regard to machine translation and artificial intelligence, some scholars and scientists thought that the complete automation of libraries was only a few years away.

These expectations were never realized. 'Digital libraries' are becoming a viable concept only in the last decade, and are not conceived as fully computerized institutions, but rather as hybrid forms of libraries in the traditional sense integrated with automated services and the handling of electronic documents (see [Mackenzie Owen, 1998]).

What could be achieved given the state of the art of that time was the application of computers to clerical work, such as the creation of permutations in pre-coordinative systems, generating KWIC and KWOC indices (see Table 3.1), the rearranging of existing indices, and other tasks aimed at the creation of paper catalogues. Such tasks were performed in batch-mode, as computers were relatively slow and on-line use was clumsy and expensive until in the late seventies. Nevertheless, progress was made in the creation of full indices of document surrogates, such as abstracts or lists of keywords. Incidentally, further experiments have led to the important realization that such inversion of documents was not really a good starting point for the retrieving of information. We will have more to say on this in section 4.3.

Searching a library collection is very much an interactive process. This

1 Unmanipulated catchword indexing

```
Library classification on the march
Library classification, Prolegomena to
Library education
Library in the British Museum, The King's
Library, Mechanized acquisition procedures in the University of
Library of Canada, The National
Library service in Uncoln, The hospital.
```

2 Manipulated catchword indexing

```
Library classification' Introduction to
Library classification, Manual of
Library classification, A modern outline of
Library classification on the march
Library classification, Prolegomena to Library education
Library, The King's, in the British Museum
Library, National, of Canada Library
```

3 KWIC indexing

```
University of Maryland library/ Mechanized acquisition procedures
                Public library administration/
       Introduction to library classification/
             Manual of library classification/
   A modern outline of library classification/
        Prolegomena to library classification/
                       Library classification on the march/
                       Library education/
            The King's Library in the British Museum
          The National Library of Canada/
```

4 KWOC indexing

```
Library  A modern outline of library classification
Library  Introduction to library classification
Library  Library classification on the march
Library  Library education
```

Table 3.1: Various manipulations of precoordinate indices

is caused by the fact that it is often difficult to formulate a query in such a way that the result is fully satisfying the first time around. It is therefore not surprising that computerized information retrieval only came into its stride when on-line terminals and timesharing operating systems made interactive use of computers possible.

The post-coordinative approach looks as if it is made for computers. Its central operations are Table lookup and the application of set operations on the results; typically tasks where computers outperform humans. An added bonus is that the original texts, or at least abstracts or titles, can be displayed immediately, and even now, shortly before the year 2000, most automated library systems use such fifty-year old techniques: they first look up keywords in an index, they then perform basic Boolean set operations, and finally they display, as the result of the query, a list of titles and, perhaps, abstracts.

3.2 Indexing by computer

There are essentially two ways of assigning keywords to documents:

- selecting the keywords from an existing classification system, thus assigning the document a place in this classification system (assigned indexing);

- identifying all words occurring in a document and selecting the keywords by some weighting method (derived indexing, also called *indexing by extraction* [Lancaster, 1976].

Assigned indexing is generally applied when human effort is used to index documents. Derived indexing systems only came into being with the development of the computer.

As already noted, computers can not only manipulate existing indices, but also create indices from machine-readable text without human intervention. They can identify the word tokens in a text and store the occurrences in an inverted file, with the corresponding word token or a derivation of it as entry point (key). Together with fast lookup and the application of Boolean set operations this held the promise of fully automated retrieval systems, but as we will see below, this was not to be.

Therefore the observations of Cleverdon on the subject of human indexing [Cleverdon, 1984] and the famous Blair/Maron experiment in full text retrieval [Blair and Maron, 1985] still loom darkly over all attempts to substantially improve the effectivity of information retrieval techniques. The probabilistic and vector space approaches (see chapter 4), that have been perfected in the last twenty years, cannot claim to imply real understanding of documents, however sophisticated they may be. On the other hand, the AI- or linguistic based approaches are hampered by the fact that the creation and maintaining of involved knowledge representations only works for very small domains (as e.g. in the systems SCISOR

[Rau et al., 1989, Rau and Jacobs, 1990, Rau and Jacobs, 1988] and the German TOPIC[1] [Hahn and Reimer, 1987, Hahn, 1990]). And, as Cleverdon observed, human indexing just is not consistent enough to guarantee acceptable recall and precision over sizable databases.

The obvious reason for this rather pessimistic outlook lies in the fact that the ultimate vehicle for the transmission of knowledge and information between humans is natural language. In whatever form we may conceive our ideas and information needs and even if 'a picture is worth a thousand words', in our civilization the written word, i.e. the document, will remain the principal vehicle for ideas and information for a long time, whether the document is printed on paper or stored electronically. Indeed it might be said that the document is the memory of the species. And natural language texts are extremely difficult for computers to analyse semantically.

The history of research in IR after the introduction of the computer may be described as a movement of a pendulum. In the sixties and seventies much work was done on the quantitative aspects of text and the models that were proposed were based on word frequencies and word occurrences. A second phase took place in the late seventies and in the eighties, in the wake of research in artificial intelligence by Schank and others, that held out hopes of in-depth analyses of natural language texts. Efforts were made to 'understand' the contents of documents by combinations of top-down and bottom-up parsing and by creating structured, conceptual document representations. However, it was soon found that such representations do not scale well. In the early nineties such efforts were largely abandoned and mainstream research returned to the frequency-based, word-oriented models, now aided by the corpora of machine redable text that had been collected. A typical test collection for IR or text classification now is in the magnitude of several dozens of megabytes and contains tens of thousands of records: the TREC collection even contains three gigabytes. Not unexpectedly, such resources have given a big impetus to renewed statistical analysis of texts. It was found that methods from machine learning could be applied to the classification and categorization of texts; in all cases this meant a return to the document vector model.

3.3 Measuring performance

It is far more difficult to measure the performance of an IR system than that of a non-documentary database management system. As we have seen already in chapter 1, the translation of the properties of a non-documentary object into the attributes and tables of, e.g., a relational database is not hampered by the same difficulties of vagueness and ambiguity as the translation of the content of a document into the document representation. Thus, after normalization of a relational database management system, its performance is mainly a matter of its efficiency in file- and table access. Of course, similar aspects play a role in IR systems too, but as [Blair, 1996] remarked, advancements in these

[1] Not to be confused with the TOPIC system of Verity Inc.

technical areas have often obscured the fact that the critical problems in IR are very different. A satisfactory performance is only in part brought about by the efficiency of hard- and software in reading and manipulating the datafiles. Far more important is the question whether the documents that are retrieved are of any use for the searcher. And even a perfect matching of keywords in the query and documents does not necessarily mean that the retrieved document is at all relevant for his information need, or even has the same 'topicality' as his query.

[Cleverdon and Keen, 1966] listed the factors that are to be taken into consideration for the appraisal of an IR system:

1. the *coverage* of the collection, that is, the extent to which the IR system includes relevant matter;
2. the *time* lag: the average interval between the time a request is made and the time an answer is given;
3. the form or *presentation* of outputs;
4. the *effort* involved on the part of the user to obtain answers to his search requests;
5. the *recall* of the system: the proportion of relevant material that is actually retrieved in response to a search request;
6. the *precision* of the system: the proportion of retrieved material that is actually relevant.

The first two items of this list lie outside the scope of IR models: the model, or rather the combination of models underlying an IR system, has no relation to the coverage of a collection of documents, and the time lag between request and answer is never the critical phase in automated IR-processing - the real bottleneck is the effort needed for indexing the documents. Moreover, the time lag of a system depends heavily on factors outside IR-models, such as the hardware that is used, and software implementation decisions.

The next two items are somewhat more dependent on the underlying models. For instance, the presentation of search results may be influenced by whether relevance feedback plays a role in the system, or the query language may have Boolean operators, because the system is based on the Boolean model.

The real value of an IR system for its users is best described in terms of the ratio between the relevant and the irrelevant documents in response to a query. Therefore we will in this chapter concentrate on the last two items.

3.3.1 Precision and recall

Assume a universe of documents $\{A, B, C, D\}$. Let $A \cup B$ be the documents that are retrieved by a certain query; $C \cup D$ the documents that are not retrieved. Let A be the documents in $A \cup B$ that are relevant to the query; B the set of documents that are irrelevant. Let $C \cup D$, the set of documents that are not retrieved, contain the set C of documents that are relevant to the query and the set D the set of documents that are irrelevant.

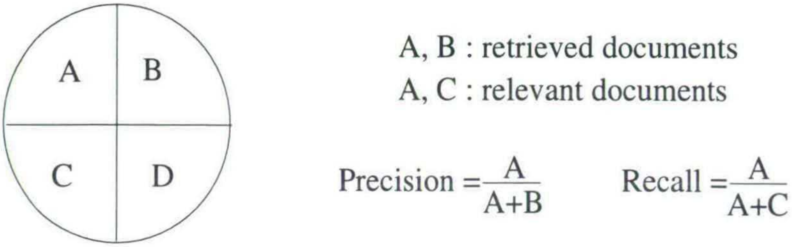


Figure 3.3: Precision and recall

Now the *precision ratio* is $A/A+B$ and the *Recall ratio* is $A/A+C$. In the ideal case of course B and C are empty, signifying that all relevant documents have been retrieved and no irrelevant ones (see Figure 3.3), but owing to the imprecise character of the concept 'relevance' this rarely is the case.

The obvious problem in computing these measures is that for a good estimate of the recall, the number of relevant documents in the database is needed, including the relevant documents that have *not* been retrieved. Of course there are statistical techniques to estimate this number (see also [Blair, 1996]). However, when the model allows for relevance ranking, precision and recall are not computed over the entire database, but over the first n documents that have been retrieved. To do this, they are first judged on relevance and then precision and recall are computed for the individual ranks in the list, or for some selected levels. To facilitate comparison between experiments, the precision values are often only given for the recall ratios of 0.25, 0.50 and 0.75.

Consider the example of Table 3.2. Fourteen documents have been retrieved and the relevance judgements are added (these are displayed in the third column). The total number of relevant documents in this set is found to be five. Now the precision and recall can be computed after every next document (see the last two columns). For example, after the third document, the number of 'seen' relevant documents is two, on a total of five relevant documents in the retrieved set, giving a recall of 0.4. In the same way, the number of relevant documents divided by the documents seen so far, gives a precision of 0.67.

These values can be entered in a graph (Figure 3.4). When the values are connected by lines, a saw-toothed curve emerges. This curve is not monotonic: a single precision ratio of e.g., 0.7, would return two corresponding recall values. To avoid this phenomenom the saw-toothed graph is converted to a blocked graph (the dotted line). In [Gordon and Kochen, 1989] an attempt is made to find a mathematical description of the relation between precision and recall. However, the function that they propose is monotonic and therefore cannot count as a correct description of the actual relation between the two measures.

n	Doc#	Rel.	Recall	Precision
1	588	y	0.2	1.00
2	589	y	0.4	1.00
3	576		0.4	0.67
4	590	y	0.6	0.75
5	986		0.6	0.60
6	592	y	0.8	0.67
7	984		0.8	0.57
8	988		0.8	0.50
9	578		0.8	0.44
10	985		0.8	0.40
11	103		0.8	0.36
12	591		0.8	0.33
13	772	y	1.0	0.38
14	990		1.0	0.36

Table 3.2: Recall-precision after judging n documents in linear order (Salton/McGill:1983, p. 166)

Micro- and macro evaluation

When the results of several searches have to be combined in single measures for recall and precision, the averages may be computed in two ways, called the *micro-* and the *macro evaluation* (see Fuhr [Fuhr, 1995] for a discussion of these concepts).

In the macro evaluation the individual values for precision or recall are computed first and then averaged. This can cause problems when one or more searches (or classification experiments) in a series yield no positive results. It has as advantage that every individual search attempt has the same weight, i.e. that the result is not biased towards the large classes or to queries that have many relevant documents in the database. The macro evaluation for precision is computed as follows:

$$p_{macro} = \frac{1}{N}\sum_{i=1}^{N} \frac{\parallel REL_i \cap RET_i \parallel}{\parallel RET_i \parallel}$$

where N is the number of queries, REL is the number of relevant documents found, and RET the number of retrieved documents.

The other way in which to compute an overall measure for the precision of an IR system is to first add the number of retrieved relevant documents, and the number of retrieved documents and divide the two afterwards. This circumvents the problem of empty sets and causes every individual document to have an equal influence on the result. The micro evaluation for precision is computed as:

$$p_{micro} = \frac{\sum_{i=1}^{N} \parallel REL_i \cap RET_i \parallel}{\sum_{i=1}^{N} \parallel RET_i \parallel}$$

The micro- and macro evaluation for recall are computed in a similar way.

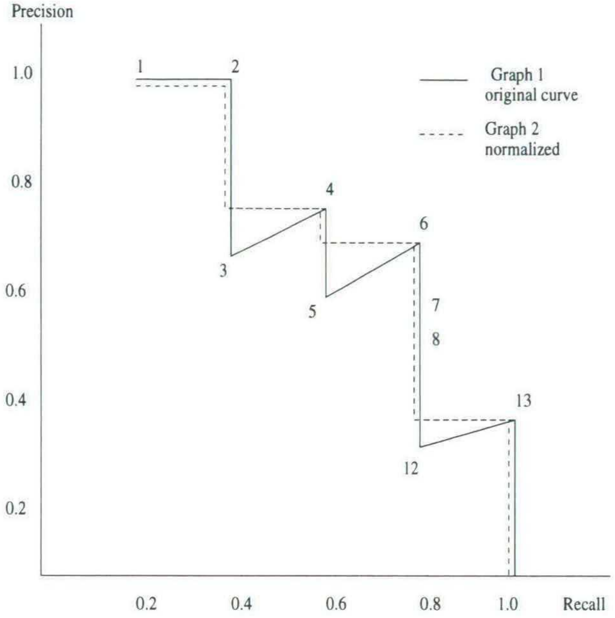


Figure 3.4: PR-graph for Table 3.2

The literature on classification systems prefers the micro evaluation but one must be aware of the fact that a few large classes can bias the outcome considerably. Therefore the differences between the two types of evaluation may be considerable, as we will see in chapter 7.

The breakeven point

Precision and recall are not ideal measures. They clearly vary inversely to each other, but there is no simple relation between the two. When a single measure is needed for the performance of retrieval or classification experiments, sometimes the *breakeven point* is used: the point where precision and recall have the same value. This breakeven point may be calculated from precision and recall scores, where the precision is computed for a number of different recall levels (or vice versa). The breakeven point is generally arrived at by linear intrapolation. Thus if the two points on a precision-recall curve that bracket this point are $< fp, fr >$ and $< sp, sr >$, the breakeven point is $< b, b >$, where

$$b = \frac{sr * fp - fr * sp}{sr - fr + fp - sp}$$

If $fp = fr$ or $sp = sr$, then the breakeven point is on the curve, not just bracketed by two points.

This concept is applicable only if the algorithm under consideration has some parameter that governs the trade-off between precision and recall, so that at least two different precision/recall values can be computed for every experiment. We encounter this limitation in chapter 7, when we use the C4.5 classification program. This program returns only a single decision for every classification experiment, and there is no way to influence its decision to include or exclude documents.

The breakeven point is not beyond controversy. In a personal communication[2], Lewis, who first published this measure [Lewis, 1991] stated serious shortcomings:

1. Interpolation gives values that are not necessarily achievable by the system. Although plotting of recall against precision usually gives a smooth, monotonically decreasing curve, more precise plots for single classes display a less smooth and not even monotonic curve.

2. Recall=precision is not a desirable or informative target. A system tuned for an optimal breakeven point is in a rather extreme state, where precision and recall are at their minimum and this does not necessarily reflect the preferences of the user.

3. The breakeven point acts as an average over diverse categories and is therefore of dubious value.

The breakeven point was used in [Apté et al., 1994a] and as we wanted to compare the results of our own 'local dictionaries' (see chapter 7) with those of the authors mentioned, we had to compute breakeven points and were confronted with the shortcomings of it. Apart from the work of [Apté et al., 1994a], [Apté et al., 1994b] we only know of its application in the Rainbow suite of programs of [McCallum, 1996]. We can conclude that it has never really entered the mainstream of IR evaluation.

Be this as it may be, there is an intuitive relation between a breakeven point and the performance of a classification or retrieval system and where it may not be much better than other measures, it certainly is not worse for comparison purposes.

The Harmonic Mean

Sometimes the two outcomes of precision P and recall R are combined in one single figure by taking the harmonic mean F of the two:

$$F = \frac{2}{\frac{1}{P} + \frac{1}{R}}$$

The magnitude of F varies from 0, when no relevant documents are retrieved, to 1, when all and only the relevant documents are retrieved. Moreover

[2] Also in the mailing list dlbeta@research.att.com of 11 Sept. 1997

F is strongly weighted towards the lower of the two values P and R; therefore this measure can only be high when both P and R are high (see also Figure 3.5). [van Rijsbergen, 1979] describes many other derivates from precision and recall in chapter seven. In Figure 3.5, the relation between the harmonic mean and precision / recall is clearly visible.

3.3.2 Specificity and exhaustivity

For completeness we mention two other measures that are used to describe the performance of IR systems. We have seen in the previous chapter how several approaches have been developed to create document representations that exist of keywords, and how the 'importance' of each keyword for a particular document or for the database as a whole can be estimated in the document vector model according to a veritable arsenal of techniques.

The results of the indexing procedure then may be measured in two dimensions:

- Specificity: the extent to which the system was able to recognize keywords that describe the topics more or less precisely, i.e. the depth of the analysis.
- Exhaustivity: the extent to which the system was able to recognize all the topics that are deemed relevant, i.e. the breadth of the analysis.

In the literature these two concepts are recognized as important parameters of IR systems, and several researchers [Salton and Yang, 1973], [Jones, 1973] have attempted to quantify these concepts and to relate them to term statistics. For example, exhaustivity can be related to the number of keywords that finds its way in a document representation, and specificity to the number of document representations in which a keyword occurs. As [van Rijsbergen, 1979] states: "I am arguing that in using distributional information about index terms to provide, say, index term weighting we are really attacking the old problem of controlling exhaustivity and specificity." and he adds that the trade-off between specificity and exhaustivity reflects the trade-off between precision and recall.

3.4 Computing a baseline

The baseline, or low performance standard of an IR system, is its performance when the selection of retrieved documents happened by pure chance. In that case the result of a query is equivalent to the blind selection without replacement of balls from an urn that contains white and black balls. The number of white and black balls equals the number of documents in the database; the white balls signify documents that are relevant to a query, the black balls those that are not relevant.

For a collection with N documents and R relevant documents, the probability of retrieving exactly r relevant documents by selecting at random n documents from the database is given by the hypergeometric distribution:

$$Pr(N, R, n, r) = \frac{\binom{R}{r}\binom{N-R}{n-r}}{\binom{N}{n}}$$

where the bracketed expressions on the right represent the binomial coefficients. So given a collection of 3000 documents and a query for which 10 documents in that collection would be relevant, the probability of having exactly one relevant document in a draw of four would be 0.01. If one wanted a probability of 0.5 to have at least one relevant document in the draw, 201 documents should be retrieved.

For every query and its corresponding number of relevant documents in the collection, we can compute the number of documents that should be retrieved to have a probability p of retrieving exactly r relevant documents in that draw. This gives us a precision and a recall ratio for that probability: in the two examples we would compute expected precision ratios of $\frac{1}{4} = 0.25$ and $\frac{1}{201} = 0.005$ and a recall of $\frac{1}{10} = 0.1$ for both. With one of the formulas described in the sections above, such as the harmonic mean, such double measures can be reduced to a single value that gives an indication of the effectiveness of the retrieval at that point. The harmonic mean would result for these examples in an effectiveness of 0.14 and 0.004 respectively. See Figure 3.5 for the relation between recall, precision and effectiveness. The horizontal axis shows the number of relevant documents in the experiment, the vertical axis the precision, the recall and the effectiveness. It is clear that the harmonic mean lies between the precision and the recall, but always nearer the lowest of the two.

But there is more to this figure than just the relation between these three measures. We can fix the probability at a certain level that marks the difference between outcomes that can be attributed to chance and those that cannot. [Shaw et al., 1997b] have suggested to take a significance level of $\alpha = 0.01$, because the risk of assigning an unlikely outcome to chance is more acceptable than that of assigning a likely event to non-chance factors. The baseline for a query in a database of N documents with R relevant documents can now be found in the following way. For a value n of documents drawn, the highest value of $r <= R$, denoted r_m, for which the probability of retrieving r_m or more relevant documents crosses the 0.01 threshold, defines the highest level of performance for a *random* retrieval process. The effectivenes of this point can be computed as shown above. When we repeat this procedure for all values of n ranging from 1 to N, we can be certain that all possible combinations of $n <= N$ and $r <= R$ have been considered and the n for which the effectiveness is maximized, while $p = 0.01$ is the retrieval baseline for queries that have R

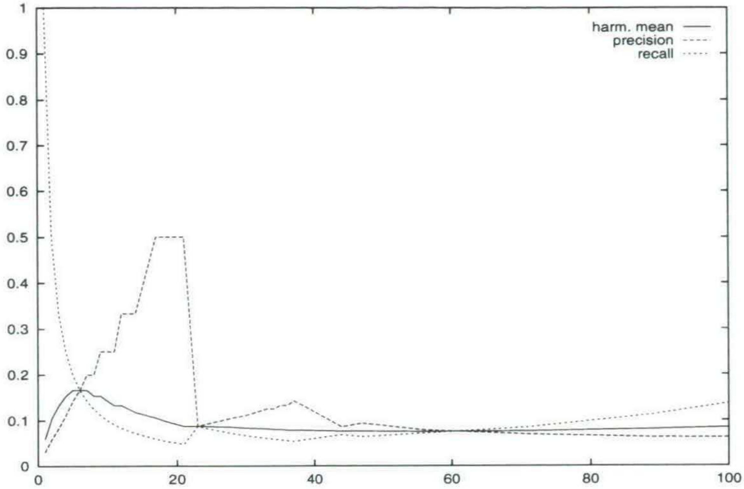


Figure 3.5: Baseline effectiveness as function of precision and recall

relevant documents in a database of N documents. In Figure 3.5 this baseline is drawn for the precision, the recall and the harmonic mean for 1-100 relevant documents in a database of 3000 documents.

As [Shaw et al., 1997b] state: "The low performance standard is based on identifying the highest level of retrieval effectiveness an *exceedingly patient* searcher can produce by such an random process" (emphasis added by the authors).

The graph labeled "3000 docs" in Figure 3.6 displays the low performance standard for the classes with less than 200 documents from a test collection with 3299 documents from the Reuters database (see chapter 7 for details). Again, the horizontal axis shows the number of relevant documents for each experiment and the vertical axis the effectiveness. We observe a peak of 0.16 when five to seven documents would be relevant for the query and a subsequent decrease to an effectiveness of 0.075 for 56 relevant documents. The performance then rises to an effectiveness of 0.36 for 719 relevant documents and 0.50 for 1087 documents (outside the graph).

The other graphs in the figure demonstrate the correlation between the number of documents in the database and the low level baseline: when the total number of documents in the database increases, the effectiveness decreases for each number of R (the total of relevant documents) and vice versa. The local high at the low end of the horizontal axis and the lowest point that follows it in the U-shaped part of the graph, move to the right with increasing numbers of documents.

Shaw, Burgin and Howell have compared the results of various experiments in clustering [Shaw et al., 1997a] and other IR-models [Shaw et al., 1997b] with the low performance baseline described above, with alarming results. They

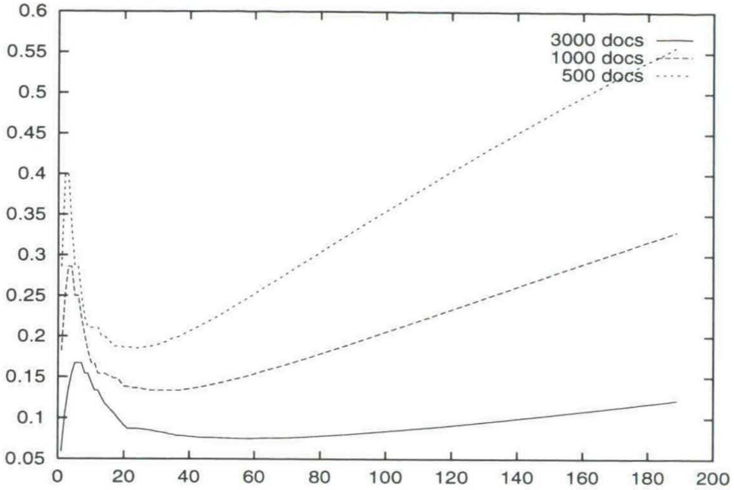


Figure 3.6: Hypergeometric distribution for 0-200 relevant documents on databases of different sizes.

found that the results that were reported using the cluster model did not significantly differ from this baseline. The results of systems that were based on the vector space model and other comparable models did display an effectiveness that was greater than the baseline, showing an improvement of on average 22% for experiments on 'traditional' databases, with $N < 30{,}000$ documents and 34% for the TREC databases, which are an order of magnitude bigger. This does not mean that the techniques used for retrieval in the TREC experiments were superior to the other systems. Rather, the baseline for databases of this size was so low that even a few retrieved relevant documents would strongly influence the effectiveness; Shaw, Burgin and Howell report an uniformly mediocre operational level of performance for the strategies used for experiments with the TREC collections.

3.5 Intermezzo: artificial intelligence

In this section we will look at the combination of artificial intelligence and information retrieval, and consider some examples of AI-based systems. These examples will cover a rule-based IR-system (TOPIC/RUBRIC) and two systems that depend on parsing and frames (SCISOR and the German TOPIC). We will also look at an example of a neural network in IR. Approaches based on genetic algorithms (GA) may be argued to also belong to artificial intelligence but because of their vector oriented operation we will treat these separately in section 4.3.3.

3.5.1 Frames and scripts

The notion of a *frame* is nothing but a reflection of the fact that almost any concept may be analysed into smaller concepts. A concept is represented by a frame and the sub-concepts may themselves be frames, that fit in 'slots' in the bigger frame. Concretely, a frame is a memory structure with a number of fixed slots. A car may for instance be the frame and the motor of the car fills the slot 'motor'. The motor itself may again be represented by a frame and then the slots may be the cylinders, the spark plugs and the carburetor, each to be filled by a value or a new frame. In short: frames can be seen as collections of semantic nodes and slots that together describe a stereotyped object or event.

Frames are often used in the context of *scripts*, which may be considered as frames with a temporal sequence. The keyword again is 'stereotype': a script describes expectations in certain stereotyped situations. If, in a restaurant-script, the waiter puts something on the table, the knowledge in the script expresses the expectation that the 'something' will be plates, food or the bill; not a boulder or the tyre of a car. It must be stressed here that the sheer size of knowledge to be analysed and described in advance prohibits the use of frames and scripts outside dedicated applications in small domains. More about frames and scripts can be found in e.g. [Schank and Abelson, 1977], Minsky [Minsky, 1981] or in the Handbook of Artificial Intelligence [Barr and Feigenbaum, 1989].

3.5.2 Bottom-up and top-down modelling

Frames and scripts are especially useful when a system wants to extract relevant information using top-down analysis of the events or situations described in a document.

Early attempts at intelligent IR, such as FRUMPS [DeJong, 1982] started as it were 'from the top', hence they were called *top-down* systems. This 'top' consisted of a collection of known, stereotyped situations such as earthquakes or railway accidents. The systems then tried to match incoming documents with such stereotypes, and if a match could be made, to fill in slots in frames, such as the number of casualties, the place and time of the quake, or the strength on the Richter scale. As these systems work with 'expectations' about what concepts will occur together in a text or text passage, they also are known as *expectation-driven* systems. Later systems, such as RESEARCHER [Lebowitz, 1986] or the German TOPIC ([Hahn, 1990] were more sophisticated in that they actively tried to build new representations of objects, instead of acting passively on a number of pre-cooked stereotypes. However, to do so they had to incorporate knowledge that was distilled from the texts themselves; this is called *bottom-up* processing.

Bottom-up modelling is more difficult, as it starts with the text itself and the individual words. A process called 'parsing' then tries to identify the parts of the sentences and their relation to each other. Obvious problems here are disambiguation and the resolution of anaphors and deixis. On the other hand, it can produce accurate results in arbitrary texts and texts that contain unexpected information.

In the following pages we will describe two systems that made use of these AI-techniques: SCISOR and TOPIC. It should be stressed that these systems are not IR systems. SCISOR is an example of a *question answering* system, whereas TOPIC is presented as a text condensation system. We include a description of both systems to offer an insight into the problems that emerge when a computer program tries to 'understand' a document and build a document *knowledge* representation, rather than a document representation that just contains keywords, as is customary in IR.

SCISOR

Developed at General Electrics, SCISOR (System for Conceptual Information Summarization, Organization and Retrieval) is an experimental system, that detects and stores information about financial transactions, such as mergers, takeovers etc. in an input stream of financial news (the Wall Street Journal). It subsequently answers simple questions about this domain, for instance "What was offered for Polaroid", or even incomplete questions as "Acquisitions by Shamrock" [Rau and Jacobs, 1990]. The system contains the following main functions:

1. selecting the stories that fit the domain.
2. creation of a conceptual representation.
3. storage and retrieval of the representation.

Selecting the stories that fit the domain The system is connected to an input stream of financial news stories. Leaving aside for the moment the rather trivial processing needed to recreate the story structures, such as headers, bylines and datalines, the first task of the system is to analyze the input stream to decide whether an incoming story is about corporate mergers and take-overs. It passes the story through a number of sieves, each trying to decide whether the story is definitely about the merger/take-over domain, or definitely not about this domain, or if there still is doubt left. In the latter case it is passed to the next sieve.

The sieves start with rather coarse filtering on headlines and keywords, becoming more sophisticated and thus computationally more expensive later on. This arrangement ensures that the expensive techniques have to be called in only on a subset of the documents. The modular architecture also makes it easy to plug new algorithms in or out, making comparisons between them relatively easy.

The performance of this system of sieves in terms of precision and recall may be estimated from an example given in [Rau and Jacobs, 1990], where respectively 88.5% recall and 92% precision are obtained. This may seem

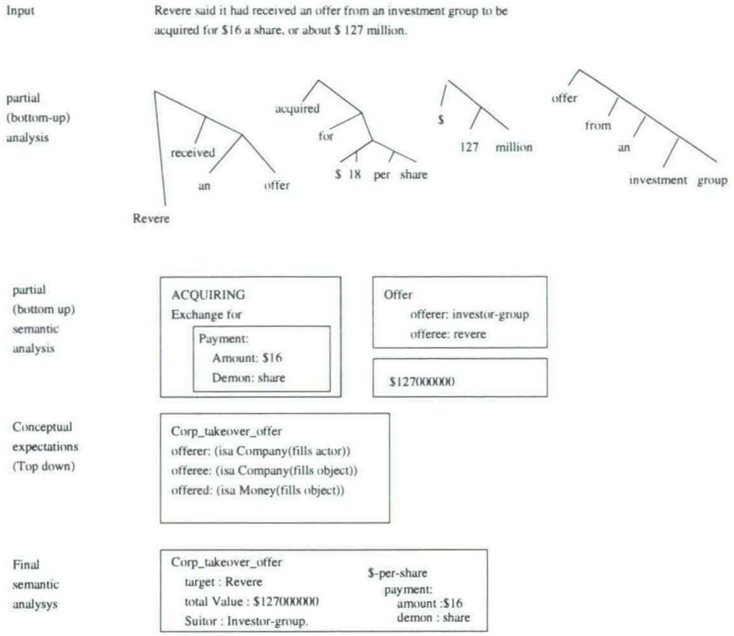


Figure 3.7: Bottom-up and top-down in the SCISOR system

rather high, but the measure for correctness was the estimate of a single person, and a 10% error margin was assumed. We will come back to this estimate in chapter 9.

Creation of a conceptual representation The next step in the processing is the application of natural language analysis to the stories thus selected. This analysis consists of an integration of both bottom-up linguistic parsing and top-down conceptual analysis. The bottom-up parsing module, TRUMP, identifies linguistic structures and tries to map these to a conceptual framework; the top-down analysis module TRUMPET tries to fit partial information from the text in conceptual expectations; see Figure 3.7. The input sentence here is "Revere said it had received an offer of an investment group to be acquired for $16 a share, or about $127 million". As the words are already in a lexicon, TRUMP understands all words, but it cannot complete its conclusions by bottom-up parsing alone. For example, the phrase starting with "to be acquired", might be attached to "an investment group" or "an offer", but in this example "Revere" is the subject of the phrase. The knowledge in TRUMPET is such that the *offerer* must be the same as the *acquirer* and that the *acquirer* must be different from the *acquiree*. This implies that "Revere" is the acquiree and therefore the *target* of the take-over.

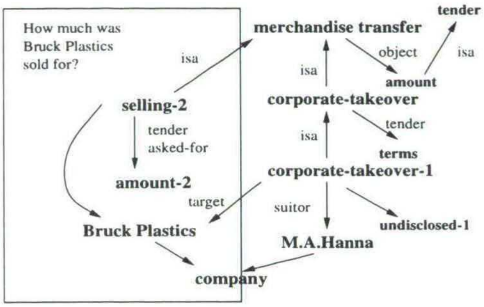


Figure 3.8: A question in the SCISOR system

Storage and retrieval of the representation The conceptual representation of the story that is created in this way is stored as a network of unique instances, i.e. as individual members of conceptual categories in a knowledge base. These instances serve as indices for information retrieval.

Answering of questions takes the form of reporting on slots. Consider the question "How much was Bruck Plastics sold for?" (figure 3.8). The question is also passed through TRUMP and TRUMPET to obtain a representation that is compatible with the knowledge base. The processing takes place in two stages. First a rough comparison is made with the features of stored representations. The second stage consists of a precise match of relationships that are asked for or implicit in the question. In the figure, the category *selling-2* causes the system to pass through *merchandise-transfer* and *corporate-takeover* and to find a match for the *target* "Bruck Plastics" with the *suitor* "M.A.Hanna". It then finds that the slot *terms* is filled with the slot-filler "*undisclosed*". This knowledge in its turn is passed to the module KING (Knowledge INtensive Generator), not shown in the figure, that generates English responses.

Nothing new about SCISOR has been published after [Rau and Jacobs, 1990] and the authors have since that time been involved in other occupations. It must be assumed that the system did not survive the paradigm shift from rule- and frame-based AI to statistical and quantitative methods.

The German TOPIC

Although the German TOPIC has the same name as the commercial descendant of RUBRIC, that is mentioned later in this chapter, it is an entirely different system. It belongs to those systems that create a structured knowledge representation, as do SCISOR and RESEARCHER. However, the difference with these systems is that whereas SCISOR and RESEARCHER are centered on the knowledge representation of a certain domain, and try to fill this representation with knowledge that is taken from an input stream of documents, TOPIC describes the individual documents and is therefore closer to the classical definition of

IR.

TOPIC is presented as a *text condensation* system. Text parsing augments an initially given frame knowledge base that describes the domain of discourse by adding text-specific knowledge. The extraction of knowledge is driven by script-like structures and controlled by so called *word experts*. Word experts are lexicalized grammatical modules, which do the actual job of mapping text items onto the knowledge representation structures. These word experts also resolve anaphors and other referential expressions.

The frames are connected by semantic relations. The central feature that defines the use of TOPIC is that the system maintains counts of the references to these structures. Every time that a reference is found to a frame, slot or slot filler in the knowledge representation of a text that is being parsed by TOPIC, a corresponding counter, the so-called *activation weight*, is incremented, and by inheritance this also happens to the counters of the higher structures. In this way a basis is created for judging the importance of the individual structures. For example, in Figure 3.9 we see how the top frame, ZENON-X, has an activation weight of six by virtue of the fact that four of its slots have been referred to once and one (*programm.lang.*) twice.

The combination of this hierarchical knowledge structure and the activation weights assigned to the various structures and substructures is exploited as a powerful tool for text summarization and the determination of dominant concepts in the text.

Identification of dominant frames A major measure for identifying an important concept in a text is the frequency of its explicit *and implicit* mention in the text. The activation weights that are attached to the structures in the TOPIC text representation are rather independent of linguistic surface phenomena, since in TOPIC these weights are not only adjusted by the explicit occurrence of the concept in the text, but also by implicit references. Hence an activation weight will be incremented by the occurrence of the same word or synonym, but also by the resolution of anaphors or by the recognition of a lower level concept, which as we have seen causes the higher concept to be updated by inheritance.

The actual text condensation is done in two steps. In the first one, the dominant concepts are identified; in the second step these concepts are recombined to form the *topic description* of a thematically coherent part of the text.

The dominance of a concept is judged according to the following considerations:

1. The height of the activation weight that was assigned to it. A slot filler should have a significantly higher activation weight than the average slot filler, before it is judged dominant.

2. The number of slot fillers that are assigned to a slot. Measures will have to be taken to account for structural biases inherent to a concept, e.g.

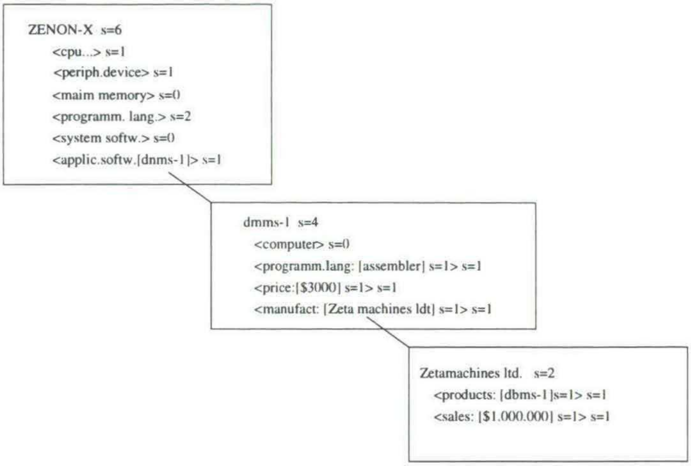


Figure 3.9: Frames in the German TOPIC system

the type slot in a computer frame will only have one possible slot filler, but the peripheral devices for that same kind of computer may have any number of potential fillers. All other things being equal, an instance of a computer frame where many periperal slots are filled, is more dominant than one where the peripheral slots have not been referred to.

3. The depth of nesting of the slot fillers. See for instance Figure 3.9, where slot fillers that are themselves frames, are nested. A slot is considered dominant if a frame is assigned to it such that the majority of the slots of that lower-level frame have been filled too (i.e. a significant degree of occupancy), or if a slot filler exists that is elaborated in more detail.

A further measure of importance that was investigated by Hahn and Reimer is the role of connectivity patterns based on a generalized hierarchies of frames. A number of active frames with a common superordinate frame may constitute a cluster of frames. This superordinate frame is called the *cluster frame.* Cluster frames are detected by recursively searching downwards from the most general concepts in the frame database and checking the number of active subordinate frames. At the level where significant loss of active concepts occurs, or rather at the level above, a cluster frame is declared. In this way, when sufficiently many frames that are directly referred to in the text, belong to a superordinate frame, this superodinate becomes part of the document knowledge representation.

The dominance measures result in a collection of formally unconnected concepts, which may be represented as linear graphs. The result is a text graph, which allows flexible, content-oriented access to full-text information, and may also serve as a nucleus for text generation.

3.5.3 Rule-based information retrieval

The system TOPIC by Verity Inc., is a commercial offshoot of the experimental RUBRIC system [McCune et al., 1985], [Appelbaum and Tong, 1988].

TOPIC is a complete system, with indexing modules, a retrieval engine and a user interface for interactive querying, but we will only consider the document representations and related issues here. We will refer to the system by the original name to avoid confusion with the German TOPIC system, the more so because after commercialization nothing much interesting has been published about the system.

RUBRIC (see also chapter 5) approaches document retrieval in two stages. In the first stage an inverted file is created of all strings occurring in the document. Positional information about paragraphs or particular segments of the documents is preserved in this inverted file. Together with Boolean and proximity operators and the ability to recognise fields in the document, this puts RUBRIC alongside systems such as STAIRS, that also enable Boolean retrieval on strings in full-text documents. The document representation that consists of the set of words occurring in the document, and that is stored in the inverted file, acts as a primary access mechanism. At retrieval time the original document is consulted to obtain information on the proximity of the words. Thus the document representation so far may be said to consist of terms from the complete text of the documents, which makes it a typical derived index.

The second stage that is grafted on top of this retrieval engine is a knowledge representation tool that may be thought of as essentially a weighted thesaurus, but that is implemented as a rule base. More importantly, this rule base is filled by the user and thus RUBRIC is a system for expressing personal preferences about documents, rather than an 'expert' on specific topics [Appelbaum and Tong, 1988]. Concepts are arranged in trees, or rather acyclic graphs, in which the strings that occur in the documents are the leaves (see Table 3.3). The occurrence of such strings, using Boolean and/or proximity operators, is taken as weighted proof for the relevance of the higher concept, and such concepts in their turn support other concepts.

To do this, RUBRIC uses two rules: EVIDENCE and IMPLIES. For example:

EVIDENCE moscow((*OR* "MOSCOW","KREMLIN")0.6))

where "MOSCOW"and ''KREMLIN" are text strings, and the number 0.6 is the degree of belief to be assigned to the concept *moscow* if either of the two strings are found in the document. If neither string is present, a zero degree of belief is assigned. The IMPLIES rule works slightly differently:

IMPLIES Jeltsin(moscow 0.3)

```
GENERAL-MOTORS
* 1.00 GM-COMPANIES
  ** 0.50 GENERAL-MOTORS-ACCEPTA-PHRS
    *** "general"
    *** "motors"
    *** "acceptance"
    *** "corp"
  ** 0.50 "gmac"
  ** 0.50 "hughes aircraft co"
* 0.50 GM-PEOPLE
  ** 1.00 GM-EX-CEO
    *** 1.00 "roger smith"
  ** 1.00 GM-PRES
    *** 1.00 LLOYD-REUSS
      **** "lloyd"
      **** "reuss"
  ** 1.00 GM-CEO
    *** 1.00 ROBERT-STEMPEL
* 0.50 GM-PRODUCTS
  ** 0.50 "pontiac"
  ** 0.50 "oldsmobile"
  ** 0.50 "buick"
. . . . .
```

Table 3.3: A topic in the RUBRIC system (indentation added)

where 0.3 is the degree of belief that the user wishes to assign to a document that describes Moscow, when in fact he is interested in documents about the Russian president with that name.

In a typical RUBRIC application many of such concepts (topics) will be built in advance by an information specialist, thus effectively adding a knowledge base to the system. Subsequently the user can build and add topics of his own, and these topics may or may not be accessible to other users.

This second layer may be considered part of the document representation too. Indeed, if a topic is built and documents are recognized by the rules in that topic, these documents are added to a list of postings for that topic. Thus it is relatively easy to extract the sets of topics that may be said to belong to a document (i.e. score above a threshold for that document).

The difference between the topics of RUBRIC and the entries of an orthodox classification system or a thesaurus is that the topics here ultimately are defined as properties of documents instead of in semantic terms. This gives the system a great flexibility, but also ample opportunity for snap decisions, ad hoc constructs and heuristics that may work fine in small collections, but may break down when applied to large databases. A possible reason for this breakdown is that in large databases different sub-populations of documents will come into existence, that all cover more or less the same subject, but approach it from widely different angles and (therefore) will use different vocabularies. Experiments [Gey and Chan, 1988] in which the performance of RUBRIC and the vector space model were compared, showed that the VSM using the cosine similarity measure, yielded comparable results to RUBRIC, with a slight edge of RUBRIC over the VSM for marginally relevant documents. We will return to the implications of this experiment in chapter 9.

An extensive summing up of the limitations of the commercial offshoot of RUBRIC is given in [Inc., 1990]. One should keep in mind, however that this report was written by an unsuccesful competitor of Verity, Inc. for the library system of Tilburg University.

3.5.4 Neural networks

As already mentioned, AI-approaches using scripts, frames or rule bases, foundered on the scaling problem, and mainstream IR returned to statistical and quantitative models and methods. Neural networks and genetic algorithms can also be considered numerical in that they operate on quantities of simple data, rather than by recognition and reconstruction of symbolic representations. A survey of models and methods in information retrieval would not be complete without mention of the attempts to use connectionist approaches to the problems of aboutness and document matching.

Neural networks are programs that are modelled after the workings of the animal brain. The brain consists of cells, called neurons, that are connected to each other. Also, some neurons are connected to sensory organs whereas other neurons are connected to e.g. muscles. In neural networks these cells correspond to the input and the output layers, respectively. When selected neurons in the input layer are activated by placing a pattern on the input, this activation spreads through the network by a system where every neuron collects the input it gets from other neurons and if the sum of these inputs is above some treshold, it in turn sends output to other neurons. Finally some neurons of the output layer may be activated, and the pattern of this activation is the output of the network.

Neural networks are *learning* systems, and their application in IR therefore is limited to the relevance feedback model (see chapter 4.3.3). However, in that chapter we only describe relevance feedback that is based on the presentation of the document as a vector. Here we give an example of the application of a neural network outside the document vector model: the AIR system [Belew, 1989].

In this system, AIR, ways are explored to improve the performance of retrieval by changing its document representation, using relevance feedback from users of the system. It operates on a database of bibliographic citations; each document is represented by the title, the author(s) and a number of keywords or descriptors. In the experiment described here, the keywords are taken from the title.

To start with, a representation of the information in the database is built by creating nodes for all documents. These nodes are connected with the nodes for the authors (one for every author) and the nodes for the keywords (one for every keyword), where for every connection there are two links. The links are initially weighted according to an inverse frequency weighting scheme. The

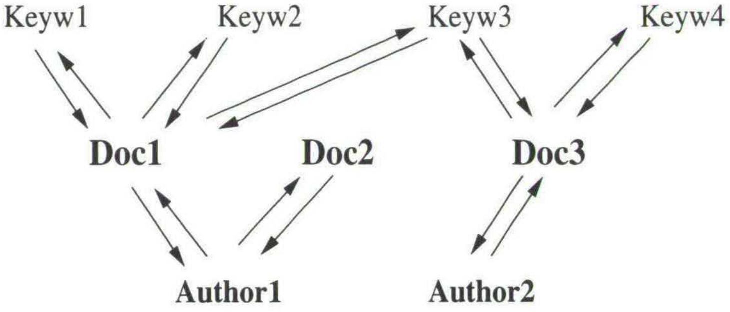


Figure 3.10: AIR neural network

sum of all the weights departing from a node is constant. Figure 3.10 shows a very simple network with four keywords, three documents and two authors.

When an initial query is put to the system all nodes that correspond to that query are activated, and this activity is allowed to propagate through the system. The answer of the system is ranked according to the final activation of the nodes and presented to the user.

Subsequent queries in the same session are performed differently. The user has to indicate which features (nodes) from the ranked features he judges relevant and which irrelevant on a scale of $++, +, -, --$. Not all features have to be commented upon. The system creates a new query based on this feedback, strengthening or weakening the links according to this scale, and so is effectively trained by the users to recognize associations that are useful for IR.

When a query contains a new term, i.e. one for which no node exists, the query is first handled without that term and subsequently (after the user's response) a new node is created for that term and connected to the network.

The net result of all this is that the network will evolve towards a consensus of users about what keywords and documents belong together. This 'democratic' view of the aboutness of documents contrasts with the omniscient notion of aboutness that is present in almost all other IR-systems. That is: the relevance of a document with respect to a query in classical IR systems tends to be absolute, as if determined by an omniscient indexer.

3.6 Conclusions

In this chapter we have given an overview of how automation became an important tool for IR. In the beginning of the era of automation, computers could only perform elementary clerical tasks, but soon attempts were made to extend the assistance of the computer to more complicated areas, notably those to do with natural language processing and even 'understanding' of texts. This culminated in the late eighties in a number of AI-based IR systems, of which we presented some salient examples.

In the meantime, experiments with frequency-based processing went on. In the late sixties and seventies the principles were already established by prominent scholars like Edmundson, Salton, Van Rijsbergen and Sparck Jones, but progress was hampered by the absence of sizable quantities of machine-readable texts. When the micro-computer became a common appliance, and the availability of electronic storage increased correspondingly, at the beginning of the present decade critical mass was reached and quantitative and statistical NL-processing soared.

As we already observed, there are many techniques that only could be implemented efficiently when computers are available, and the most important of these techniques have to do with vectors. In the next chapter we will present techniques and models for the creation of document representations as vectors, and the similarity functions that can be applied to compare those vectors.

Chapter 4

A survey of vector-based IR models

As we have seen in chapter 1, several models of IR have been proposed, but there seems to be no generally accepted taxonomy that describes the various models in relation to each other. The models that are mentioned in the literature often are local descriptions of *parts* of the general model that was introduced in chapter 1 as the 'classical' model and that will be described in more detail below. For example, the Boolean model and the vector space model describe similarity functions and the way that documents are divided into relevant and non-relevant, but they have little or nothing to say about, for example, the translation of documents into the index language, although this translation obviously has a strong influence on the behaviour of the system as a whole.

In this chapter we will describe the models that rely on a vector representation of the document and discuss their relation to each other and to the general, all-compassing 'classical' model of IR as depicted in Figure 4.1 and Figure 4.2. We will consider these models as a separate group and in section 4.8 we will argue that this 'document vector family' can act as a model in its own right.

4.1 The general model of IR

We will first consider the general model of IR as explained in [Fuhr, 1995] and displayed in Figure 4.1. Here, $\underline{D}$ and $\underline{Q}$ denote respectively the documents in a database and a set of queries. If $\mathcal{R} = \{\mathsf{R}, \bar{\mathsf{R}}\}$ denotes the set of possible relevance judgements (assuming that a document is either relevant or not relevant to a query), then the relevance relationship can be regarded as a mapping $r : \underline{Q} \times \underline{D} \to \mathcal{R}$. D and Q are semantic representations of the documents and the queries, created from the original objects by the mappings α_D and α_Q. The mappings β_D and β_Q are used to translate D and Q into compatible rep-

$$\begin{tikzcd}
 & & \underline{Q} \arrow[r, "\alpha_Q"] & Q \arrow[r, "\beta_Q"] & Q^D \arrow[dr, dash] & & \\
\mathcal{R} & \mathrm{r} \arrow[l] \arrow[ur, dash] \arrow[dr, dash] & & & & \rho \arrow[r] & \mathrm{R} \\
 & & \underline{D} \arrow[r, "\alpha_D"] & D \arrow[r, "\beta_D"] & D^D \arrow[ur, dash] & &
\end{tikzcd}$$

Figure 4.1: Conceptual model of IR

resentations D^D and Q^D, e.g. to create document vectors and a query vector in a common vector space.

The similarity function ρ then compares D^D and Q^D and computes the retrieval weight R, which may be either real- or binary valued. A possible weakness of this model is the assumption that the relevance judgments $\mathcal{R} = \{\mathsf{R}, \bar{\mathsf{R}}\}$ in this model are binary-valued even if the retrieval weight R may be real-valued.

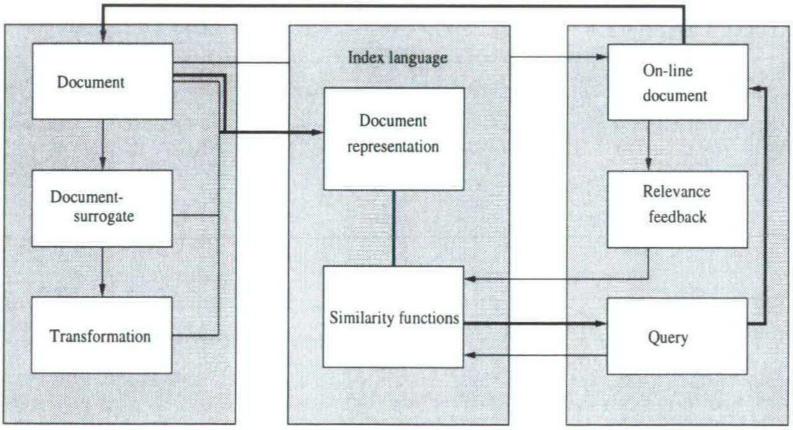


Figure 4.2: Classical model.

We also present a less formal description of IR in Figure 4.2. As in the Fuhr model described above, a set of documents is distinguished, a set of requests and some mechanism for determining which, if any, of the documents meets the requirements of the request. These three items correspond with the three shaded areas in the figure. The following areas and activities may be recognized:

1. the translation from the document to the index language (IL) , which is shown in the left-hand box (and a corresponding translation from the query to the IL in the lower right-hand box). This translation may be done directly from the document, indirectly from a document surrogate, via transformations of existing vectors, or by a combination of these possibilities.

2. the index language itself, in which the document representations and a representation of the query are stored. The similarity functions that are used to compare documents and queries operate on expressions of the index language. Similarity functions are discussed in detail in the second part of this chapter.

3. the interaction between system and user, which centers on the presentation of the results of the query. Note the *on-line document*, often the bibliographic reference, that is yet another translation from the original document, and that takes the place of the document in the presentation of the results. Relevance feedback also belongs to this area as it requires the availability of some on-line representation for the user to judge the relevance of each document.

The thick line indicates the way through the system that is necessary for a complete retrieval action, including the retrieval of the original document. This last step is often omitted, e.g. in cases where no relevant documents were retrieved or when the on-line document offered sufficient information.

The tacit assumption in both models, indeed in all models of information retrieval, is that the document keeps its identity in the index language. This is the property that makes IR different from other information systems, such as expert systems or database management systems: the items that are retrieved are *documents* or pointers to documents. For instance, the SCISOR system mentioned in the last chapter does not build document representations but knowledge structures that are filled with information from several documents, and the identity of the individual documents is lost. The answer to a query in the SCISOR system therefore is not a list of documents, but a fact or a piece of information. Therefore SCISOR is not an IR system, but should properly be called a *data retrieval* or *question answering* system. On the other hand, the German TOPIC should be considered as a IR system, because a query put to the system leads the user to a document or text passage (we will ignore the issue that 'information retrieval' actually is a misnomer for this discipline and that 'document retrieval' would be a more accurate description of the field).

But what if a system like SCISOR would also return the documents that had contributed to the information it returns? This would be possible only if the documents had kept their identity in some way or other, in which case it would again conform to the models presented here.

4.2 The document–IL translation

In the chapters 1 and 2 we already introduced the concepts of the document vector and the index language. The term 'language' is justified by the index language (IL) being a representation format with both a syntax and semantics. The shape and contents of the IL depend on at least two design decisions: the features in the original document that are extracted and the format in which they are stored. We use the term *document representation* for the residue of the document that is represented in the IL. This representation can be as simple as a set of keywords, or as complicated as a term from Ranganathan's colon classification or one of Farradane's analets (see chapter 2).

A fundamental distiction between approaches to document translation is that of *assigned* vs. *derived* indexing. The difference between the two is whether the keywords that are chosen for use in the IL are taken from the document itself (derived indexing) or from an independent list of terms, a classification system or an ontology.

Assigned indexing is not found in automated IR, although text categorization and classification systems often assign documents to classes whose names do not occur in the documents. It may be argued that the vector elements resulting from singular value decomposition (SVD, explained below) do not occur in the documents either. Nevertheless, such elements are often derived automatically from items that are themselves derived directly from the documents. The question whether a SVD matrix is assigned or not therefore depends on how the original vectors were obtained.

Derived indexing, where the descriptors are taken from the document (after a certain amount of preprocessing, possibly involving SVD or similar methods), is the prevalent method in automated IR. These descriptors can be words, n-grams, collocations or any other feature that is judged useful for the division of the mass of documents into relevant and not-relevant. The majority of systems use words. In chapter 1 we already identified the most important methods of selection and reduction of such tokens: filtering, truncation and the computation of weights. These actions may be considered as the first step towards the creation of document vectors, and they define the length of the vector (which equals the number of word types in all documents). The weight that is stored for an individual word-document combination (and that hopefully is an indication for the information value that this particular word has for the database, or even for the individual document) may then be computed in two ways: as *plain word weights*and as *word-document weights*. Before we discuss the methods that are employed to compute such weights, we will first consider the Boolean model and its position with regard to the document vector model.

4.3 The Boolean model

The Boolean model does not represent a document as a vector, but as a set. Such a set may be mapped into a binary vector, but this is only possible when the system knows *all* the word types that occur in the database. In other words, the essential difference between the Boolean model and the document vector model with regards to its storage format, is not that the word weight in the Boolean model is a binary value, but that in the latter model the length of the document vector is known and may be used in computations.

The Boolean model can be retraced to the first postcoordinate systems, such as Taube's Uniterm system ([Foskett, 1982], pag.435). We have already noted that, despite its many shortcomings, this model has remained in general use till the present day. In the Boolean model the document representation consists of the set of keywords that have been assigned to it by one way or another; the similarity function manipulates these sets with the Boolean operators AND, OR and NOT. Thus the retrieval- or similarity function $sim(q, d)$ between a query q and a document d may be defined recursively as:

$$\begin{aligned} t_i \in T \Rightarrow sim(t_i, \vec{d_m}) &= d_{m_i} \\ sim(q_1 \vee q_2, \vec{d_m}) &= max(sim(q_1, \vec{d_m}), sim(q_2, \vec{d_m})) \\ sim(q_1 \wedge q_2, \vec{d_m}) &= min(sim(q_1, \vec{d_m}), sim(q_2, \vec{d_m})) \\ sim(\neg q, \vec{d_m}) &= 1 - sim(q, \vec{d_m}) \end{aligned}$$

where $T = t_i, ..., t_n$ are the terms in the database, q_1 and q_2 are query terms, d_m is a document and $\vec{d_m}$ is de corresponding document vector. As the weights in the document vector are binary, the result of $sim(q, \vec{d})$ also is either zero or one and an unadorned Boolean system always produces a weak ordering of the documents into retrieved and not retrieved.

If all document representations are different, any document or selection of documents can be retrieved in the Boolean model using this similarity function, ([Fuhr, 1995], see also Lewis' *Perfect Query Paradox* in chapter 1, section 1.4.5). This theoretical advantage is canceled by the fact that in order to achieve this, the user must know the targeted document representations in detail. In a practical information retrieval situation this is almost by definition *not* the case. As the Boolean model returns a weak ordering of documents in terms of $R, \bar{R}$, a trade-off between precision and recall has to be made. The user will run the risk that he will either reach the *futility point* (a term coined by Blair [Blair, 1980] and explained in the next section) before he has seen all relevant documents in the result set, or that many relevant documents will not be included in the result at all.

If the Boolean model is used with a well-designed system of keywords, and if the use of the retrieval system is restricted to users who are well acquainted with this keyword system, then it may perform well. But if the set of keywords that represent the documents are created by derived indexing of a full-text document, new problems arise at retrieval time.

The problem with all retrieval systems of this type is that human language is fuzzy. There may be as many as a dozen different terms and words pointing to one and the same object (synonyms), whereas one word may have widely different meanings (homonyms). But even where words are used in the same sense, expressions containing the words can vary dramatically in meaning as a result of the syntactic and semantic combinations with other words.

```
This book is about dogs, not about cats.
This book could have been about dogs if it had not been about cats.
This book is about intestinal parasites of dogs and cats.
```

In the above example, all three texts will be retrieved on the query 'dogs AND cats', but it is highly improbable that anyone of them will be relevant for the query. Successful indexing then becomes a matter of predicting which terms will be used in a query to retrieve documents that contain information on a certain subject, and successful querying on the other hand is guessing which terms have been used to index the document that is sought. The result of both may be expressed as probabilities, and the probability of retrieving the documents in which one or more keywords occur in an AND query is the product of these two probabilities for every keyword that was used in the query.

In information retrieval according to the Boolean model, this will lead to either of two extremes. Either one aims at a high precision, when almost all the retrieved documents are relevant (but an unknown number of other relevant documents are not included), or one goes for high recall, but then a number of irrelevant documents will be included in the result. When in a retrieved set of documents the proportion of irrelevant documents is high, the user may well stop looking at the documents before he has found all the relevant ones: his *futility-point* has been reached. In such a case the net result is equivalent to the situation where those relevant documents that would be presented after the user reached the futility-point were not retrieved at all. Therefore the concept of ranking, i.e. the ordering of retrieved documents on estimated relevance, is of great importance in automated information retrieval, allowing the user to be reasonably certain that the most relevant documents are presented first.

The *fuzzy retrieval model* is an attempt to improve on the Boolean model by trying to ascertain a ranking of the retrieved documents on estimated relevance, where the keywords both in the queries and in the document may have a weight, e.g. the $tf.idf$. Within the Boolean retrieval function as defined above we give an example below, where the query t_1 AND t_2 is presented to a database with two documents with respectively the weights (0.4, 0.4) and (0.39, 0.99) for these two terms:

$$\begin{aligned} T &= \{t_1, t_2\} \\ q &= t_1 \wedge t_2 \\ \vec{d_1} = (0.4, 0.4)\ ,\ \vec{d_2} &= (0.39, 0.99) \\ sim(q, \vec{d_1}) = 0.4\ ,\ sim(q, \vec{d_2}) &= 0.39 \end{aligned}$$

Because according to the definition of the Boolean model that was given earlier, the retrieval function of $q = t_1 \wedge t_2$ returns the lower of the two values, the two documents are ranked on respectively 0.4 and 0.39. The example also exposes the weakness of the fuzzy Boolean model: although the second document clearly has the higher indexing weight, it is ranked below the first one.

4.3.1 Frequency–based weights

We now return to the discussion of word weights. Word weights come in two 'flavours': one in which the weight is related only to the keyword itself, so that it is the same for all occurrences of a keyword, and one in which the properties of individual documents are also taken into consideration. As we know of no standard terminology for these two groups, we have coined the terms *plain word weights* for this first group of weights and for the second group *word-document weights.* In almost all extant models, the properties that combine to form the weight of the keyword or term i are the three frequency figures *term frequency* (tf_{ik}), *document frequency* (df_i) and *collection frequency* (cf_i), being respectively the frequency of term i in the document k, the number of documents in which the term i occurs and the total number of times that the term i occurs in the collection. N is generally reserved to represent the number of documents in the database.

The relation between the collection frequency and the document frequency plays an important role in most weighting schemes. This is intuitively clear when we consider two words that have the same collection frequency, but differing document frequencies. This last measure then indicates how well the word discriminates between documents. Bad discriminators are spread out over all the documents, while the words that are good discriminators will appear many times in a small number of documents.

In this subsection we will consider two examples of plain word weights: the Poisson models and the discrimination value model, followed by the most popular of the word-document weights: the $tf.idf$ weight.

Plain word weights

Poisson models Perhaps the simplest scheme by which to weight the usability of a word as a keyword, i.e. a word that by its occurrence separates the body of documents into two separate groups, relative to an information need, is its deviation from the Poisson distribution. This distribution describes the probability that a certain random event occurs a certain number of times over units of fixed size. When we equate the event to the occurrence of a keyword i, the number of times that it may occur to k and the units to documents, we can use the equation:

$$P_i(k) = e^{-\lambda_i}\frac{\lambda_i^k}{k!}$$

Applied to documents and keywords, $P_i(k)$ is the probability that a document has exactly k occurrences of word i if documents are random collections of words. The Poisson distribution can thus be used to predict the number of times a term occurs in a document, if the terms are distributed at random over the documents. In the formula, λ_i is the collection frequency of term i divided by the number of documents, $\frac{cf_i}{N}$. As the mean M and the variance V of a Poisson distribution are the same, both also equal λ and therefore for every keyword i it is true that:

$$M(P_i) = V(P_i) = \lambda_i$$

The Poisson distribution applies to terms in documents if the probability of an occurrence of that term in a piece of text is proportional to the length

word	coll.fr	doc.fr	lambda	Poisson(0)	N(1-P(0))	overest.	rank
note	112	110	0.1120	0.8940	105.9557	0.9632	1
st	117	113	0.1170	0.8895	110.4148	0.9771	2
previously	46	46	0.0460	0.9550	44.9580	0.9773	3
result	44	44	0.0440	0.9569	43.0460	0.9783	4
says	43	43	0.0430	0.9579	42.0886	0.9788	5
fullerton	10	1	0.0100	0.9900	9.9501	9.9501	9812
levy	10	1	0.0100	0.9900	9.9501	9.9501	9813
pentland	10	1	0.0100	0.9900	9.9501	9.9501	9814
revlon	10	1	0.0100	0.9900	9.9501	9.9501	9815
statoil	10	1	0.0100	0.9900	9.9501	9.9501	9816

Table 4.1: Words with lowest and highest Poisson-overestimation from the Reuter corpus.

of that text and if the occurrence of terms is independent from previous or subsequent occurrences. This latter assumption holds for function words and does not hold for content words.

The document frequency is the easiest way to check whether a word is Poisson-distributed. The Poisson distribution predicts that the document frequency, or the number of documents in which keyword i occurs at least once, equals $N(1 - P_i(0))$ or the complement of the predicted number of documents *without* the word i. If this is the case, the word under consideration may be considered a function word or a content word with a low information value - we will call such words *low-content* words. Content words with a high information value, on the other hand, tend to cluster and will cause the estimated document frequency to be higher than it is in reality.

In Table 4.1, the second and third columns give the collection frequency and the document frequency respectively of words in the Reuter corpus (see chapter 7 for more information about this corpus). In the next three columns we see the value of lambda, the probability of zero occurrences of the word under consideration, and the estimated document frequency $N(1 - P_i(0))$. The last two columns give the overestimation for that word, calculated as $\frac{N(1-P_i(0))}{D_f}$ and the rank of the word according to its overestimation. All examples of this chapter are taken from a small corpus of a thousand documents from the Reuter collection and N therefore equals 1000.

For content words it is found that the simple Poisson distribution (simple as opposed to the two-Poisson and multi-Poisson models) overestimates the df_i, an effect that is caused by the fact that content words do not occur independently. On the contrary, once a content word occurs in a document it is likely to occur again. The effect is demonstrated clearly in the table by the fact that the words with the highest overestimation all are company names, i.e., words with a very high information value.

Better fits for content words are found by the *two-Poisson* model [Bookstein and Swanson, 1975]. This model assumes that a content term is better described by two classes of documents associated with that term: one class (1) with a low average number of occurrences and one (2) with a high average of occurrences:

$$P_i(k) = \pi e^{-\lambda_1}\frac{\lambda_1^k}{k!} + (1-\pi)e^{-\lambda_2}\frac{\lambda_2^k}{k!}$$

where π and $(1-\pi)$ are the probabilities of a document being in class (1) and (2) respectively. [Harter, 1975] describes how to estimate the parameters π, λ_1 and λ_2 if no a priori information on relevance or class membership is available. Other approaches embroider on this idea by postulating an infinite number of Poisson functions (e.g, the *Negative Binomial* [Mosteller and Wallace, 1984]; we will meet these authors again in chapter 8), but these approaches are computationally very expensive.

The discrimination value Another expensive method is the computation of the discrimination value of a term, which is the influence that a term has on the mutual similarity of the documents. The documents are viewed as a cloud; keywords that represent the documents influence the density of the cloud: 'good' keywords bring similar documents closer to each other and farther away from dissimilar documents. The discrimination value of a keyword Dv_i is computed by comparing the average density Q of the document-cloud in which the keyword i is part of the document vector, with the average density Q_i of the cloud *without* keyword i:

$$Dv_i = Q - Q_i$$

If the database is represented as a term-document matrix with documents as rows of M distinct terms $t_1, t_2, ..., t_M$, Q is computed by taking the average $(N(N-1))$ pair-wise similarity values of all possible document pairs:

$$Q = \frac{1}{N(N-1)}\sum_{i=1}^{N}\sum_{k=1,i\neq k}^{N} sim(D_i, D_k)$$

where N is as usual the number of documents and D_i and D_k are documents. This is simplified by constructing a dummy document at the centre of the document cloud: the *centroid* $C = (c_1, c_2, ..., c_T)$, where T is the number of terms and where every c_j is the mean of all j^{th} terms in the document base:

$$c_j = \frac{1}{N}\sum_{k=1}^{N} d_{kj}$$

where d_{kj} is the j^{th} term in the k^{th} document. The formula can be now simplified to:

$$Q = \frac{1}{N}\sum_{k=1}^{N} sim(C, D_k)$$

There is a variety of techniques with which to compute the similarity of document vectors; see e.g. [van Rijsbergen, 1979]; we will discuss the

frequency-based				atc-based			
highest Dv-values		lowest Dv-values		highest Dv-values		lowest Dv-values	
japan	0.000378	the	-0.044711	gte	0.000068	blah	-0.003791
first	0.000380	of	-0.014533	roto	0.000069	vs	-0.002254
march	0.000388	to	-0.009495	usbc	0.000072	earn	-0.001631
oil	0.000408	said	-0.007872	diluted	0.000073	cts	-0.001463
quarter	0.0004307	and	-0.007607	snat.o	0.000074	net	-0.001236
debt	0.000448	in	-0.005252	mar	0.000079	shr	-0.000952
he	0.000452	topics	-0.004737	bkne.o	0.000082	mln	-0.000925
texaco	0.000491	a	-0.004521	ibcp.o	0.000083	qtr	-0.000836
that	0.000507	title	-0.004324	wpi	0.000089	inc	-0.000662
u.s	0.000547	reuter	-0.003072	fcom	0.000090	dlrs	-0.000627
trade	0.000568	mln	-0.002192	orange	0.000093	st	-0.000581
bank	0.000601	for	-0.001223	rhnb	0.000099	pct	-0.000538
shares	0.000729	it	-0.000909	ncr	0.000101	corp	-0.000529
loss	0.000754	vs	-0.000491	lei	0.000104	year	-0.000429
billion	0.000826	its	-0.000159	oper	0.000105	from	-0.000401
pct	0.000938	earn	-0.000127	fx	0.000121	it	-0.000400
blah	0.002272	dlrs	-0.000109	kdi	0.000136	its	-0.000400

Table 4.2: Words with lowest and highest discrimination value computed for different word weights

most important similarity functions in section 4.4. The most commonly used method involves the cosine function, which is also used in the experiments of [Willett, 1985], [El-Hamdouchi and Willett, 1988] and [Crouch, 1988].

As we already indicated, this method is computationally rather expensive. For our own experiments we adapted a program for the computation of discrimination values that was originally published in [Frakes and Baeza-Yates, 1992], and that we optimized for speed[1].

We can conclude from the equations given above that the discrimination value is not only dependent on the data, but also on the similarity function and on the word weights. As an example we present Table 4.2. It shows both the highest and lowest discrimination values of the words in the Reuter database and two different methods to compute this value are displayed. The left-hand half of this table shows discrimination values that are based on plain term frequencies. We observe here similar tendencies as in the Poisson-based table 4.1; there are strong differences between the words in both columns. The words that rank highest tend to be nouns and proper names, while the low-ranking words mainly are composed of function words (the words 'title', 'reuter' and 'topics' in this database are special cases as they occur in every document).

In the right-hand half of the table, the values are computed from the *tf.idf*, a measure that also takes both the term frequency and the document frequency of the term into account and that is very succesfull as a weight in, e.g. vector space comparisons. We will describe it in more detail below. It is clear that this ranking is rather different from the left-hand part of the table and

[1] The source of this program and other relevant programs can be downloaded from http://pi0959.kub.nl/Paai/Publiek.

it is difficult, if at all possible to detect a pattern. This particular weight, when combined with the discrimination value, clearly does not perform well in separating content from non-content words.

Word-document weights

The *tf.idf* family of weights The document vector model lends itself well to systems that can create a different weight for every word–document combination. This weight can also be applied as a threshold that causes certain document-keyword combinations to be ignored. The vector that contains the frequency of the word in the document may be considered as the simple case.

In the literature on IR we often see the use of weighting schemes that produce a word-document weight presented as more or less identical with the vector space model (VSM, described in the second half of this chapter). In our view this is not correct. For one, the similarity functions of the VSM model work with plain word weights or even binary weights as well as with word-document weights. Moreover, word-document vectors in their turn can be applied in many non-VSM models. We therefore prefer to see the document vector model as a model in its own right, that may be used as foundation for other, more specialized models.

There are a number of weighting schemes that use the frequency of the words within documents and their distribution over the database as a measure for the suitability of a word as a keyword for a particular document. The most popular of these schemes is the so-called *tf.idf* weight, or rather *one* of the *tf.idf*-related weights, as there are several variations (see Table 7.8 at the end of chapter 7). The *tf.idf* is composed of the term frequency (tf) and the inverse document frequency (idf) or one of its derivates or normalizations (these terms have been defined above, in section 4.3.1). An appropiate indication of a term as a document discriminator can be computed by taking an inverse function of the document frequency of that term, e.g. $idf = N/df_t$, for N documents, or $idf = logN/df_t + 1$. The product of the term frequency and the inverse document frequency, *tf.idf*, may then be used as an indicator of the importance of a term in a document.

A popular variant of this *tf.idf* is the so-called *atc*-weight (again refer to Table 7.8 for the meaning of the abbreviation 'atc'). It calculates the *tf.idf* in three steps. First the value new_tf for the term-frequency (tf) is calculated as

$$new\_tf = 0.5 + 0.5 * \frac{tf}{max\_tf}$$

where max_tf is the frequency of the term with the highest frequency in the document. This normalizes for the length of the document and dampens the importance of the frequency of the word. Then the weight new_wt is calculated as

$$new\_wt = new\_tf * log\frac{N}{df_t}$$

lowest		highest	
0.011224	said	0.570510	sign.o
0.011765	of	0.572890	clothiers
0.012653	and	0.573540	dominion
0.014061	the	0.582080	oakite
0.014166	to	0.582080	okt
0.015881	a	0.583650	sqd
0.016254	in	0.599450	pch
0.020348	for	0.599450	potlatch
0.021512	it	0.605270	parc
0.026160	on	0.638100	usbc
0.027267	its	0.665720	snat.o
0.028655	with	0.677160	ibcp.o
0.029178	by	0.680160	roto
0.030047	is	0.696910	fcom
0.030340	from	0.711320	rhnb
0.031866	be	0.772500	kdi
0.032112	at	0.906979	blah

Table 4.3: Lowest and highest (average) *atc* weights, sorted from low to high

where N is the number of documents and df_t the document frequency of term t. The logarithm scales the effect of the number of documents in which the keyword occurs. It gives full weigth to words that occur in one document ($logN - logdf_i = logN - log1 = logN$), but a word occurring in all documents gets zero weight ($\log N - \log df_i = \log N - \log N = 0$).

A final normalization is applied by taking the square root of the sum of the squares:

$$new\_wt' = \frac{new\_wt}{\sqrt{\sum_{i=1}^{T} new\_wt_i^2}}$$

where T is the number of terms in the document vector. For a detailed discussion of these and similar techniques see e.g. [Salton and McGill, 1983] and ([Salton, 1989]).

In Table 4.3 we display the highest and lowest scores for words weighted with this particular algorithm. Because in these word-document weights a word does not have a single weight, but a separate weight for every document it occurs in, we have taken the mean of the weights for a particular term and displayed the terms with the highest and lowest *averages*. Again, we see a clear difference between the two columns; the words with low $tf.idf$ weights are almost all function words; the words with high $tf.idf$ weights in this case are names and abbreviations.

An interesting term is 'blah', the word with the highest average *atc* value in the table. The string 'blah blah blah' was used by the editors of the Reuter collection to fill empty documents, and the word 'blah' therefore has after normalization as described above a very high term frequency, and hence displays a very high *atc* value for all documents that it appears in.

4.3.2 Differences between the two methods

The interpretation of plain word weights is very unlike that of word-document weights, such as the *tf.idf* variations. In the latter case a document-word weight is computed that will be different for every document; in the former case a word weight remains constant over the database. If words in documents are to be weighted individually using a discrimination value, they are multiplied with the frequency of the word, which according to [Salton et al., 1975] gives 'excellent' retrieval results.

Perhaps the earliest attempt to weight words according to some objective estimate of informativeness is by [Luhn, 1958]. He proposed to first rank the words of a document according to their frequency and draw a Zipf curve showing the relation between frequency and rank (Figure 4.3). Words with a high frequency are function words and therefore poor indicators of the contents of a document. Rather controversial is that, according to Luhn, the same is true for words with a very low frequency, that therefore do not contribute significantly to the information contained in the document.

This theory can be extended to using the word frequencies of a database of several documents. Again, function words are found in the high-frequency end, whereas for words with a very low frequency it can be argued that they also are poor discriminators. The best discriminators according to this school of thought, are words that divide the mass of documents in two sizable parts. Their distribution is a bell-shaped curve between the upper- and the lower cutoff points (see Figure 4.3).

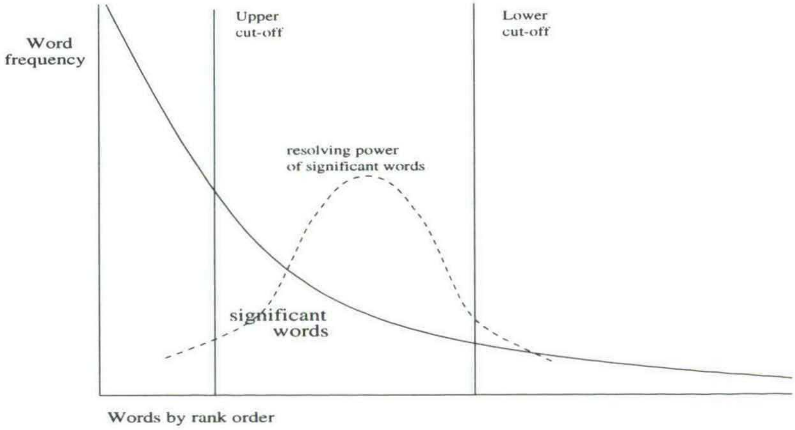


Figure 4.3: The relation between the word frequency and the rank order [van Rijsbergen, 1979].

4.3.3 Feature reduction models

In the two methods described above, the features created by the translation from the document into the document representation in the index language remain intact. After the application of filters and different weighting methods, a number of features are selected to represent the document. This is why we favour the expression 'feature *selection*' for this kind of operations.

A different method is the re-mapping of the original features on a smaller number of new features. Here we like to use the expression 'feature *reduction*' or 'transformation'. A method of feature reduction that has received much attention is *latent semantic indexing* [Deerwester et al., 1990]. This reduction is brought about by applying singular value decomposition (SVD) to the original document-keyword matrix, creating a new semantic space in which both documents and keywords can be mapped. If the relation between each keyword and each document is expressed in a $d : t$ matrix of weights (w), where d is the number of documents and t the number of terms, the application of SVD creates three new matrices; a $d : s$ matrix (W), a diagonal $s : s$ matrix (S) and a $s : t$ matrix (T).

$$\begin{bmatrix} w_{0,0} \cdots\cdots w_{0,t} \\ \cdots \\ \cdots \\ \cdots \\ w_{d,0} \cdots\cdots w_{d,t} \end{bmatrix} = \begin{bmatrix} W_{0,0} \cdots W_{0,s} \\ \cdots \\ \cdots \\ \cdots \\ W_{d,0} \cdots W_{d,s} \end{bmatrix} \begin{bmatrix} S_{0,0} \cdots \\ \ddots \\ \cdots S_{s,s} \end{bmatrix} \begin{bmatrix} T_{0,0} \cdots\cdots T_{0,t} \\ \cdots \\ \cdots \\ T_{s,0} \cdots\cdots T_{s,t} \end{bmatrix}$$

This new s-dimensional space describes the co-occurence of the original keywords and the diagonal matrix S is ordered in such a way that the upper left elements describe strong co-occurence tendencies of documents when expressed in keywords and vice versa. Towards the right lower part of the diagonal, only spurious co-occurrences and weak relations occur. By keeping the n first singular values and zeroing out the others, a semantic space can be defined in which to compare douments, keywords or combinations of both.

4.4 Vectors and similarity functions

In the previous sections we have considered examples of document–vector translations, where each element in the vector contained a measure for the (estimated) relevance of a keyword for that document. In this chapter we will see how such vectors may be compared to each other or to a query vector. We will also discuss the potential of the *relevance feedback* model, which is largely dependent on vector representations and similarity functions.

The majority of the models and methods that are described in this chapter have been developed in the late sixties and have been applied at least experimentally in working programs [Salton, 1971]. Nevertheless, as late as in 1989 the same methods are characterized as 'advanced methods' [Salton, 1989] and even in 1996 [Blair, 1996] concludes that the majority of commercial systems prefer to ignore the work done in this field, and to adhere to the Boolean model, notwithstanding the severe doubts as to its effectivity. He suggests a number of causes for the reluctance to use these 'advanced' methods, not the least of which is the strong suspicion that these methods are relatively difficult to understand, while the Boolean system is conceptually somewhat simpler (see also [Paijmans, 1996]).

4.4.1 Topical similarity

The purpose of similarity functions according to the Boolean model of IR is to divide a collection of documents in two classes, those that are similar to a query, and those that are not. The similarity functions in the vector space model and other models mostly do not result in two mutually exclusive classes, but they *rank* the documents according to the similarity with the query. What exactly makes a document similar or dissimilar to a query is not a simple matter, but centers around the concept of the 'aboutness' or 'topicality' of documents and queries, also expressed as *semantic similarity.* An obvious interpretation of semantic similarity is to consider it as a weak form of synonymy. But it can also be used to convey that words are from the same semantic domain, such as the words *merger*, *take over*, *acquire* or *Brucks Plastic* in the SCISOR-examples of chapter 3. Another example of such relations between keywords would be the *document space*, defined by the documents in which they occur. In this way, one can distinguish many 'spaces' in which the relations between the keywords, semantic or otherwise, can be represented.

In Table 4.4 we give an example with six 'documents', five keywords and three 'spaces'. Apart from the document space, we demonstrate also a *nationality space* and a *keyword space.* The figures in the table labeled *keyword space* give the co-occurrences of the keywords, the table *document space* gives the frequencies of the keywords in the six documents and the table *modifier space* the co-occurrence of modifiers and the keywords.

In the area marked 'keyword space' is demonstrated how words may be considered similar to the extent dat they co-occur with other keywords. For instance, *Gaudí* co-ocurs in three documents with *Barcelona* and *history*, in two with *art*, but never with *dog* or *cat.* Therefore, *Gaudí* may be considered semantically similar to *history* and *art*, even if it can in no way be considered a synonym.

In the middle part of the table, document space may be demonstrated intuitively by comparing the fact that documents with similar contents tend to have the same, or similar words. Documents on Barcelona also tend to contain the words *history*, *art* and *Gaudí*, but cats and dogs seem to live in a different universe.

In the lower part of the figure, keywords are matched with modifiers, in this example nationality adjectives. *Art* and *history* do match with *greek*, *dog* does match with *english* and *german* and *cat* with *english* and *persian*. *Gaudí* and *Barcelona* co-occur somewhat more strongly with *spanish*, perhaps indicating that both concepts are more related to this nationality than the animals do with their respective countries.

Documents

1. The Spanish city of Barcelona has except for the Spanish architect Gaudí many more examples of art. The museum boasts a large collection of Greek art.
2. Barcelona is a city rich in Spanish and Greek history. Architecture is represented by Gaudí.
3. Art, history, Gaudí.
4. The care and feeding of Persian cats is unlike that of the English cats. Of course it is rather dissimilar to that of German and English dogs too. German dogs in history are not to be confused with English cats for obvious reasons.
5. The Spanish city of Barcelona has many examples of Italian art, among which some monumental buildings by the Spanish architect Gaudí. Some say that there also exists an Italian Barcelona, but this is not true.
6. The Spanish architect Gaudí created many monumental buildings.

keyword space						
	Barcelona	art	history	Gaudí	dogs	cats
Barcelona	3	2	1	3	0	0
art	2	3	1	3	0	0
history	1	1	3	2	1	1
Gaudí	3	3	3	5	0	0
dogs	0	0	1	0	1	1
cats	0	0	1	0	1	1
document space						
	Barcelona	art	history	Gaudí	dogs	cats
d1	1	1	0	1	0	0
d2	1	0	1	1	0	0
d3	0	1	1	1	0	0
d4	0	0	1	0	1	1
d5	1	1	0	1	0	0
d6	0	0	0	1	0	0
modifier space						
	Barcelona	art	history	Gaudí	dogs	cats
spanish	2	0	1	3	0	0
greek	0	1	1	0	0	0
german	0	0	0	0	2	0
persian	0	0	0	0	0	1
english	0	0	0	0	1	2
italian	1	1	0	0	0	0

Table 4.4: Keywords in three kinds of spaces

4.4.2 The vector space model

In research environments the document–vector translation approach, in particular the vector space model (VSM) has received the most attention. It was developed thirty years ago by Salton and his staff in the context of the SMART project [Salton, 1971], [Salton and McGill, 1983] and it has been the underlying model for many experiments and improvements since.

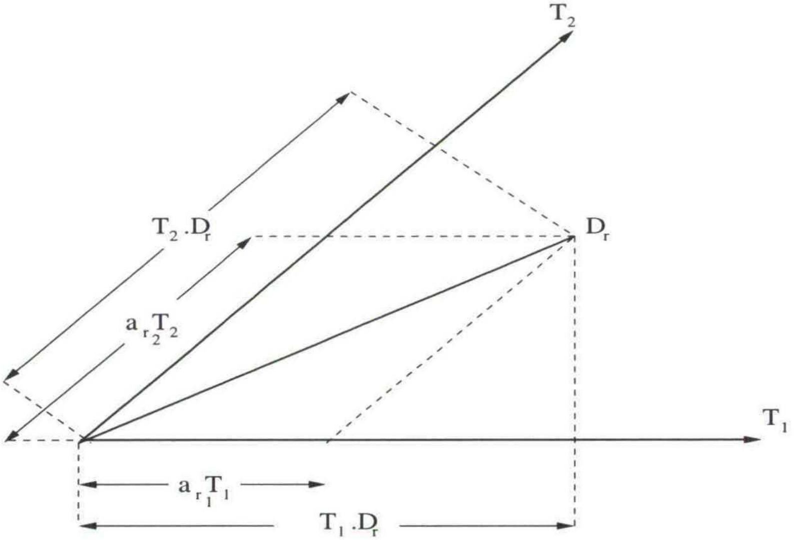


Figure 4.4: Non-orthogonal vectors.

Documents in the VSM are represented by vectors of keywords. The elements in the vectors may have any weight; this includes fractions, negative weights and weights greater than 1. The database can be represented as a term-document matrix with documents as rows of M distinct terms $t_1, t_2, ..., t_M$, the keywords as columns of N documents $d_1, d_2, ..., d_N$ and for every document–keyword combination a weight a_{td}. Hence the rth document D_r can be written as

$$D_r = \sum_{i=1}^{M} a_{ri} T_i$$

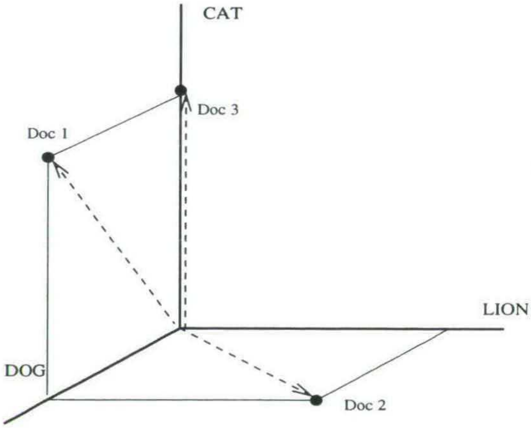
where every a_{ri} is a component of D_r along the vector T_i. Now the documents can be expressed as vectors in the keyword space, as illustrated by a two-dimensional example in Figure 4.4, where document D_r is drawn in a space that consists of the term vectors T_1 and T_2.

The difference with the document vector model is that the vectors in the vector space model are associated with to a set of similarity functions with the exclusion of other functions and methods. For instance, the probabilistic model described below is based on the same vector representations, but it applies different functions to rank the documents in relevant and not relevant.

In vector space the similarity between two vectors $\vec{a}$ and $\vec{b}$ can be computed by $a.b = \mid a \mid\mid b \mid cos\ \alpha$, where α is the angle between $\vec{a}$ and $\vec{b}$. Therefore the similarity between two documents in this representation can be computed as

$$D_r.Q_s = \sum_{i,j=1}^{M} w_{ri} q_{sj} T_i.T_j$$

but note that this formula depends not only on the contents of the document vectors themselves, but also on the term correlations $T_i.T_j$ for all term pairs. The contents of the document vectors can be generated by relatively simple indexing and weighting operations, but it is more difficult to generate dependable term associations. In practice therefore, it is assumed that the terms are uncorrelated, in which case the term vectors are orthogonal and $T_i.T_j = 0$, except when $i = j$, and $T_i.T_j = 1$. Referring again to Figure 4.4, we can see how $a_{r_1}T_1$ equals $T_1.D_r$ and $a_{r_2}T_2$ equals $T_2.D_r$ when T_1 and T_2 are orthogonal.



Doc	1	2	3
cat	1	0	1
dog	1	1	0
lion	0	1	0

Query: documents on cats.
Doc 3 is nearest, followed by Doc 1

Figure 4.5: Positioning vectors in keyword space

In Figure 4.5 we see an example that is situated in a three-dimensional keyword space, with the three orthogonal axes representing the keywords *cat*, *dog* and *lion*. Three documents are indexed as respectively $< cat, dog >$, $< dog, lion >$ and $< cat >$. The binary weight is used to plot the document vectors (dotted lines) in this three–dimensional space, and the angle between the vectors can be computed.

4.4.3 Similarity functions

The similarity functions may be selected from vector algebra functions. [Noreault et al., 1981] identifies sixty-seven of such functions, most of which come from IR literature. Expressed in terms of the similarity $sim(d_j, d_k)$ of two documents d_j and d_k, the measures that occur most in the literature are:

- The *matching coefficient*, which counts the number of dimensions on which both documents have a non-zero entry.

$$sim(d_j, d_k) = \mid d_j \cap d_k \mid$$

- The *scalar product* or *sum of products*, which is the general case of the matching coefficient for non-binary vectors.

$$sim(d_j, d_k) = \vec{d_j}.\vec{d_k}$$

- The *Dice coefficient*, which is the matching coefficient normalized for length by dividing the total number of non-zero entries.

$$sim(d_j, d_k) = 2\frac{\mid d_j \cap d_k \mid}{\mid d_j \mid + \mid d_k \mid}$$

- The *Jaccard coefficient*, which is the matching coefficient penalizing a small number of shared entries.

$$sim(d_j, d_k) = \frac{\mid d_j \cap d_k \mid}{\mid d_j \cup d_k \mid}$$

- The *Euclidean distance.*

$$sim(d_j, d_k) = \sqrt{\sum_{i=1}^{m} (d_{ji} - d_{ki})^2}$$

- The *binary cosine*, which is similar to the Dice coefficient for documents with the same number of non-zero entries, but penalizes less when the number of non-zero entries in both documents is very different.

$$sim(d_j, d_k) = \frac{\mid d_j \cap d_k \mid}{\mid d_j \mid^{\frac{1}{2}} . \mid d_k \mid^{\frac{1}{2}}}$$

- The *weighted cosine*, which is a generalization of the binary cosine for real-weighted vectors.

$$sim(d_j, d_k) = \frac{\sum_{i=1}^{m} d_{ji}.d_{ki}}{\sqrt{\sum_{i=1}^{m} d_{ji}^2 . \sum_{i=1}^{m} d_{ki}^2}}$$

An interesting property of the cosine is that when applied to normalized vectors, it will give the same ranking as Euclidean distance. A vector is normalized if $\sum w_i^2 = 1$.

4.5 The cluster model

The *cluster model* for IR is different from most other models mentioned here, because it is centered on *browsing* instead of on *querying*. The underlying assumption is the 'cluster hypothesis', that postulates that relevant documents are more similar to each other than to irrelevant documents, or in a slightly different form: "closely associated documents tend to be relevant to the same requests" [van Rijsbergen, 1979].

This assumption is exploited by organizing storage of the documents in such a way that similar documents are kept together in 'clusters' (see Figure 4.6). For every cluster a centroid is computed as a (possibly imaginary) 'average' document (see section 4.3.1). The centroids and supercentroids of every cluster are stored in a database. At retrieval time the query is first compared with the supercentroids and then with the centroids under the highest scoring supercentroid; the documents in the cluster or clusters with the highest retrieval weight are ranked and presented to the user. In this way, the individual document is not visible in the initial stages of the search, but this would be compensated by the improved efficiency of the search process.

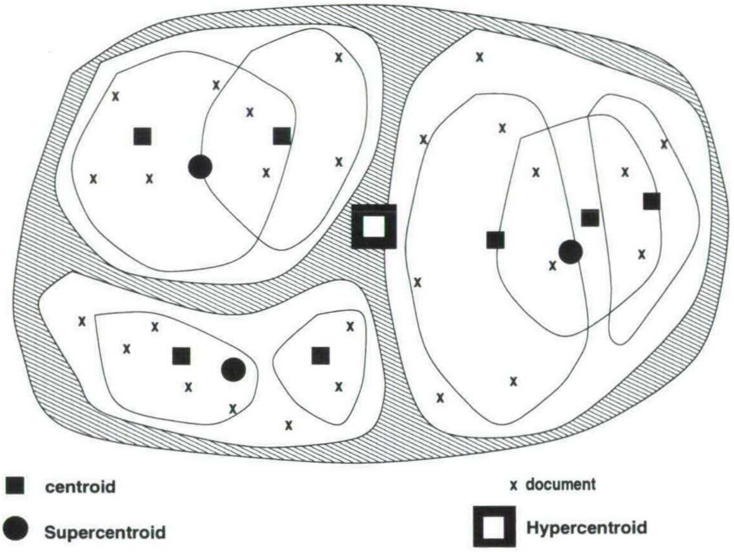


Figure 4.6: Clustered file organization

There are a number of alternative approaches to clustering. The most important division is that between *hierarchical clustering* and *heuristic* or *flat clustering*. In hierarchical clustering all documents or existing clusters are recursively compared to each other and the two most similar items are combined in a new cluster. Flat or non-hierarchical clustering starts out with a number of randomly assigned centroids and then tries to improve this initial partition by repeated passes of reallocating the documents to the currently best cluster, i.e. the centroid that is closest. After a full iteration, when all documents have been allocated, the centroid is recomputed from the mean of the documents that have been allocated to it.

Another difference in clustering technique may be whether documents are allowed to belong to one cluster only (*hard clustering*) or to two or more clusters at the same time (*soft clustering*). Finally the way in which clusters are compared is an issue: *single link* comparison, in which the most similar items from each group are compared, *complete link* comparison, in which the least similar pair of items is compared, and *group average* comparison, where the averages of each cluster are computed and compared.

The efficiency of document clustering has been emphasised by [Salton, 1968]: "Clearly in practice it is not possible to match each analysed document with each analysed search request because the time consumed by such operation would be excessive...." Salton believes that although document clustering saves time, it necessarily reduces the effectiveness of a retrieval system. On the other hand, [Jardine et al., 1971] and [van Rijsbergen, 1979] state that a case has been made showing that on the contrary document clustering has potential for improving the effectiveness.

Neither assumption has in fact been proven true. On the contrary, [Shaw et al., 1997a] proves convincingly that the cluster hypothesis, if not invalid, certainly does not live up to early expectations, as experiments show that the effectiveness of cluster-based retrieval is not significantly better than random selection. In chapter 3.2 we already described these experiments in more detail.

4.6 The probabilistic models

The literature on IR does not usually rank the probabilistic models under the relevance feedback model , and we reluctantly follow this convention in that we describe them in a separate section. However, as the probabilistic models in their pure manifestations need relevance information, we will consider them as being subordinate to the relevance feedback model.

The use of probability theory in the field of IR is almost as old as the use of computers in this discipline. Early descriptions [Maron, 1965] already are given in the mid-1960's. Ten years later the principles were worked out by, e.g. [Cooper and Maron:, 1978], [Robertson, 1977] and [Robertson and Jones, 1976]. They base their models on the premise that terms appearing in previously retrieved relevant documents for a given query should be given a higher weight than if they had not appeared in those relevant documents.

As an example, in this section we will describe a model recently published in [Hiemstra and Kraaij, 1999], and described as a "linguistically motivated probabilistic model of information retrieval". In this model, documents and queries are described by compound events, which are events that consist of two or more single events. In this case, the single events that denote the compound event are the index terms in the collection (as with the other models described here, this model also assumes that the index terms in a document are independent of each other). The probability of the compound event can therefore be calculated by multiplying the probabilities of the single events as in the following equation:

$$P(T_1, T_2, ..., T_n \mid D) = \prod_{i=1}^{n} P(T_i \mid D)$$

Each document contains a small sample of natural language for which the retrieval system should build a statistical language model $P(T \mid D)$, where T is a single event. If the user enters a query $T_1, T_2, ..., T_n$ the system uses this equation to calculate the probability of that query given each possible value of the document D. The most straightforward way to estimate the probabilities $P(T \mid D)$ would be maximum likelihood estimation (MLE), which maximizes the probability of observed events and assigns zero probability to unseen events. However, this would mean that every document that does not contain all of the query terms, would get a likelihood of zero. A solution would be estimation by linear interpolation, introducing knowledge of the distribution of the term over the complete database:

$$P_{li}(T \mid D) = \alpha P_{mle}(T) + (1-\alpha) P_{mle}(T \mid D),\ 0 < \alpha < 1$$

In this equation global information $P(T = t)$ on the term t, or the a priori probability of term T, is mixed with the local information $P(T = t \mid D)$, that is, the probability that term T occurs in document D. The ratio of global and local information is determined by the value of α. It is standard practice in IR to use the document frequency for global information and the term frequency for local information, a practice on which the $tf.idf$-familiy of word weights is based. The next equation specifies the way how probabilities are estimated from document frequencies and term frequencies.

$$P(T_i = t_i \mid D = d) = \alpha \frac{df(t_i)}{\sum_t df(t)} + (1-\alpha) \frac{tf(t_i, d)}{\sum_t tf(t, d)}$$

where df_t is the document frequency and tf_{td} is the term frequency. We can extend this formula for use in a vector space environment by rewriting the formula for use with a document vector of n keywords as follows:

$$P(T_1 = t_1, ..., T_n = t_n \mid D = d) = \prod_{i=1}^{n} (\alpha \frac{df(t_i)}{\sum_t df(t)} + (1-\alpha) \frac{tf(t_i, d)}{\sum_t tf(t, d)})$$

This equation follows directly from the earlier formulas. Any monotonic transformation of the ranking formula will also produce the same ranking of documents. Instead of the product of weights we could therefore rank the documents by the sum of logarithmic weights. Now this ranking formula can be rewritten as

$$P(T_1 = t_1, ..., T_n = t_n \mid D = d) = \sum_{i=1}^{n} log(1 + \frac{tf(t_i, d)}{df(t) \sum_t tf(t, d)} \frac{(1 - \alpha) \sum_t df(t)}{\alpha})$$

This formula can be applied directly to compute term weights from the frequency information and, according to [Hiemstra and Kraaij, 1999] gives a result, that is equivalent to $tf.idf$ weights with document length normalization.

4.7 The relevance feedback model

In this section we will discuss some techniques that for their effectiveness depend on relevance information. As we noted before, the probabilistic models should also be mentioned here, but we will restrict ourselves to a single example of a more specific implementation of probabilistic relevance feedback, the Rocchio algorithm, and to an example of how genetic algorithms may be applied to relevance feedback.

The relevance feedback model is based on the assumption that documents that already are selected as satisfactory may be used to adjust the original query to retrieve even more satisfactory documents. The best known approach in IR is the *Rocchio learning algorithm* [Rocchio, 1971], that takes an initial vector of features and presents this vector as a query to an IR system. The document vectors that are returned are divided in two classes, relevant and non-relevant, and the formula then is applied to adjust the values of the query vector towards the values in the relevant set and away from the non-relevant set, according to the formula:

$$w = \alpha w_1 + \beta \frac{\sum_{i \in R} x_i}{n_R} - \gamma \frac{\sum_{i \ni R} x_i}{n - n_R}$$

where α, β and γ are adjustment parameters, w_1 is the original weight of a keyword in the query vector; x_i the weights of that keyword in the relevant, c. q. non-relevant documents; n is the number of documents, and n_R the number of relevant documents in the returned set.

The Rocchio algorithm is an obvious example of relevance feedback, and is often used as a baseline to be compared with other relevance feedback techniques.

A related, but slightly different technique, that is based on the probabilistic models, is presented by [Crestani, 1994]. The improvement on the original query consists of adding more terms to those already present in the query. This is done as follows. First the user selects a number of the documents in the retrieved set as relevant. Then the terms w_i in those documents are scored according to the following formula:

$$w_i = \log \frac{r_i(N - n_i - R + r_i)}{(R - r_i)(n_i - r_i)}$$

where N is the number of documents in the collection, n_i is the number of documents with an occurrence of term i, R is the number of relevant documents in the retrieved set, and r_i is the number of relevant documents pointed out by the user with an occurrence of term i. Essentially, what this function does is compare the frequency of occurrence of a term in the relevant documents with the frequency of occurrence of that term in the whole document collection. So if a term occurs much more frequently in the documents marked as relevant than in the whole document collection, it will be assigned a high weight.

4.7.1 Genetic algorithms

Another relatively new and exciting approach of problem solving is found in genetic algorithms. Genetic algorithms may best be explained using the metaphor of Darwinistic evolution. Every individual in a population has a set of properties. His chances of survival (and procreation) are dependent on these properties: if they are not suitable for the environment he lives in, he will die before he has had the chance of passing those properties on to the next generation. Sexual reproduction sees to it that new individuals are born who have a combination of the properties of both parents and some of these individuals may therefore have a better chance of surviving and passing this combination of properties on to the next generation.

Apart from this reshuffling of the genes there is a second mechanism to ensure that an individu acquires new properties: mutation. Mutation is an infrequent, possibly random change in the genes, caused by external factors, e.g. radiation. In this way properties may come into being that cannot be traced back to one of the parents.

Genetic algorithms imitate the above mechanisms in problem solving. A genetic program typically formulates a series of possible solutions, compares them to a goal to be attained, throws out the bad solutions and reconstructs new potential solutions from parts of the better solutions.

A series of solutions may get stuck on a so-called local maximum. This is where mutation comes in. Every hundred or thousand or whatever number of iterations a random bit in one of the parents is flipped. If that happens to drastically improve the survival chances of that individual, the trait will be passed to its offspring. If not, it will just disappear within a few generations.

How may genetic algorithms improve the quality of IR systems? An example is given by Chen [Chen, 1995]. If a query results in a number of documents, the user can indicate which documents in the result set best fit his information need; let us call those documents the 'succesful' documents. Now it is possible to use the document representations of these documents as a new query. This is what relevance feedback is about: the keywords in the succesful documents are the best keywords to search new documents. But there may also be some keywords in the succesful documents that are in themselves unwanted, as they cause unwanted documents to be retrieved. With the help of a genetic algorithm the keywords that are the best representatives of the initial succesful documents can be identified and used for continued retrieval with those keywords only.

To set up the algorithm, we first translate the documents to bitstrings. For every keyword in the succesful documents we reserve a position in the bitstring:

	cat	dog	lion	horse	fish	art	wine
doc.1	0	1	0	1	0	0	0
doc.2	0	0	0	1	0	0	0
doc.3	0	0	1	1	0	1	1
doc.4	1	1	0	1	1	0	0

and the question is which combination of 'cat', 'dog', 'lion', ... best represents the relevant documents. We can use e.g. Jaccards coefficient to compute the $N(N-1)$ similarity between the N documents; and the 'fitness' F of each document D_n becomes the sum of the similarities it has with all other documents.

A new generation is now produced in which the 'fitter' document representations have a better chance to be included than the 'unfit' documents. Offspring is generated by crossing the selected document representation at random positions, and this cycle continues until only a single representation is left over in the optimized population. This document representation is used as the next query.

If a user provides a set of initial documents that are already closely related, the overall fitness of the documents is high and does not improve much further. If on the other hand the user-selected documents are only loosely related and the overall fitness therefore is low, genetic algorithms do a good job of optimizing the representations.

4.8 Conclusions

In this chapter we have presented examples of all important models and techniques in IR that are based on vector representations of the document and we used the phrase 'document vector model' to refer to these models. The time has come to explain in some detail why we prefer this term.

In chapter 1 we already argued that traditional models explain only certain aspects of any IR-system. We also noted that the implementation of other parts of an IR system may influence the performance of that system just as much as the part that gave the model its name. We have also observed that the dividing lines between models often are not very clear: are the probabilistic models subordinated to the relevance feedback model or not? And why is it that the various strategies to decide on the weight of a word in the database, or in a particular document never found their way in one of the extant models? Frequency-based techniques, such as *tf.idf*, are taken for granted in the vector space model, but there is no reason why they should not be applicable outside that model.

The most important difference between the Boolean model and the other models, according to the literature, is that the Boolean model divides the set of documents into two classes, whereas the other models produce a ranking. But as we have seen above, the fuzzy Boolean model also returns a ranking, and even straightforward Boolean systems may offer various ranking algorithms or even relevance feedback strategies. Also, there are situations in which a system that is built according to the vector space model acts like a Boolean system in that only a weak ordening of documents in relevant and not-relevant is returned.

Alltogether, this is certainly not a desirable state of affairs for a classification of IR systems based on models.

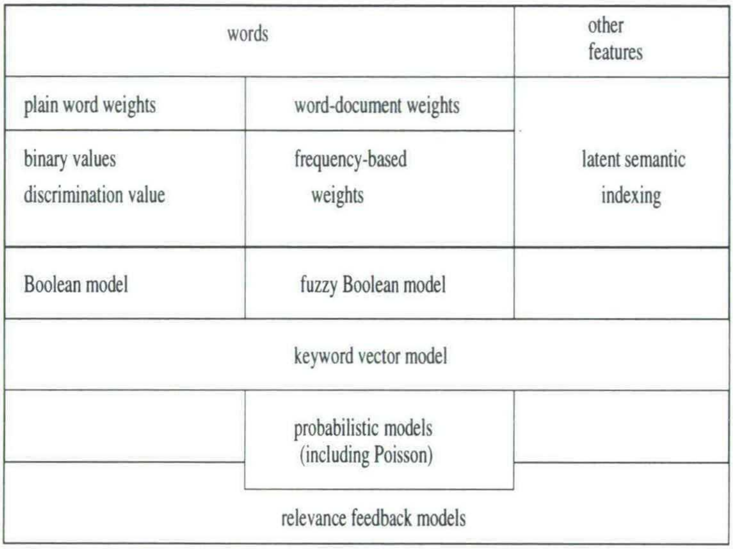


Figure 4.7: The document vector model

We want to stress the point that an implementation of an IR-system should not and cannot conform to a single model. It always is a combination of models, and ignoring this basic fact will cause confusion when comparisons between systems have to be made. At least the following points should be made clear:

1. The way in which the part of the document that is presented to the system, the *document surrogate*, is arrived at. Although this point is not really an issue when we talk about taxonomy and models, it is an extremely important issue in actual systems (see our remarks in section 2.2).

2. The metric by which the features that find their way into the system are selected and in what form they are stored. For instance, the final document-keyword matrix may have started out as a plain frequency

matrix of all word types in the documents. This matrix may have been the basis for the computation of discrimination values that themselves have been used to delete words below a certain treshold. Finally, the words that were left may have been stored in the document presentation as binary weights.

3. The similarity function itself. We have presented a few variants of these functions earlier in this chapter (for an exhausting survey of weighting and similarity functions, see [Noreault et al., 1981]), but it is evident that the performance of the individual functions depends strongly on the values that are stored in the document vectors.

4. The structure that governs the subsequent processing of the results of the similarity functions (e.g. *relevance feedback*).

The basis for all these considerations is the document vector. This is not necessarily a vector of keywords, as is shown by the fact that e.g. latent semantic indexing operates with features that are computed from singular value decomposition of the document-keyword matrix. But as we have seen before, what distinguishes an IR-system is the fact that the identity of the *document* is preserved. It is the document vector in which the results of the weighting operations are collected; the document vector is the target of the similarity functions, and finally, the relevance feedback operations are all aimed at manipulation of the document vector.

In Figure 4.7 we have presented the important models in IR *vis-a-vis* our document vector model. In the upper half we see how the document vector model can be divided into two main groups: word-based vectors (which include word stems, multi-word phrases and even n-grams) and vectors that contain features that do not correspond directly to tokens in the text. As an example of this last type we give the features that are produced by latent semantic indexing. The word-based vectors can be divided into two groups: those using plain word weights, inluding binary values and discrimination values, and those using word-document weights, such as the frequency-based weights.

Under the thick line the four main models are drawn. The width of the box corresponds to the document vector types that they can be used with. Boolean models correspond to the two word groups; the keyword vector model can be applied to all kinds of document vectors, as is the case with relevance feedback. The probabilistic models apply only to frequency-based vectors, but they overlap partially with the relevance feedback model. Using this model, the supremacy of the document vector over all other considerations seems clear.

Chapter 5

Comparing IR-systems: CLARIT and TOPIC [1]

Abstract: *The* TOPIC *and* CLARIT *information retrieval systems are discussed in terms of assigned vs. derived and precoordinated vs. postcoordinated indexing. The document representations of both systems are compared. A test was done using a sample of the Wall Street Journal corpus. The positive results for* CLARIT *in earlier tests were not observed in this general database.*

5.1 Introduction

The discipline of information retrieval is perhaps as old as the written word and with the advent of big libraries much effort was put in systems that enable people that are in need of information to retrieve that information (or rather those documents that contain that information) from those collections. Nevertheless the tone of the IR-community is not optimistic, when it talks about the performance of those systems. The observations of Cleverdon on the subject of human indexing [Cleverdon, 1984] and the famous Blair/Maron experiment in full text retrieval [Blair and Maron, 1985]), still loom darkly over all attempts to substantially alter the effectivity of information retrieval techniques. The probabilistic and vector models, that have been perfected in the last ten years, however sophisticated, cannot claim to imply real understanding of the documents and the AI-approaches are hampered by the fact that the creation and maintaining of involved knowledge representations only works in very small domains. And as Cleverdon observed, human indexing just is not consistent enough to guarantee acceptable recall and precision over sizable databases.

But the ever increasing number of articles, documents and books in libraries and elsewhere will not disappear just because there is no unified paradigm in which to solve the retrieval problem. Therefore widely different schemes and systems have evolved to solve that problem and this fact has placed an

[1] This chapter appeared under the same title in the Journal of the American Society for Information Science, vol.44 no.7, August 1993. p. 383-392 ([Paijmans, 1993]). The bibliography entries have been collected in the bibliography at the end of this book.

increasing importance on strategies for testing such schemes and systems.

In this article we will apply some testing methods that were used and published by [White and Griffith, 1987], to documents that were indexed by the TOPIC and CLARIT systems. Thus we hope to compare their performance relative to each other and gain some insights in certain problematic areas of IR in general.

5.2 Background

Although there is no general accepted taxonomy for IR-systems, it is possible to recognise several models. A model that is widely accepted and that is used in one way or another in almost every IR-system is the Boolean model. In this model, set-theory is used to handle lists of descriptors to create sets of documents that are relevant to a query. Another model that touches on our theme is the thesaural model, that tries to organise such descriptors in a semantic network. Other models define how to select descriptors from the document (e.g., the probabilistic model), or how to compare documents and queries to decide on their similarity (the vector model). Literature abounds with systems that experiment with one model or another, but when we look which of these models have been implemented in real life systems, the number is disappointingly small. Most systems adhere firmly to the Boolean model, using an exhaustive inverted list of words as document representation (for a description of this and other concepts see below). An early and succesfull system to do this was STAIRS [IBM, 1976], which was followed both on big computers and on PC's by a multitude of epigones (e.g., Freebase or Zyindex), that really only differed in user interface.

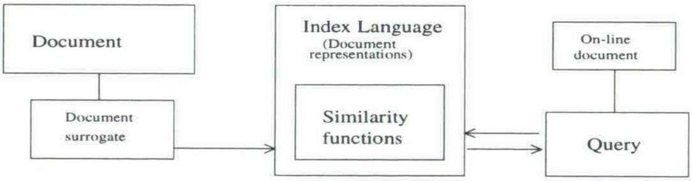


Figure 5.1: The classic model of information retrieval

If we look at models of retrieval systems, we notice a few parts that recur in almost every system. These parts are concisely arranged in the general IR model as put forward by [Salton and McGill, 1983], see Figure 5.1. Central in this model stands the index language (IL), into which both document and query are translated and which acts as a common reference to decide on the similarities between them and how to rank them in one order or another. An important part of this IL is a vocabulary of keywords or descriptors to describe the documents in the system, although attempts have been made to create more involved knowledge representations, such as frames.

In most IR systems documents have a place that is analogous to the record in normal database systems: i.e., a repository of data relating to a single item and we will often use the word record to mean document. This record also may act as a node to which other observations about the document are attached and in the process the document may disappear and be replaced by an abstract and bibliographic reference. Speaking about full text systems,we will use record and document as more-or-less synonymous. The parts of a document (possibly the complete text including mark-ups and other visual information) that are presented to the IR-system, we will call the document surrogate. The document representation (for short docrep) is that part of the document surrogate (or, if not a part, at least a description of it) that finds its way in the index language, thus representing the document in the system. We will see below, when we describe TOPIC, that this document representation, among other things, may well contain the complete document. The on-line document then is the document that is reported back to the user. In most cases this is a bibliographic reference or the reference and an abstract, but advanced systems may display the text of the document. Broadly speaking, the more advanced the system is, the more all three, the document surrogate, the document representation and the on-line document, will approach the original document.

Referring to the drawing in Figure 5.1, we can see that the three main concerns of IR are:

- translation of document and query into terms of the indexing language, including
- the modelling of the document representation, and
- supporting the user in the interaction with the index language and the knowledge contained in the document representations.

The document representation has a central role in this model, because at query time all questions will have to be matched against these docreps. Our concern in this study will be the quality of the docreps created by the two systems under consideration, given a document surrogate that contains the complete text (but without information about italics, different fonts, lay-out etc.). In particular we will analyse the document representations of the CLARIT and TOPIC systems.

5.2.1 Derived and assigned indexing

There are essentially two approaches to the creation and maintenance of this document or knowledge representation. One is to create a knowledge system in advance and assign the documents to it afterwards: assigned indexing. The other is to derive the terms of the index language from the documents themselves: derived indexing. The manual library systems, in which books were classified according to an existing classification system, e.g., Dewey or UDC, are assigned indexing systems; computerized IR-systems, that extract keywords from the documents according to some weighting scheme or another, are typical for derived indexing systems. We will extend the definition of assigned indexing systems to contain all systems that use terms in their docreps, that are not taken from the documents themselves, because such external terms belong to a knowledge representation outside the document.

The derived indexing systems became very popular when the computer made it easy to create an inverted list of all occurring words in a document base. In the seventies and eighties much effort was directed to techniques how to identify such words (phrases, sentences) in the inverted lists as were most efficient in retrieving particular documents. (for a discussion of both derived vs. assigned and precoordinate vs. postcoordinate systems, see [Foskett, 1982]).

5.2.2 Pre- and postcoordination

Many concepts that rightfully belong in the indexing language, may only be expressed using more than one word, e.g., 'computer programming' or 'aluminum welding'. In the older, manual indexing systems such compound phrases generally were recognized, kept together and entered in the system as complete terms. Such systems are called precoordinate systems. As a direct consequence of the inverted list and the ease with which sets of records could be handled using such lists, the emphasis came to lie on postcoordinative indexing, i.e., potential multi-word descriptors were separated in their components (or even word stems) and only these components were stored in the inverted list. At query time, Boolean operators such as AND, OR etc. were used to try and reconstruct such terms. Not unexpectedly much information was lost in the process and precision suffered. Therefore the specificity of the older, precoordinative systems sometimes was better than that of the newer, keyword-driven systems. Indeed we will see how Salton and Cleverdon found that such multiword phrase indexes became overspecific and that performance dropped as compared with single-word indexes.

TOPIC and CLARIT, the two systems described here, may be considered as each belonging to one of the derived and assigned indexing systems. Also, at least CLARIT tries to combine the efficiency of the inverted file while recovering the specificity of the precoordinative systems: it was built to detect significant phrases (noun phrases). TOPIC, that superimposes a semantic hierarchy on the inverted file system, implements a kind of higher order thesaurus and in such a thesaurus the compound terms that are typical for precoordinate systems may also be defined. We will argue that TOPIC (when the TOPICs are used) essentially is an assigned indexing system, although it maintains a complete inverted file of all words in all documents. The reasons for this will be explained below. That CLARIT is a derived indexing system is easier to see, as the terms that make up the document representation are extracted from the original document and no other terminology is added.

If we test the two system side by side, this is not because we want to compare the two systems in a kind of consumer's test. To do this would need a better understanding of both the systems and the prospective users. Also, the data used for the tests would need to be selected in a different way. We just want to do a few explorative experiments in order to gain a better understanding of the way the philopsophy of the two systems as it is displayed in the difference between the document representations, might influence performance.

Following the methodology of White and Griffith we will try to establish the performance of CLARIT and TOPIC relative to each other on three points:

1. The ability of both systems to link related documents.
2. The ability to discriminate between these linked subsets.
3. The ability to discriminate finely between individual documents.

5.3 The systems

We have chosen the CLARIT and TOPIC systems for our tests, because the TOPIC system was selected by the new library of Tilburg University as database system for the management of the On-line Contents database: a so-called current awareness service in which articles in journals are made accessible for retrieval immediately after appearance [Roes, 1992]. The system was installed on DEC equipment. This company also has supported the development of the CLARIT system through DEC's External Research Programme.

5.3.1 TOPIC or RUBRIC

The first system, TOPIC by Verity Inc. [anon., 1990], is the commercial offshoot of the rather well publicized experimental RUBRIC system [McCune et al., 1985]. Although TOPIC is a complete system, with indexing modules, retrieval engine and a user interface for interactive querying, we will limit ourselves to the document representations and related issues.

TOPIC approaches the problem of document retrieval in two stages. In the first stage a complete inverted file is created of all occurring strings in the document. Positional information about paragraphs or particular segments of the documents, is preserved in this inverted file. Together with Boolean and proximity operators and the ability to recognise fields in the document, this puts TOPIC alongside systems such as STAIRS, that also enable Boolean retrieval on strings in full-text documents. The docrep that consists of the set of words occurring in the document, and that is stored in the inverted file, acts as a primary access mechanism. At retrieval time the original document is consulted to obtain information on the proximity of the words. Thus it might be said that the document representation so far consists of terms from the complete text of the documents, which makes it a typical derived index.

However, the second stage that is grafted on top of this retrieval engine, is a knowledge representation tool, that may be thought of as essentially a weighted thesaurus, but that is implemented as a rulebase. Concepts are arranged in trees, or rather acyclic graphs, in which the strings that should occur in the documents are the leaves. The occurrence of such strings, using Boolean and/or proximity operators, is taken as weighted proof for the relevance of the higher concept and such concepts in their turn support other concepts (see Table 5.1). In a typical TOPIC application many of such concepts (topics) will be built in advance by an information specialist, thus effectively adding a knowledge base to the system. Subsequently the user can build and add topics of his own and those topics may or may not be accessible to other users.

This second layer may be considered part of the document representation, too. Indeed, if a topic is built and documents are recognized by the rules in that topic, these documents are added to a list with postings for that topic. Thus it is relatively easy to extract the sets of topics that may be said to belong to a document (i.e. score above a threshold for that document) and we have done this in order to conduct the experiments. We consider such topics also to be a document representation, one that really is separate from the inverted file and complete document mentioned above.

One might ask what would be the difference between the topics of TOPIC and the entries of an orthodox classification system or a thesaurus. The answer is that the topics here ultimately are defined as properties of documents in stead of in semantic terms. This gives the system a great flexibility, but also ample opportunity for snap decisions, ad hoc constructs and heuristics that may work fine in small collections, but break down when applied to big databases. A possible reason for this is that in big databases different subpopulations of documents will come in existence, that all cover more or less an identical subject, but approach it from widely different angles and (therefore) will use different vocabularies. There is a distinct danger that the assessment of the weights in such cases will become progessively more subjective, whereas attempts to introduce objective methods (e.g., statistics) will in fact cause a return to the frequency-based models.

An extensive summing up of the shortcomings of TOPIC is given in [Inc., 1990], 1990, but it should be noted that this report was written by an unsuccessful competitor for the library system of Tilburg University.

5.3.2 CLARIT

The CLARIT system works on totally different principles. The most important component of the system is a dictionary builder, that tries to extract the most informative phrases from natural language texts [Evans et al., 1991], although more components, such as a retrieval interface are being added. The most interesting part is the said dictionary builder, that creates a list of NPs (noun phrases). The designers of the system call such a dictionary a first order thesaurus but the point should be stressed that no semantic relations are defined between the individual terms.

```
GENERAL-MOTORS
* 1.00 GM-COMPANIES
    ** 0.50 GENERAL-MOTORS-ACCEPTA-PHRS
        *** "general"
        *** "motors"
        *** "acceptance"
        *** "corp"
    ** 0.50 "gmac"
    ** 0.50 "hughes aircraft co"
* 0.50 GM-PEOPLE
    ** 1.00 GM-EX-CEO
        *** 1.00 "roger smith"
    ** 1.00 GM-PRES
          *** 1.00 LLOYD-REUSS
            **** "lloyd"
            **** "reuss"
    ** 1.00 GM-CEO
        *** 1.00 ROBERT-STEMPEL
* 0.50 GM-PRODUCTS
    ** 0.50 "pontiac"
    ** 0.50 "oldsmobile"
    ** 0.50 "buick"
. . . . .
```

Table 5.1: A topic from TOPIC system (indent added)

CLARIT's approach to indexing a document is as follows: the first activity consists of readying the document for processing by normalising the character-set etc. The document is then parsed by a very robust parser, that identifies the noun phrases in the document and extracts them to a list of candidate NP's.

These NP's are scored by applying various frequency-based formulas to the individual words, in the course of which they are also compared with the characteristics of a domain corpus and a general English corpus. Word co-occurrence statistics are not considered.

Finally the candidate terms are matched with one or more lists of certified terms (a general English corpus and, if applicable, a domain corpus), thus creating three groups: exact, general and novel terms, based on the fact whether an exact match is found, the candidate NP consists of a constituent or sub-term of a certified term, or that the NP is a new term.

The end product of a CLARIT-indexing run is a document representation that consists of a weighted list of such terms, sorted out in exact, general and novel terms (Table 5.2). Tests run on rather small document collections as publicised by Evans and others show exceptionally good performance as compared to human indexers [Evans et al., 1991]. However, these tests are limited to the comparison of individual documents and not, as we intend to do, taken over sets of documents.

```
# 140
\=
    2.322273 (nasdaq) () 0
    1.161136 (est) () 0
    0.387045 (toronto) () 0
    0.379069 (wholly owned subsidiary) () 0
    ........
\=
\>
    5.392150 (petroleum) () 0
    5.392150 (prnewswire) () 0
    0.928909 (asset) () 0
    0.928909 (subsidiary) () 0
    .........
\>
\?
    2.818624 (prnewswire giant) () 0
    2.696075 (prnewswire) () 0
    0.122549 (giant) () 0
    1.985585 (pacific petroleum incorporated) () 0
    ........
\?
```

Table 5.2: Result of CLARIT indexing

5.4 First discussion

As we already indicated, CLARIT is a typical derived indexing system in that the document representation is derived from the original document. One might argue that the knowledge in the parser is external knowledge, but this certainly is not the kind of knowledge into which we may map the documents or concepts occurring in the documents. The same is true for the various frequency-based formulas that are used by CLARIT. Such formulas capture the intuitive notion that the frequency with which a word occurs, is in some way meaningfull for its informative value, but they do not contain the semantic or pragmatic knowledge that is commonly associated with meaning and understanding. For instance: the sentence

```
Dog, dog, dog, dog, dog, dog!
```

might well land this same document in the lap of somebody, who is looking for literature on dogs, because the word dog now possibly scores above some such statistic threshold. But this certainly does not mean that this document is about dogs! On the other hand the certified list or lists against which the candidate phrases are matched, certainly is a knowledge representation, albeit a very shallow one.

The fact that terms exist that are composed of several words, thus improving precision, gives CLARIT a decidedly precoordinate flavour, although of course such terms also may be combined again at query time to create new concepts to search for.

Things are different if we consider TOPIC. The first docrep, which is used to access the document, is the set of all occurring word strings in the document, stored in the inverted file. Moreover, the user may restrict his use of TOPIC to those strings in various combinations with Boolean and proximity operators and in that case TOPIC does not essentially differ from older systems like STAIRS. But the creation of TOPICs as described above, means that a knowledge system comes into existence that is independent from the document representations. If a document 'fits' the rules for a certain TOPIC, it may be considered as assigned to that TOPIC, rather than that the TOPIC is derived from the document.

It might be argued that humans index a document in very much the same way: by forming hypotheses about the topicality of the document and by testing those hypotheses by checking for the existence of other concepts and strings in the document. TOPIC really does the same thing. Also, as individual words and concepts are used to construct bigger concepts and these concepts are available at query time, TOPIC, too, resembles the older pre-coordinate systems. Contrary to CLARIT, these concepts are assigned to the documents, not derived from them.

This makes the philosophy behind both systems very dissimilar. TOPIC departs from concepts that are created by the user more or less independently from the database and which reflect his interests, but not necessarily the contents of the database. The system then tries to find evidence in terms of strings and combinations that the concept may be found in a document. Such knowledge is incremental and some claim that it may soon lead to conflicting TOPICs (see the arguments put forward in the Basisplus report [Inc., 1990]). Also, there naturally is no support for TOPICs and concepts, that never have been declared. This may result in poor performance, when a user approaches the system with a new need.

CLARIT, on the other hand, tries to identify such parts of the document as give a good indication what the document is about and creates an index of keywords and key phrases. This supposedly makes for a very regular performance as all documents are treated equal; but as we will see this is not necessarily the case. The statistical nature of the frequency-based formulas applied by CLARIT carries an inherent danger of important terms narrowly missing some threshold.

So both system try to escape from the flat knowledge representation that is stored in the docreps of the inverted file of keywords type: CLARIT by extending the keywords to (weighted) key phrases, TOPIC by adding a user-supplied semantic hierarchy to the docreps. What the systems are actually attempting, is to try and improve on the older, single keyword systems in the face of the evidence brought forward by Salton and Cleverdon, as we will see presently.

That the performance of 'normal' free text databases of the STAIRS type (with a complete inverted file of single keywords) was insufficient, was demonstrated by the Blair/Maron study. On the other hand, already in 1983 it was stated by Salton and McGill that '...the simple uncontrolled indexing language produce the best retrieval performance, while the controlled vocabulary and phrases (simple concepts) furnished increasingly worse results.' (Salton and McGill, 1983, p.102). They came to this conclusions after extensive testing of both automatic indexing (Salton and SMART) and manual indexing (The CRANFIELD experiments by Cleverdon). To quote the last author: '...as narrower, broader or related terms are brought in ...performance decreases ...The simple concept (phrase) index languages were overspecific.' (in:Salton and McGill, 1983, p.102).

As Salton observed, these results are rather counterintuitive. It seemed that the adding of knowledge to the document representation actually lowered the performance of the whole system. Of course the SMART and CRANFIELD tests were designed to operate in a large and heterogeneous user population and the addition of thesaurus-like tools and multi-word concepts probably created an environment that was too specific to accommodate such a population. Still, those are not auspicious words under which to start the exploration of two systems that are either based on a thesaurus-like structure with broader and narrower-term relations, like TOPIC, or in the case of CLARIT, on the assumption that multi-word phrases are better index terms than single words.

The approach taken by TOPIC seems to counter the problems by placing the onus of building a thesaurus on the individual user, thus enabling him to concentrate on the relations he deems necessary and to ignore other relations. The designers of CLARIT concentrate on the possibilities offered by newer parsing algorithms to extract such phrases as are most descriptive for the document. It is interesting to note that the conclusions of the CRANFIELD experiments of Cleverdon were based on manual indexing: so unless CLARIT outperforms humans on detecting the salient terms in a document, the observations of Cleverdon on the relatively bad performance of such phrase indexes seem to remain valid.

5.5 Methodology

Of course there is the question whether it is legitimate to compare two systems that differ so widely. But as they both claim to offer very similar services (the retrieval of full-text documents from a document base), there should be no reason why they should not be compared. The question is, which methods should be used. We wanted to test how the two systems, or rather their docreps, would perform on a document base of free text documents on general subjects. In this section we will outline the methods used, taking the work of [White and Griffith, 1987] as an example. While working on the TOPIC-system, we noticed a peculiarity in the demo database, which raised some new questions. We will cover these questions separately.

```
Company      : Giant Pacific Petroleum Inc. (GPPXF
Industry     : Oil Drilling; Oilfield Equip \& Service
Subject      : Acquisitions, Mergers, Takeover
Market Sector: Energ
Region       : Canada
```

Table 5.3: Original classification of document 140

5.5.1 Selection and preparation of the databases

White and Griffiths used existing databases, each containing several millions of bibliographic records with attached descriptors. We had no such databases available for CLARIT and the TOPIC database of the library of Tilburg University contained no documents of the kind we wanted to use. Therefore we decided to use the demonstration database of 200 documents (500 Kb) that comes with TOPIC (eventually 128 documents were used). The database contained articles from the Wall Street Journal. Thanks to the co-operation of Drs T. van den Aker of the computer centre of our University we were able to obtain a list of document-titles with the TOPICs that were considered by the TOPIC-system to be relevant for each title. The TOPICs themselves were also part of the demo database.

Initially we wanted to use the documents exactly as they occurred in the demonstration database. Then we found that to each of the original documents entries from a classification were attached (Table 5.3) and that this classification was also searched by the TOPIC system. We thought this a rather unfair tactic to use in a demonstration database and it made the records in their original form unsuitable for testing purposes so we decided to delete these classifications from the documents and to create a new TOPIC-database. The rather interesting question if and how much the classifications in the original database influenced the performance of the TOPIC system will not be addressed here.

It may be argued that using TOPIC's document base gives this system an unfair advantage over CLARIT. We don't see why, especially not after the excision of the classifications. Of course we could have chosen a different collection of documents, but then we would have had to construct a new hierarchy of TOPICs and we wanted to avoid the criticism to have worked with a substandard set of TOPICs.

We created for every record in the test collection a document representation to be used in qualitative comparisons, both for the CLARIT and for the TOPIC database (Table 5.4). Although both docreps consisted of weighted terms, we decided to ignore the weights in the quantitative considerations and disregarded the differences between the three term-groups of CLARIT. Also the very long CLARIT-lists generated for each document were truncated at a weight below 1.0000 or after thirty-five terms, whichever came first. Using this threshold, we found a total of appr. 3400 postings of 2000 different terms in the 128 records of the CLARIT-set: for TOPIC we noted 1215 postings of 103

```
140 Acquisition of Patriot Energy Company Ltd. Announced

\acr{TOPIC}              evidence  weight    \acr{CLARIT}
-------------------------------------------------------------
TRADE-ACTION        0.55       5.392150  prnewswire
MERGER-ACTIVITY     0.83       5.392150  petroleum
MERGER-ACQUISITION 0.86       2.818624  prnewswire giant
FINANCIAL-TOPICS    0.43       2.696075  prnewswire
COMPUTER-PRODUCTS  0.80       2.322273  nasdaq
IBM-PRODUCTS        0.80       1.985585  pacific petroleum inc
IBM                 0.48       1.861511  announced calgary
COMPUTER-COMPANIES 0.48       1.840358  calgary
                               1.797383  petroleum
                               1.283685  giant pacific
                               1.254242  vancouver stock exchange
                               1.226905  vancouver
                               1.161136  pacific
                               1.161136  est
. . . . .
```

Table 5.4: Final document representations in TOPIC and CLARIT

TOPIC-terms.

It should be noted that the TOPICs that constituted our TOPIC-docrep, are all non-terminal nodes of trees, whose leaves are literal strings. Every posting in the docrep signifies the occurrence of at least one lower TOPIC or string and although both the higher and the lower TOPIC are in our docrep, the string(s) that caused the firing of the TOPICs, are not included. We extracted those strings separately (see below, virtual docreps) and found a total of 1351 postings of 205 distinct literals.

The next step was the creation of subsets of related documents in a manner that was independent of the indexing processes to be tested. The method followed by White and Griffiths, who used co-citing and co-referencing, could not be used directly for this database because of the nature of the documents. However, the same classification that we had to cut away from the original documents, suggested itself as an independent system (it was not created by TOPIC, we only omitted it from the tests because it was an enrichment of the original document). Thus we created a dictionary of the terms in the industry-, subject and market-fields of that classification and asked two independent persons (not professional indexers, but knowledgeable in the general field of the documents) to check documents and dictionary for consistency. Subsequently this classification was judged too unspecific to identify really interesting subsets. Then we asked them to identify six groups on the general subjects of networks, mergers, war/violence, software, legal matters and drugs/pharmaceuticals, and to identify in each group at least ten documents that were most alike. We took the intersection of these groups, thus obtaining one group of six documents (violence), three groups of seven documents (software, mergers and legal matters) and two of eight (pharmaceutics and networks).

5.5.2 Collecting the data

Then we collected the docreps of every document in every group. Referring to Tables 5.5 and 5.6 (in appendix I), we see the results for every group for both CLARIT and TOPIC.

The captions *post.* and *terms* indicate how many postings were scored on each group by the two systems and how many different terms were involved. As was expected, CLARIT shows higher scores than TOPIC does, but as the cut-off point for CLARIT was rather arbitrarily set on 1.0000 or 35 terms, one should not look too closely to the exact figures. Also, the CLARIT-output laboured under some irksome irregularities, such as the mis-interpreting of quotes. We edited all quotes out from the CLARIT docrep so that "IBM" was afterwards read as IBM. Spelling errors that occurred in the documents we left alone, because such errors were met too by the TOPIC system.

Another peculiarity of the CLARIT-output was, that the same phrase might occur more than one time in a single docrep; sometimes with varying and sometimes with equal weight As we were only interested in binary values (a term occurs in the docrep, or it does not occur), we disregarded double postings when computing the frequencies. To get an insight in the performance of the systems in recognising similar documents, we had to find for each group the terms that spanned the groups totally or partially (i.e. that occurred in more than one docrep) on the assumption that such terms indicate similar properties. The exact figures may be found under the caption *spanning* for all terms spanning 2-8 records. Figure 5.2 gives an impression of the relative scores.

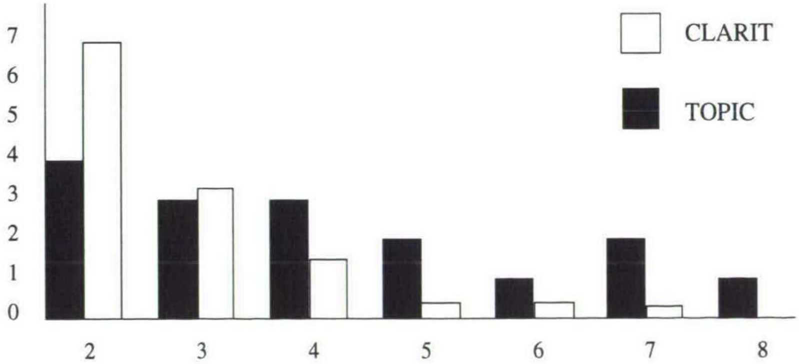


Figure 5.2: Average number of terms spanning 2-8 docreps.

Then a second measure has to be found to check if those terms do not lump too many documents together. The ideal terms would span all records in the cluster and would occur in no other records. Therefore the number of documents in a cluster, that is spanned by a particular term is not complete as a measure without the frequency of that same term in the complete database. Therefore we added for each cluster a second and a third table under both TOPIC and CLARIT. In the second table the average frequency of the spanning terms in the database is shown.

In the TOPIC docrep occur TOPICs like NUMBERS, POSITIVE-INDICATORS etc. that in itself have no bearing on the TOPICality of the document, but that are used to support other TOPICs. Such TOPICs typically have a very high frequency, whereas other TOPICs might occur in the same spanning group that do have a bearing on the TOPICality of the document, and that may have a much lower frequency. Therefore we also included for each system a third table that shows the frequency of the terms with the lowest frequency. We have omitted the average and lower score of all terms where $n < 3$ because terms that span less than 3 records in our groups were considered to be of no importance for identifying the group as a whole (Figure 5.3).

As we see, CLARIT displays far less spanning terms than TOPIC. Only in the software-group we find a CLARIT-term spanning the complete cluster and that term ('software') occurs in one of every five documents (19.6%). TOPIC has eight terms that span all documents in three of the six clusters: the average score of the frequencies is 38.5%, the average lowest frequency of these terms is 32%. When we compute the scores taken over all terms spanning more than half of the documents in all clusters, we find 29 terms with an average frequency of 26.7 (19 for terms with lowest frequency) for TOPIC and five terms with avg. 7 and avg-lowest 6 for CLARIT. Plotting these values in a precision/recall graph, we obtain the two graphs of Figure 5.4. The centre of the CLARIT-graph (i.e. the point where the number of points above and below, c.q. left and right are equal) lies at $< 0.2, 0.4 >$;, the TOPIC centre at $< 0.08, 0.5 >$, confirming the general impression that TOPIC performs better in recall, whereas CLARIT is better in precision.

5.5.3 Virtual docreps

Another interesting question is how high (or low) the agreement between the two systems lies (agreement understood as the number of identical terms in the docreps that two different indexing systems create for the same document). An obvious problem in comparing the TOPIC-docreps with the CLARIT-docreps was, that the TOPIC-docreps consisted of artificial terms, whereas the CLARIT terms were strings occurring in the documents - assigned vs. derived terms. The agreement as displayed in [Evans et al., 1991] could therefore not be computed immediatly.

What we did was first expanding every TOPIC term to those strings in the original document on which the TOPIC had fired, but that are not visible in the TOPIC docrep as displayed in Table 5.4. We will call these invisible docreps virtual docreps.

In the list with TOPICs that came with the database, we found 495 strings

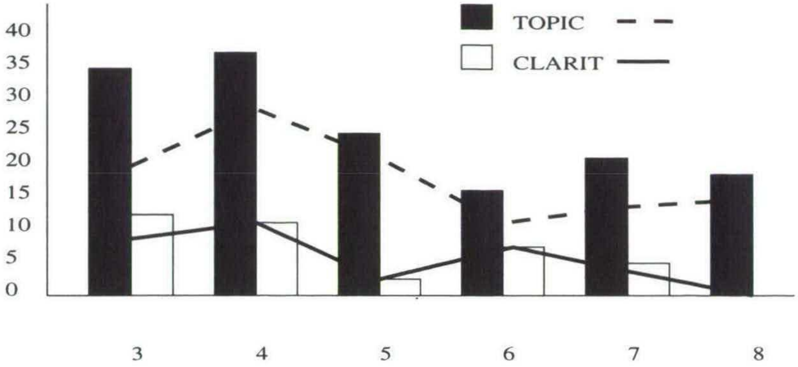


Figure 5.3: Average and lowest frequency for terms spanning 3-8 docreps.

that were considered by the authors as important. Checking in the 128 documents of the test, it was found that 205 of these leaves actually occurred. Comparing this list of 205 terms with the 2019 terms from the truncated CLARIT-list, we found that the number of terms that are both in the CLARIT docreps and in the TOPIC docreps was surprisingly small. Only twenty-six terms were identical. Of these 14 are proper names or acronyms. Checking how many times such TOPIC-leaves were constituents of CLARIT terms resulted in 424 cases (note that in this last case the CLARIT-term "computer company" would score on 'computer', on 'company' and on 'computer company' if all three existed as separate TOPIC-leaves).

This is rather surprising, because this result does not tally with the high level of agreement reported by Evans. If we take TOPIC to be an alternative indexer that performed on the same level as the human indexers in the Evans-experiment, one would expect on grounds of the tables on page 51 and 52 of the CLARIT evaluation that the agreement by words would be higer (typical 20-40) in every article and at least a few terms in every docrep would overlap. This evidently is not the case in our database: TOPIC and CLARIT seem to group its documents on widely different terms. Of course TOPIC is not a human indexer, but at least the keywords in the TOPICs are selected by humans, so one would expect more agreement. Possibly this lack of agreement is a result of the assigned vs. derived technique, the assignments of TOPIC converging towards more general concepts, while the compound terms of CLARIT diverge towards ever more specific terms.

5.6 Conclusions and suggestions

It should be stressed that the differences that the two systems display relative to each other in the quantitative tests, should not immediatly be translated in qualitative judgements. Offhand TOPIC seems to score better when groups of

				spanning								
TOPIC		postings	terms	1	2	3	4	5	6	7	8	
violence	6	46	23	12	5	3	0	5	0	-	-	
software	7	129	44	20	2	5	8	1	3	5	-	
pharma.	8	32	22	18	1	1	1	1	0	0	0	
networks	8	127	42	20	5	1	4	4	1	5	2	
mergers	7	61	26	8	11	3	1	0	2	1	-	
legal	7	45	27	19	1	4	3	0	0	0	-	
		73	31	16	4	3	3	2	1	2	1	
TOPIC	TOPIC: avg. freqencies in %											
violence	6					44	0	30	0	-	-	
software	7					7	16	33	37	36	-	span. max
pharma	8					56	73	70	0	0	0	37.7
networks	8					44	37	34	7.8	14	35	
mergers	7					36	70	0	27	42	-	avg. freq.
legal	7					36	33	0	0	0	-	26.4
						37	38	28	12	18	18	
TOPIC	TOPIC: avg. lowest-freqencies term in %											
violence	6					4.7	0	10	0	-	-	
software	7					3.9	6	33	30	19	-	span. max
pharma	8					56	73	70	0	0	0	31.3
networks	8					44	10	6.2	7.8	9.5	33	
mergers	7					13	70	0	24	42	-	avg. lowest
legal	7					7	14	0	0	0	-	18.9
						21	29	20	10	14	17	

Table 5.5: Test results for TOPIC

documents are to be retrieved, that cover a broad concept, or when a concept is described using many different, but identifiable terms. CLARIT seems to perform better when the documents display a marked terminology, because such terms are readily recognized against the background of the corpus. This would lead to the conclusion that the TOPIC-system is more apt for big libraries that cover a rather wide spectrum of subjects. The very specificity that CLARIT displays, would point to a possible use in smaller document collections, or collections that limit themselves to a very specialized subject with users that know the specific terms of the trade.

Observing that the retrieval engine of TOPIC works with a derived dictionary, one might ask if CLARIT, itself a derived dictionary system, could be applied to create the TOPIC dictionary for subsequent use with the 'intelligent' TOPICs.

There are some problems in this respect. The evidence that TOPIC looks for in order to decide on the assignment of the document to a TOPIC, is the occurrence of any kind of string (not only NP's) or combination of strings (positional information included). CLARIT, on the other hand, only looks for NP's and discards all positional information after the candidate terms are generated. However, an inspection of the leaves of TOPIC hierarchies shows that they are almost exclusively nouns and noun phrases, so there is no inherent reason why they could not occur in a CLARIT-dictionary of the document. A second condition for inclusion would be that the frequency of the noun, or

				spanning								
CLARIT		postings	terms	1	2	3	4	5	6	7	8	
violence	6	104	95	89	3	2	0	0	0	-	-	
software	7	199	170	154	10	3	1	1	0	1	-	
pharma.	8	179	150	134	10	3	1	1	1	0	0	
networks	8	223	191	169	10	6	5	0	1	0	0	
mergers	7	138	125	115	5	4	1	0	0	0	-	
legal	7	118	114	111	2	1	0	0	0	0	-	
		160	141	129	7	3	1.3	0.3	0.3	0.2	0	
CLARIT	CLARIT: avg. freqencies in %											
violence	6					22	0	0	0	-	-	
software	7					5	12	5.5	0	19	-	
pharma	8					3	25	7	27	0	0	
networks	8					23	5	0	1.8	0	0	
mergers	7					13	25	0	0	0	-	avg. freq
legal	7					15	0	0	0	0	-	7.3
						14	11	2	8	4	0	
CLARIT	CLARIT: avg. lowest-freqencies term in %											
violence	6					16	0	0	0	-	-	
software	7					4	12	5.5	0	19	-	
pharma	8					1	25	7	27	0	0	
networks	8					2	2.3	0	18	0	0	
mergers	7					1	25	0	0	0	-	avg. lowest
legal	7					15	0	0	0	0	-	5.8
						6	11	2	8	4	0	

Table 5.6: Test results for CLARIT

noun phrase, is such that it will be kept by the various statistics that CLARIT applies. This, however, seems to offer not much perspective as the agreement between CLARIT and TOPIC seems to be very low in this respect. Or, the words and phrases that CLARIT extracts, typically are not the words and phrases that have been included in the TOPICs.

The various counts and experiments in this explorative paper should be followed up by more research in bigger databases, using a real-life test. After a solid measuring system for these and similar programs has been arrived at, the human-selected TOPIC leaves should be replaced by phrases that have been selected by CLARIT (or from a CLARIT-generated list) in order to check if performance will improve.

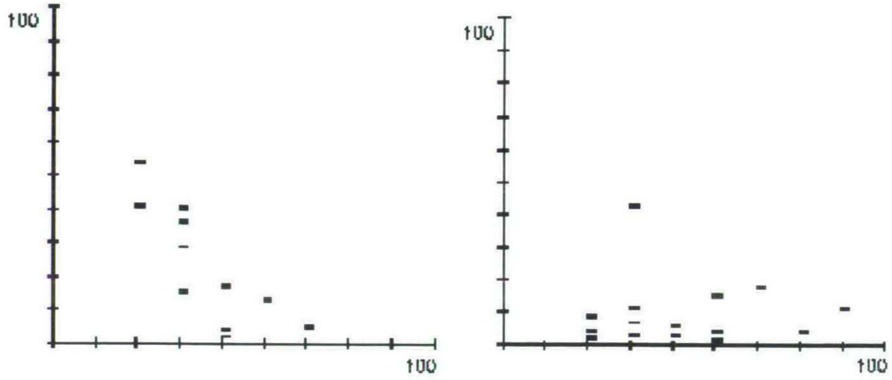


Figure 5.4: Precision vs. recall in CLARIT (left) and TOPIC (right)

5.7 APPENDIX

Example of the documents in the database. Most figures have been drawn from this document.

5.7.1 Document140

```
ACQUISITION OF PATRIOT ENERGY COMPANY LTD. ANNOUNCED

  CALGARY, Alberta, Feb. 12 /PRNewswire/ -- Giant Pacific Petroleum Inc.
('Giant Pacific') (NASDAQ, Vancouver: GPP) today announced that it has acquired
all of the 5,333,528 common shares of Patriot Energy Company Limited
('Patriot') representing approximately 94 percent of the outstanding shares of
Patriot, tendered pursuant to the take over bid offer for all the shares of
Patriot.

Payment for the shares tendered will be made as soon as possible.  The warrants
to be issued pursuant to the offer will be listed for trading on the Vancouver
Stock Exchange ('VSE') in due course provided that the listing requirements of
the VSE are complied with.  Giant Pacific intends to exercise its statutory
rights under the compulsory acquisition provisions of the Business Corporations
Act (Alberta) to acquire the remaining shares of Patriot and consequently to
hold Patriot as a wholly
owned subsidiary.

  Giant Pacific trades on the VSE and NASDAQ.  Its principal assets are oil and
gas producing properties in Texas.  Patriot is an active junior oil and gas
```

exploration company which has holdings in western Canada with oil production in Saskatchewan.

The Vancouver Stock Exchange has neither approved or disapproved of the information contained herein.

/CONTACT: Bruce Weaver, president, Giant Pacific Petroleum Inc., Toronto,
416-941-9440/
10:35 EST

Chapter 6

Gravity wells of meaning[1]

Abstract: *Four term-weighting schemes are used to detect passages in texts that may be rich in information and the results are compared. It is demonstrated that word categories and frequency-derived weights have a close correlation but that weighting according to the first mention theory or the cue method shows no correlation with frequency-based weights.*

6.1 Introduction

Although several attempts have been made to create information retrieval systems that are based on more involved document representations, the models that are based on indices, or inverted files of words occurring in the text proper, still offer the most efficient solutions.

The obvious problem when creating such indices is how to decide on the relative importance of a word in a text as an indicator for the topicality of that text. Various strategies have been proposed, among which the methods that are based on the frequency characteristics of words in texts have attracted much attention because of the relative ease with which they may be applied, while the performance in terms of precision and recall is reported to be among the best. Nevertheless the performance of IR-systems that have to operate on big document bases is an ongoing concern, if only because the quantity of on-line, machine readable texts is growing rapidly.

Other methods, while intuitively attractive and also easily applied, don't seem to score as high as the frequency based methods (see for instance the conclusions of [Keen, 1992] or [Cleverdon, 1991]) and there are not many real life applications. Among these alternatives for term weighting are the cue methods and the positional methods (for a concise survey of these and other methods see [Paice, 1990]). It must be added that the frequency based weighting is not popular in commercial systems either. With a few exceptions the

[1] This chapter appeared under the same title 'Gravity Wells of Meaning: detecting Information-Rich passages in Scientific Texts' in the Journal of Documentation 53(5), 1997, pp. 520-536 ([Paijmans, 1997]). The bibliography entries have been collected in the bibliography at the end of this book.

Boolean model reigns supreme, although the inferiority of this model was exposed conclusively by [Blair and Maron, 1985] over ten years ago.

A possible reason for this reluctance from the part of commercial ventures to invest in non-trivial IR techniques is offered by [Blair, 1996], who argues that it is notoriously difficult to measure the performance of systems in the intellectual accessing of documents as opposed to the physical access and that industry is naturally slow to invest in techniques that can show no obvious and precise results. Straightforward keyword retrieval according to the ubiquitous Boolean model is easy to understand and the improvements offered by new indexing and storage techniques are easily measured, but it is far more difficult to measure the benefits of the application of statistics, fuzzy logic, semantics or other techniques that are concerned with conceptual retrieval. As he points out: in the 85 commercially available systems described by the Delphi survey [DELPHI, 1992] only five were listed as offering 'concept based retrieval', meaning retrieval following other models than the Boolean model. In the late sixties and seventies, when word-frequency characteristics emerged as good content indicators, their application on Natural Language documents was done on very short texts such as abstracts. It is important to realize that the general availability of the full text of documents has occurred only in the last decade or so and in the meantime the focus had shifted to other areas of IR such as the application of techniques from AI research, more in particular the creation and manipulation of knowledge structures.

The interest in statistics and frequencies has increased again after 1990, owing to the accessability of really large corpora and increased procesing power of computers. Often such research centered on how to divide the text in coherent and manageable passages [Buckley and Salton, 1991], [Salton and Buckley, 1993], [Hearst and Plaunt, 1993] or [Callan, 1994]. Researchers recognize three classes of such passages: *discourse* passages that are based on textual discourse units, *semantic* passages based on the subject or content of the text and *window* passages that contain a certain number of words. The experiments by Callan showed that at least passages based on paragraph boundaries were less effective than passages based upon overlapping text windows of varying sizes. On the other hand [Hearst and Plaunt, 1993] and others [Morris and Hirst, 1991] showed that boundaries that define areas with different topicality do exist in texts and that such boundaries may be detected e.g., by chaining or by comparing frequency characteristics. Moreover, Hearst showed that in her experiments those boundaries co-occurred with logical document partitions such as chapters and sections.

The importance of such findings for the design of IR systems is obvious. If a system can immediately present of a retrieved text those passages that best match the query, a user can more quickly decide if the document meets its needs. Also it may be important that the system can rank documents on the basis of the fact that word-matches are either scattered through the document or clustered in a dense region of matches, e.g. by so-called 'tile-bars' [Hearst, 1995].

This gives raise to a different question: if passages of differing topicality can be recognized, would it also be possible to recognize the relative *importance* of various passages? That is: if we have a set of retrieved documents, we would certainly profit from a technique that would display the most important passages from those documents. These passages do not necessarily coincide with passages that most resemble the query. This works the other way around, too: if we had a way to identify such passages, would not such parts of the text be the obvious candidates for indexing?

6.2 Organization of experiments

The experiments of which the outcome is presented here are part of a line of research concerning the behaviour of information-rich words in relatively long natural language documents such as journal articles and reports.

The working hypothesis of this paper will be that words in different parts of the document have differing informational values. Or, information-rich words are not distributed at random, but they tend to gravitate towards certain places in the document. As a theory this is not new. The article of Paice mentions some of the strategies with which to compute the informational value of certain text passages, such as sentences. However, we will try to add some new vantage points and, more important, try to measure such methods in terms of each another.

Consider Figure 6.1. Depicted are the average weights of the words in every sentence of a single document (a scientific article on qualitative reasoning). The vertical axis indicates the *tf.idf* weights (to be explained later); the horizontal axis gives sentence numbers. The dotted line shows the average *tf.idf* of the words in every sentence. The thick line is a smoothing function applied to those values and the straight vertical lines indicate new logical sections in the document as defined by the author. Visible are a few marked peaks where sentences seem to have a consistent high average score. Let us borrow a term from physics and call such places *gravitation wells.* In this paper we will try to establish whether the occurrence of such gravitation wells can be predicted by the weighting methods that are under consideration here.

To test this, we will divide sets of documents into sub-documents that correspond to interesting parts of the original documents (interesting according to for instance the *first mention* theory of Kieras). Then we will check if significant differences in the weights of the terms occur between those sub-documents. The outcome of the experiments will give us, if not an indication of the absolute value of the weighting methods that were applied, at least an insight in the information value of the different parts of the document relative to the methods.

When looking for these peak distributions of words with high information content, we will for the purposes of this paper focus on three main areas of interest:

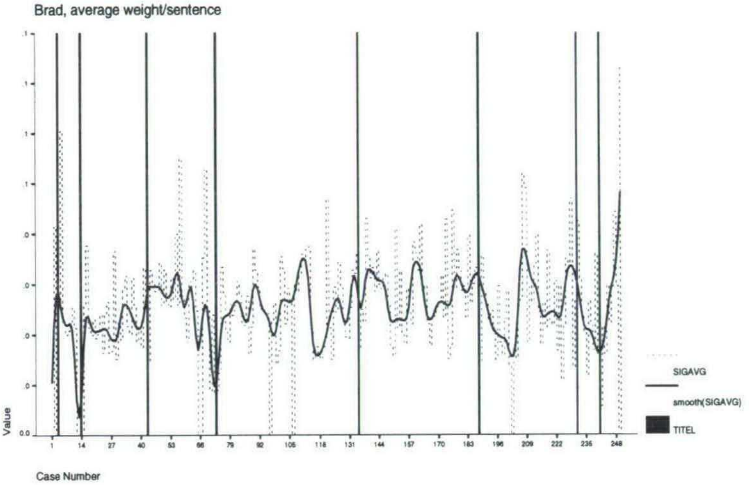


Figure 6.1: tf.idf weights of sentences in a single document

Positional aspects. The question here is if the position of the word in relation to the structure of the document is indicative for its information value. 'Position' may be understood as indicating a logical part, e.g. the title, abstract or perhaps 'itemized' parts (as this one is). It may also be another positional feature such as the n-th sentence from the beginning of a section or a paragraph.

Weak semantics. Another approach, already proposed by early workers in the field like [Edmundson, 1969] is called the cue-method. The assumption is that words with a bearing on the contents of a text will be found near certain cue-words or -phrases. We prefer to call this weak semantics because use is made of the general semantic 'mood' of the cue-words rather than its exact meaning.

Word categories. The search for gravitation wells may also extend in other dimensions like the difference in syntactical structure or the distribution of word categories. An early attempt in this direction was done by [Earl, 1970], who tried to find correlations between information content and sentences with certain syntactical structures. The results, however, were negative. On the other hand it has been established that nouns, or more generally NP's, have a greater informational value than other words of the document [Ginther-Webster et al., 1991].

files	24
avg.length	33.3 Kb
tf.idf: Mean	.04
tf.idf: Std Dev	.03
tf.idf: Min.	.00063
tf.idf: max.	.18245
Tokens	21439
Types	7257
Tokens (stemmed)	12392
Types (stemmed)	3989

Table 6.1: Descriptives for SACJ.

We will try to confirm or dismiss such theories by comparing them with the results obtained by frequency-based word weights. In this paper we will first look at the positional aspects as in the first item, more in particular to the first and last sentences of paragraphs, where according to the *first mention theory* concepts that are central to the text should be mentioned. Edmundson's theories of cue-words will be represented by only two cue-strings: 'importa' and 'significa'. Finally we will look at the word categories of the documents onder consideration.

Although we are working with several corpora, in this paper we will only present the outcomes of a small corpus with 24 scientific articles from the South African Computer Journal (see Table 6.1 for general statistics of the database).

The original SACJ articles often had tables, figures and formulas inserted. As they were marked up in LaTeX it was relatively easy to skip these and similar constructs at will. Therefore the SACJ-texts that we ultimately worked with consisted of the original text minus tables, figures and formulas. Itemized and enumerated parts of the text were kept and marked as such. In Table 6.1 the entries *tf.idf: Mean, Std.dev, Min* and *Max* refer to the *tf.idf* weights computed over the complete set of documents. The entries *Tokens* and *Types* should be self-explanatory; *Tokens (stemmed)* and *Types (stemmed)* tell us how many tokens and types are left after the application of stemming and the pruning of function words.

6.3 Weighting schemes

In this section we will consider six strategies to detect information bearing words in full-text documents: the *tf.idf*, the discrimination value, logical structure, the first mention theory, weak semantics and word categories. The first two strategies will be explained in some detail because in recent experiments by [Littin, 1995] on the automatic classification of documents such methods, notably the *tf.idf*, still showed the best performance in recognizing the subject matter of a document.

6.3.1 Frequency-based term-weighting schemes

There are a number of weighting schemes that use the frequency of the words within documents and over the database as a measure for the suitability of that word as a keyword. The most popular of these schemes is the so-called $tf.idf$ weight, or rather *one* of the $tf.idf$-related weights, as there are several variations. The $tf.idf$ is composed of the term frequency (tf) and the Inverse document frequency (idf) or one of its derivates or normalizations. For each term t and document d, tf is the term frequency of t in d. The collection frequency of term t for N documents is the sum of all tf's. The document frequency of term t, D_t, is the number of the documents in which t occurs.

An appropiate indication of a term as a document discriminator can be computed by taking an inverse function of the document frequency of that term, e.g. $idf = N/D_t$, or $idf = logN/D_t + 1$. The product of the term frequency and the inverse document frequency, $tf.idf$, may then be used as an indicator for the importance of a term in a document.

A popular variation is the so-called *atc*-weight. It calculates the $tf.idf$ in three steps. The first step creates the value new_tf for the term-frequency (tf) as

$$new\_tf = 0.5 + 0.5 * \frac{tf}{max\_tf}$$

where max_tf is the frequency of the term with the highest frequency in the document. Then the weight new_wt is calculated as

$$new\_wt = new\_tf * log\frac{N}{D_t}$$

where N is as before the number of documents and D_t the document frequency of term t. Finally the cosine normalization is applied by

$$new\_wt' = \frac{new\_wt}{\sqrt{\sum_{i=1}^{T} new\_wt_i^2}}$$

where T is the number of terms in the document vector.

For a detailed discussion of these and similar techniques see e.g. [Salton and McGill, 1983] and [Salton, 1989].

6.3.2 The term discrimination value

A different method to approach the information weights of individual words is the computation of the term discrimination value. The documents are imagined as a cloud in which documents that are similar to each other form clusters. The keywords that represent the documents influence the density of the cloud: 'good' keywords bring similar documents closer to each other and farther away from dissimilar documents. The discrimination value of a keyword is computed by comparing the density Q of the document-cloud without the keyword i with the density Q_i of the cloud with keyword i added to it:

$$dv_i = Q - Q_i$$

Q is computed by taking the average $(N(N-1))$ pair-wise similarity between all possible documentpairs:

$$Q = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{k=1, i \neq k}^{N} sim(D_i, D_k)$$

This is simplified by constructing a dummy document at the centre of the document cloud: the *centroide* $C = (c_1, c_2, ..., c_t)$, in which every c_j is the mean of all j^{th} terms in the document base:

$$c_j = \frac{1}{N} \sum_{k=1}^{T} d_{kj}$$

and the formula is simplified to:

$$Q = \frac{1}{N} \sum_{k=1}^{N} sim(C, D_k)$$

That leaves us with the problem how to compute the similarity between documents. There exist a number of techniques for that as described a.o. by [van Rijsbergen, 1979]. Following Salton we used the cosine function

$$cos(d_j, d_k) = \frac{\sum_{i=1}^{m} d_{ji} \bullet d_{ki}}{\sqrt{\sum_{i=1}^{m} d_{ji}^2 \bullet \sum_{i=1}^{m} d_{ki}^2}}$$

which is the most common similarity measure in this context and used in the experiments of [Willett, 1985], [El-Hamdouchi and Willett, 1988] and [Crouch, 1988]. For our own experiments we adapted a program, originally published in [Frakes and Baeza-Yates, 1992].

6.3.3 Differences between the two frequency-based methods

The final interpretation of the term discrimination weights is very unlike that of the *tf.idf* variations. The latter computes a document-word weight that will be different for every new document; the former a word weight that remains constant over the database. If words in documents are to be weighted individually using the discrimination value, it is multiplied with the frequency of the word, which acoording to [Salton et al., 1975] gives 'excellent' retrieval results.

For an understanding of how the term discrimination value is used to weigh words, we first rank the values from 1 for the best discriminator to t for the worst. Then we rank the terms according to their document frequency from low to high.

The distribution of term discrimination values may be divided in three regions:

- Terms with very low document frequencies are poor discriminators, but the average rank of the Dv is high.
- On the other end of the scale the terms with high Df are the worst discriminators. The average rank of the Dv here is very low.
- The bottom of the curve is inhabited by keywords with a Df which is neither too high or too low; the Dv generally is below $t/5$.

If the value of the Dv is replaced by its rank, the correlation between information value and discrimination value may be drawn as an U-shaped curve (see e.g. [Salton et al., 1975]). This U-shape is already evident in in the small SACJ-database (see Figure 6.2) when the average Dv is plotted against th

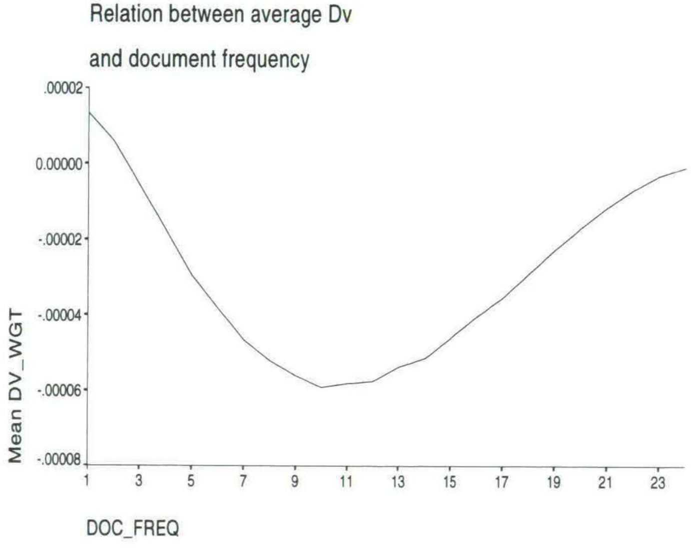


Figure 6.2: Information values of the Dv in SACJ

When comparing the values of the *atc*-weights and the *discrimination values* as explained above, the correlation between the two was striking. The correlation between the Dv and the mean of the *atc*-weights in the SACJ-database produced a Pearson's r of 0.84. The correlation between the product of the Dv and the term frequency was rather lower, but with an r of 0.27 still significant. Please note that in our experiments the function words, that typically exhibit a very low discrimination value, were omitted.

6.3.4 Logical document structure

Other approaches are based on the emphasizing of terms that occur in certain parts of the document. An obvious example is the increased importance that may be attached to words occurring in the title or the bibliography. This idea is easy to grasp for users of relational databases. They are familiar with the idea that the 'semantics' of data are defined by the position in the record.

In documents, however, this is only true for such data as *author* or *place and date of publication*. The relation between meaning and 'fields' like *title* or *abstract* is far more tenuous. On the other hand it has been demonstrated that a case can be made for different 'weights' of certain parts of a document. This is demonstrated by the experiments of Kwok on the relevance of words occurring in the bibliography [Kwok, 1984]. In this study Kwok found that words taken from the bibliography of a document performed better as content-indicators than words from other parts. When this experiment was repeated by [Salton and Zhang, 1986] on different databases they concluded that the effect was not as general as Kwok had suspected. A more involved approach to using document structure as a factor in both information retrieval and the presentation of the results is to be found in the Druide-project [Mulders et al., 1992].

6.3.5 The first-mention theory

Another theory was described by [Kieras, 1985]. He put forward the assumption that an author works following a reasonably fixed scheme or pattern of first stating the topic of a paragraph, then expanding on it and finally reaching a conclusion. The central claim of Kieras is that *"there seem to be common text grammars that specify where in the passage important information is likely to appear, and there are several surface-level signals that mark individual items of information that are important to the passage macrostructure"* [Kieras, 1985], p.95. However, seen from the information retrieval point of view, the question is if such patterns are consistent enough to use them as markers for the topicality of texts.

6.3.6 Cue-words or 'weak semantics'

An intuitively attractive theory for deciding on the relative importance of words and phrases is based on the assumption that the author himself gives explicit signals to this effect. For example, sentences (or perhaps rather passages) in which cue-words or -phrases like 'This is important...' occur are *ipso facto* considered to contain terms that are relevant as content indicators. In an early experiment by [Edmundson, 1969] it was found that such methods did "somewhat better" (to quote [Paice, 1990]) than the frequency-keyword methods of that time. The experiments of Edmundson are over twenty years old and according to Paice no other detailed evaluations have been reported since that time.

	Pearson	P
R-1ST (first sentences)	.0047	P= .299
R-LAST (last sentences)	-.0293	P= .001
R-FILA (one-sentence paragraphs)	.0286	P= .001
R-TIT (titles and subtitles)	-.0589	P= .000
R-ABS (sentences in abstracts)	-.0549	P= .000
R-CUE (sentences with cue-words)	-.1068	P= .001

Table 6.2: Correlation coefficients for SACJ.

6.3.7 Word categories

Finally it is assumed that the interesting information from an IR point of view is stored in nouns or noun phrases [Ginther-Webster et al., 1991]. Therefore we would expect that document representations consisting of nouns or NP's perform consistently better than document representations built from other word categories.

6.4 Methodology

In this study we chose as the most important yardstick for the information weight of a keyword the *atc*-variation of the *tf.idf* method. For the computations of the weights we used the SMART retrieval system. This program incorporates both a stemming algorithm and a list with stop- or function words; we used both features.

The assumption was that if the various parts of a document differed in information value and if the *atc*-weight was a good measure for that information value, that then the means of the weights of the words in different document passages would show significant differences and that a positive correlation would exist between this weight and the tendency of the word to occur in a specific place of the document.

We used three methods to test this assumption:

- the Pearson test to check for a correlation between the *tf.idf* weights and the positions of the words in the document,
- the T-test to compare the means of word weights in various positions in the document and
- the T-test to compare the means of *sentence weights.*

Finally we checked the hypothesis that the *tf.idf* was a good information indicator again by comparing the weights of function words and non-function words.

6.4.1 The Pearson-test

Before we performed the correlation-test, we first computed a variable that indicated the tendency of a word to occur in a certain position. To obtain such a measure we divided for every word its frequency in a particular subdocument (e.g. the subdocument composed of all first sentences of paragraphs) by its allover frequency in a document. For instance, if the frequency *F-1ST* (frequency of the word stem in *first sentences* of paragraphs) of the stem 'abandon' was 3, and the frequency *F-TXT* in the complete document was 10, we created a new variable *R-1ST = F-1ST / F-TXT* with the value 0.3. Thus R-1ST is a measure of the tendency of the stem 'abandon' to occur in a first sentence.

In Table 6.2, we show the correlations between various subdocuments and the word weights. If a positive correlation exists between the variables and the weights of the words, it would support the hypothesis that words with a high information content tend to cluster in the first sentence. We used Pearson's r as a measure for the correlation of the data:

$$r = \frac{\sum_{i=1}^{N} (X_i - \bar{X})(Y_i - \bar{Y})}{(N-1)S_X S_Y}$$

where S_X and S_Y are the standard deviations of the two variables.

On first sight the correlations between weight and the relative occurrences in various subdocuments look promising. Except for the group *R-1ST* (first sentences), all other groups displayed a high probability of correlation (a $P < 0.05$ in the table). However, the Pearson coefficient also was very low and the high correlation probability has to be attributed to the great number of observations (= number of stems; 12,392 for SACJ). Also, on closer inspection four out of five variables with a significant P had the wrong sign in the correlation coefficient, indicating a negative correlation, where a positive correlation was expected and so there is a suggestion that, in fact, the weights of the words in our subdocuments is on average *lower* than that in the original document.

6.4.2 The T-test on word stems

We then proceeded to compute the differences between the means of the word-weights occurring in the subdocuments and in the complete documents as another test to decide if the word weights were from different populations (refer to Table 6.3). This was done by comparing the means of the weights of word stems that occurred once or more in a group of subdocuments and the weights of stems that never occurred in that group (its complement). In the table the subdocuments are indicated by the variables whose names begin with F-. Every row describes the results of comparing the weights of stems in a group of subdocuments with its complement. The rows F-RN0, F-RN1, F-RN2 and F-RN3 describe four control variables and give the results for four groups of subdocuments that were collected at random from the original documents.

	Variances	tvalue	df	2Tail Sig	SE of Diff
F-1ST	U	-9.82	9144.41	.000	.000
F-LAST	E	-11.88	12390	.000	.000
F-FILA	U	-1.44	1398.14	.150	.001
F-TIT	U	-3.91	593.84	.000	.001
F-ABS	U	-9.27	1183.70	.000	.001
F-RN0	E	-6.99	12390	.000	.001
F-RN1	E	-7.89	12390	.000	.001
F-RN2	E	-6.98	12390	.000	.001
F-RN3	E	-9.79	12390	.000	.001

Table 6.3: T-tests for SACJ.

Table 6.3 shows the results. '(E)qual' or '(U)nequal' in the first column (*Variances*) is dependent on the outcome of Levene's test for equality of variances. The probability of equal means is in the column *2Tail.Sig.* A low score indicates a low probability that the means are equal. This table shows the same tendencies as Table 6.2. Although the variables under consideration show a high probability of real differences in the means of the weights, the control variables F-RN0, F-RN1, F-RN2 and F-RN3 show the same pattern in all columns except for the variance of the means. Here Levene's test indicated that the subgroups compared at least had differing variances, while the control variables all had equal variances. This in itself is interesting, but we have not yet looked for an explanation.

We then made a new selection of word stems from the original SACJ-database, excluding all stems with a *tf.idf* weight lower than 0.04 (inclusive) and calculated the T-tests again, to test the hypothesis that a smaller number of words with higher weights would show clearer results. However, this was not the case and the conclusion again has to be that the normal T-test does not reveal important differences between the sentences under consideration, or, if they do, that such differences are not marked enough to be used for automatic indexing purposes.

6.4.3 T-tests on sentences

Then a third approach was choosen in which not the individual words were considered but the complete sentence. From the above tests it might be conjectured that, perhaps, the words selected after application of stopwords and stemming algorithm might not differ, but that the first or last sentences of paragraphs differed in the ratio *function words - significant words* or that another unexpected factor accounted for the low differences of the means.

For each original database a second database was created in which for every sentence in every file the mean of the weights of the words in that sentence was computed (see Table 6.4). In fact we computed two averages, one with only the significant words (stop- or function-words excluded) and a second one for all words, stopwords included. These groups are marked AVG-SIG and AVG-TOT in the table. The first column of the table refers to the subdocuments

	avg.	Variance	tvalue	df	2Tail Sig	SE of Diff
FST-SEN	AVG-SIG	U	-3.58	2593.95	.000	.001
	AVG-TOT	U	6.76	3692.19	.000	.000
LAST-SEN	AVG-SIG	E	2.04	4598	.042	.001
	AVG-TOT	E	4.16	5361	.000	.000
FILA	AVG-SIG	U	1.62	437.14	.107	.001
	AVG-TOT	U	-5.73	735.80	.000	.001

Table 6.4: T-tests for average sentence-weights (SACJ).

(FST-SEN and LAST-SEN refer to the first and last sentences respectively, while FILA refers to the single-sentence paragraphs). The second column (avg) indicates whether only the weighted words (AVG-SIG) are used to obtain the average, or that all words are used (AVG-TOT). The other columns are the same as in Table 6.3. Table 6.4 displays the results. Note again the differences in variance.

6.4.4 Word categories and word weights

Seeing that the results of the tests described above were inconclusive, we had to face the possibility that in our database the *tf.idf* was not suitable after all to be used as a central measure for the informational content of a word in a document. We therefore sought an independent confirmation of the informativeness of the *tf.idf* weight by correlating high weights with word categories. Also, this would add a different dimension to the position- and cue-oriented methods.

To obtain word categories we used the Xerox tagger from Cutting and [Cutting et al., 1992]. This tagger assigned tags according to the list of tags from the LOB Corpus, a code of one to five characters. This code is consistent so that a tag beginning with 'n' always is a noun, the second and following characters indicating whether it is a common noun, a proper noun, a genitive and so on.

To match a word category with a word stem we decided to just take the first match of a word(stem) in the Smart database with a tagged word in the document and assign all other similar words to that category. Of course this introduced a potential error in cases where similar words were tagged differently. We then aggregated the word categories over the database, obtaining the average weights for each category.

After computing the average weight for every word category as described above, we ordered the table on the weights and made a rather voluntary treshold on one third of the table. On that point something more than 75% of the words in the database had been accounted for. The *verbs*, *adjectives* and *common nouns* are the biggest groups in the database. As might be expected it turned out that the non-function words as nouns, adjectives and verbs had high average score whereas function words, among them inflected forms of 'to have' and 'to be' scored very low (Table 6.5). In the table, the caption *weights*

cat.	atc	#wrd	perc.
pn - pron. noun	.0627	5	.05
nr - adv. noun	.0601	64	.59
rn - nominal adv.	.0600	25	.23
np - prop. noun	.0556	307	2.85
cd - cardinal	.0520	480	4.46
vb - verb	.0471	2439	22.65
jj - adjective	.0446	1235	11.47
nn - common noun	.0416	3397	31.55
rb - adverb	.0414	536	4.98
cumul. total			78.83
ql - qualifier	.0365	112	1.04
rp - adv. partic.	.0303	31	.29
at - article	.0286	170	1.58
cc - coord. conj.	.0253	110	1.02
ap - post-determ.	.0240	175	1.63
od - ordinal	.0226	33	.31
do - 'do'	.0223	28	.26
in - preposition	.0222	489	4.54
pp - pronoun	.0216	110	1.02
wr - WH-adverb	.0211	51	.47
to - infin. 'to'	.0211	36	.33
wp - WH-pronoun	.0210	31	.29
be - conj. 'to be'	.0192	180	1.67
cs - subor. conj.	.0187	169	1.57
wd - WH-det.	.0181	36	.33
md - modal aux.	.0162	127	1.18
ex - exst. 'there'	.0150	22	.20
hv - conj. 'to have'	.0143	61	.57
dt - determ.	.0133	167	1.55
ab - pre-qual.	.0129	61	.57

Table 6.5: Word categories and average weights for SACJ

is the average weight of the words in that category, the caption *#words* shows the number of words in that category and the caption *perc* the percentage of words in the texts that fall in that category. The percentages don't quite add up to 100 because we edited some spurious categories from the table.

6.5 Conclusions

On the basis of the tests described we may come to the conclusion that words with a high information content (measured according to the frequency-based weighting schemes) have no tendency to cluster in the first or the last sentences of paragraphs nor do they concentrate in paragraphs that consist of a single sentence. The results may be disappointing for those who believe that the position of the keyword in the document is an indicator of the information content of that keyword.

The fact that the well-published first mention theory does not seem to hold is peculiar. Kieras himself already had found that titles and cue-phrases had a weak effect on the signalling of otherwise clear main ideas but several studies by him did seem to confirm the importance of the first and last sentences in passages as the source of the important ideas. The tests described in this paper do not seem to bear this out, although the unequal variances for the selected sentences in Table 6.3 certainly bear investigation.

The same is true for the cue-theory of Edmundson. Although only two cue-words were considered, it was clear that sentences in which one or both of the cuewords occurred had not a higher, but a significantly lower information value as expressed in the atc-weight. This means that, whatever the actual value of cues as indicators for important passages, such passages do not contain more 'heavy' words than average passages but on the contrary do contain words that are 'lighter'.

Also the 'logical structure' of the document is not really reflected in the atc-values of the words. Although the weights of words in titles and abstracts showed a small difference in weight compared to the complete document, this difference was very small and, more important, also in the wrong dection, again suggesting that such words are on average 'lighter' than those elsewhere in the document.

The only really significant correlation was found between word category and word weight. This is intuitively right for adjectives, verbs and nouns; the consistent high weights for the nominal pronouns (anybody, everything, none etc.), the nominal adverbs (here, now, there, then etc.) and the adverbial nouns (january, sunday, east, today, home etc.) is interesting, although the frequencies themselves are very low (all below one percent). Other categories with a similar low frequency typical have far lower weights.

6.6 Acknowledgments

I take the opportunity to thank my collegues Walter Daelemans and Roeland van Hout for giving their comments on this paper and earlier versions. Also I want to thank the editors of the South African Computer Journal for giving permission to go through their files to find the articles that I used for my experiments.

6.7 Appendix: list of articles.

The following list contains the authors and titles of the articles we used for our experiments. All articles appeared in the South Africat Computer Journal 1n 1992 and 1993.

```
sacj01  John Bradshaw: Qualitative Reasoning: An Introduction.
sacj02  A. Calitz, G de Kock, D. Venter: Selection Criteria for
        First Year Computer Science Students.
sacj03  Matthew C. Clarke: Social Responsibility for Computing
        Professionals
sacj04  Hendrik Theron, Ian Cloete: Beam Search In Attribute-based
        Concept Induction.
sacj05  J. Daniel Couger, D C Smith : Evaluating The Motivating
        Environment for Information.
sacj06  de Villiers, A Model Checker for Transition Systems
sacj07  K. MacGregor: The Design and Analysis of Distributed Virtual
        Memory Consistency Protocols in an Object-Orientated
        Operating System
sacj08  R. J. Foss, G. M. Rehmet, R. C Watkins: A Multitasking
        Operating System above MS-DOS
sacj09  P. H .Greenwood, P. E. Nash: Accessing Subroutine Libraries on
        a Network
sacj10  W. A. Labuschagne, P. L. van der Westhuizen: Logic Programming:
        Ideal vs. Practice
sacj11  G.Oosthuizen and C. Avenant: Integrating Similarity-Based and
        Explanation-Based Learning.
sacj12  A Melton, A Lattice-theoretic Model for Relational Database
        Security.
sacj13  B. C. O'Donovan: Using Information Systems Methodology to
        Design an an Instructional System
sacj14  A. Roland, ea.: a Higher Order Logic Programming Language
sacj15  N. Pendock: From Batch to Distributed Image Processing: Remote
        Sensing
sacj16  G. Wheeler P.S. Kritzinger: PEW, A Tool for the Automated
        Performa Analysis of Protocols.
sacj17  J. Daniel Couger, D C Smith: Evaluating The Motivating
        Environment for Information Systems Personnel in South Africa
        Compared to The United States
sacj18  D. C. Smith, S Newton and M J Riley: An Evaluation of the Skill
        Requirements of Entry-level Graduates in the Information Systems
        Indus.
sacj19  B. Shane ea. :Semantic Constructs for a Persistent Programming
```

Language

sacj20 P. Z. Theron, I Cloete: Automatically Linking Words and Concepts in an Afrikaans Dictionary

sacj21 G. Tredoux: Mechanizing ExecutionSequence Semantics in HOL

sacj22 Lynette van Zijl and Deon Mitton: Using Statecharts to Design and Specify a Direct-Manipulation User Interface

sacj23 Etienne van der Poel, Ian Cloete: Animating Neural Network Training.

sacj24 H. L. Viktor and M. H. Rennhackkamp: Network Partitions in Distributed Databases.

Chapter 7

Local dictionaries and feature selection for text classification and information retrieval[1]

Abstract: *A number of methods for feature reduction and feature selection in text classification and information retrieval systems are compared. These include feature sets that are constructed by latent semantic indexing, 'local dictionaries' in the form of the words that score highest in frequency in positive class examples and feature sets that are constructed by relevance feedback strategies such as Rocchio's feedback algorithm or genetic algorithms. Also, different derivations from the normal recall and precision performance indicators are discussed and compared. It was found that categorizers consisting of the words with highest* $tf.idf$ *values scored best.*

7.1 Introduction

If ten or twenty years ago authors complained about the information explosion that caused the number of articles and books to increase arithmetically, they never dreamt of the completely new channels of Internet and WWW that make the problems of twenty years ago small by comparison. Not only the quantity of text keeps multiplying, but the character of Internet is such that there is absolutely no check on the quality of the contents. The papers that had to be indexed twenty years ago often were published after a refereeing process had taken place. Moreover the cost of production and multiplication acted as a natural threshold. In the electronic publishing space of Internet there is no such arbitrage and the danger of desinformation clogging the channels is not any more an abstract menace but an everyday reality.

[1] This chapter appeared under the title 'Text Categorization as anInformation Retrieval Task' in the South African Computer Journal, 21, pp. 4–15 , 1998 ([Paijmans, 1998]). The bibliography entries have been collected in the bibliography at the end of this book.

Although the cost of software for information retrieval and corresponding hardware has decreased dramatically at the same time, many researchers maintain that the capability of the existing techniques to retrieve information from the Internet is not up to this gigantic challenge [Lawrence and Giles, 1998]. Part of the problem is the sheer quantity of documents that have to be accessed and indexed. A partial solution is the application of text categorization techniques that would screen the documents before they are processed by the more expensive indexing and retrieval systems [Lewis and Ringuette, 1994].

Although it might be argued that information retrieval itself is a classification activity, because it divides the documents of a document base in the two classes 'relevant' and 'not relevant', we will consider both IR and text classification as separate activities. We will moreover limit ourselves to the subset of text classification that is called 'text categorization', the activity in which classification is performed on classes that have been defined beforehand.

In this paper we will try to reconnoitre some of the aspects that arise when techniques from information retrieval and categorization are combined, more in particular those aspects that have to do with the selection of features from documents for indexing and classification purposes. This reconnaisance will consist of the application of a number of different techniques on a single, well-published database, the Reuters-21578 corpus, paying special attention to the performance of so-called local dictionaries.

7.1.1 The index language

In the classic model of IR, both documents and queries are translated to an intermediary language, the *index language*. A simple implementation of this index language often looks like a list of words coupled to the documents in which they occur. The preferred model for such indices is that of a keyword-document matrix, where every keyword in the database is a separate attribute that for every document can have the value zero or one (or other values, such as the word frequency), according to the occurrence of that keyword in that document. These attributes are considered to be *symmetric* or *orthogonal*, i.e. that there exist no special relations between individual keywords. If we replace the word 'keyword' by the word 'feature' we have the same model that is used for many classification and categorization tasks. However, it must be added that in those latter systems the feature vectors are much shorter, not more than at most a few dozen features, whereas the number of keywords in the average documentbase runs into the tens of thousands. This makes it necessary to find ways and means to reduce the number of keyword features, either by selection or by condension.

Term weighting

One of the more interesting problems when creating such indices is how to decide on the relative importance of a word in a text as an indicator for the topicality of that text: term weighting. Various strategies to compute such weights have been proposed, among which the methods that are based on the frequency characteristics of words in texts have attracted much attention because of the relative ease with which they may be applied, while the performance in terms of precision and recall is reported to be among the best. Other methods of creating keyword or keyphrase-based document representations don't seem to score as high as the frequency based keyword extraction methods (see for instance the conclusions of [Keen, 1992] or [Cleverdon, 1991]). Among these alternatives for term weighting are heuristics such as the cue methods and the positional methods (for a concise survey of these and other methods see [Paice, 1990] or [Paijmans, 1994], [Paijmans, 1997]).

Apart from the weights in which the relation between every keyword-document combination is stored, there is a second group of weights that are attached to the keywords but that do not describe individual keyword-document relations. A good example of these second class of weights is the so-called discrimination value, which is a measure for the variation in average document-document similarity that is observed when a keyword is omitted from or added to the index. Such weights are often used as a parameter to decide whether the keyword should occur at all in the index language.

Similarity functions

The other intrinsic part of the index language is the set of similarity functions that is used to compare queries and documents. The most widely used in production systems is the Boolean model, that produces a weak ordering in relevant and non-relevant documents according to the manipulation of sets with Boolean and proximity operators. In research environments the vector space model (VSM), sometimes called the 'bag of words'-model, has been (and is) very popular. In this model both documents and queries are represented as binary or weighted vectors of terms and comparisons between them return real-valued differences which enable the documents to be ranked on similarity to the query.

Still, after thirty years of research in document representations for information retrieval (or more than a hunderd years if we include Dewey and his disciples), the fact remains that the actual document representation only has a minor effect on the performance of the complete system [Croft, 1987], [Lewis, 1992]. The same is true for the similarity function that is used. Although Noreault, McGill and Koll in their report on a variety of ranking strategies by the weighting of keywords and the application of similarity functions ([Noreault et al., 1981]) found an improvement of 20% over random ranking, they also concluded that 'While some algorithms were bad, most produced very similar results. No algorithm or approach distinguished itself as being greatly superior to others'.

7.2 Information retrieval as a classification task

7.2.1 Classification and categorization

We already noted that a weak ranking of documents in those that are relevant to a query and those that are not, is the same as an one-class classification. Indeed, many of the tools used in text classification are similar to those in vector-based information retrieval and both rely on the comparison between the document vectors and a query- or example-vector.

The difference between the two is that in IR the query is the translation of an information need, whereas in *text classification* an equivalent of the query is created by analysing the texts that belong to the target class and the texts that do not. When this involves pre-existing categories the term *text categorization* is preferred.

If there is a fundamental difference between IR systems and Text Classification systems, it is the learning component that is inherent in classification systems. Although retrieval implies a classification in relevant and non-relevant documents, these classes are not known in advance and there are no examples of the relevant documents available. The kind of classification or rather categorization systems that we are considering in this paper starts from the situation that examples from the classes are available and the classification of new documents then depends on the similarity between the new documents and the documents that already have been classified. This situation is known as 'supervised learning'.

7.2.2 Relevance feedback

There is a model in information retrieval that is very similar to this supervised learning classification model: the relevance feedback model and this relevance feedback, when applicable, is considered one of the most successfull techniques ([Croft and Das, 1990]). In this model, after an initial query, the results are presented to the user who then can select those documents that are most relevant to his information need (or a different method to obtain relevance estimates is used). New queries then are constructed taking these selected documents as starting point and this cycle is maintained until the information need is met.

The Rocchio formula for relevance feedback [Rocchio, 1971], for instance, takes an initial vector of features and presents this vector as a query to an IR system. The document vectors that are returned are divided in two classes, relevant and non-relevant and the formula then is applied to adjust the values of the query vector towards the values in the relevant set and away from the non-relevant set (see below for a more detailed description).

In this way it can be said that the system 'learns' about the properties of documents that are relevant for a particular information need. Other examples of the application of relevance feedback are given by [Chen, 1995], among which an interesting application of genetic algorithms.

7.2.3 Surface properties of text as features

Even if we take into account the similarities between classification and IR as mentioned above, we are confronted with the fact that information retrieval and text classification both are special cases in the quantity if not in the quality of the features. In the post-coordinate, derived indexing systems where words or word stems are indexed, the number of features initially runs in the tens of thousands. Moreover the document-feature matrix is sparse but nevertheless there is a high variety in the number of non-zero features in an instance.

The first question therefore is whether a document representation that essentially consists of the set of all words occurring in the document, is the best representation of the document to start with. However, there are no other obvious candidates. For if the ultimate goal of creating a document representation is to extract relevant concepts without being misled by irrelevant information, the starting point of such a procedure would be the natural language of the document under consideration, which implies the words contained in it. It would be impossible to collect information about the contents of a document without extracting the words from it; even complicated representations in frames or logical formulas, if they could be extracted without reading the actual words, would not make much sense if they are not instantiated by those words.

7.2.4 Selection and reduction

The logical next step is to reduce this great number of features. This can be done by selecting the most promising features or by a remapping of the features in a space of smaller dimensionality.

- Feature weighting. Perhaps the most elementary form of feature selection is the assignment of weights to the individual features (keywords) and dismissing words that do not reach a threshold. An obvious example is the ubiquitous list of stopwords. More sophisticated is the use of a discrimination value as described above or the computation of the *information gain* of the keyword. In this way the number of terms the database has to cope with can be greatly reduced while the inherent loss of information can be kept within bounds.

- Word-document weighting. A different method of weighting is when the individual *keyword - document* relations are expressed in a weight. such as the $tf.idf$. In such cases the term vector is not necessarily shortened, but outliers and spurious occurrences may be eliminated from the matrix or individual word-document combinations can be given extra emphasis.

- Singular Value Decomposition. In the two methods mentioned above, the features were selected according to some threshold or fitness function. However, the feature, once selected, remained intact. This is why we favour the expression 'feature selection' for this kind of operation. A different method of reducing the number of features is the re-mapping of the original features on a smaller number of new features. Here we prefer to use the expression 'feature reduction'. A method of feature reduction that has received much attention is latent semantic indexing [Deerwester et al., 1990]. This is brought about by applying singular value decomposition (SVD) to the original document-keyword matrix, creating a s-dimensional semantic space in which both documents and keywords can be mapped.

  If the relation between each keyword and each document is expressed in a $d : t$ matrix of weights (w), the application of SVD creates three new matrices; a $d : s$ matrix (W), a diagonal $s : s$ matrix S and a $s : t$ matrix T. d stands for the number of documents and t for the number of terms.

$$\begin{bmatrix} w_{0,0} \cdots\cdots w_{0,t} \\ \cdots \\ \cdots \\ \cdots \\ w_{d,0} \cdots\cdots w_{d,t} \end{bmatrix} =$$

$$\begin{bmatrix} W_{0,0} \cdots W_{0,s} \\ \cdots \\ \cdots \\ \cdots \\ W_{d,0} \cdots W_{d,s} \end{bmatrix} \begin{bmatrix} S_{0,0} \cdots \\ \ddots \\ \cdots S_{s,s} \end{bmatrix} \begin{bmatrix} T_{0,0} \cdots\cdots T_{0,t} \\ \cdots \\ \cdots \\ T_{s,0} \cdots\cdots T_{s,t} \end{bmatrix}$$

  The new dimensional space describes the co-occurence of the original keywords and the diagonal matrix S is ordered in such a way that the first columns describe strong co-occurence tendencies and that towards the end only spurious co-occurrences and weak relations occur. By keeping the n first singular values and zeroing out the others, a semantic space can be defined in which to compare douments, keywords or combinations of both.

7.2.5 Class representations

A level of abstraction may be imposed between the individual document representation and the application of the similarity function. This occurs when documents that belong to the same class are pooled and some single representation for such a set is constructed.

- The Rocchio algorithm. We already mentioned the Rocchio algorithm as an example of relevance feedback, that in its turn is very much like supervised learning. More in general it means that during training the document vectors are divided in two classes, relevant and non-relevant and the formula then is applied to adjust the values of the query vector towards the values in the relevant set and away from the non-relevant set.

$$w = \alpha w_1 + \beta \frac{\sum_{i \in R} x_i}{n_R} - \gamma \frac{\sum_{i \ni R} x_i}{n - n_R}$$

where α, β and γ are adjustment parameters, w_1 is the original weight of a keyword in the query vector and x_i the weights of that keyword in the relevant, c.q. non relevant documents. n is the number of documents in the database or in the returned set. In this way a 'target' representation of the relevant class is created and inclusion of new documents in one or the other class is done by comparing the document to this representation.

- Local dictionaries. In the section on word weighting, it was assumed that the reduction was applied to all classes and documents at a time. However, if the task is not an IR task but a classification task, it would be a logical approach to specialize the document representation for the particular classification task at hand. For example, in the work of [Apté et al., 1994a], [Apté et al., 1994b] a different index is created for each classification task; de term vector in each case consists of the n words with the highest frequencies in the example documents.

  Of course any of a number of possible weighting schemes can be selected: in our own experiments it was found that weighting according to the *atc* weight (a variation on the *tf.idf*, see also Table 7.8) performed better than the plain frequency.

- Genetic algorithms. GA's work by presenting the potential solutions for a problem in a bitstring. These bitstrings then are manipulated by cross-over and mutation, creating new solutions from the parts of promising older solutions. Central in this manipulation is the fitness function, which measures the relative performance of every solution. In Chen's example [Chen, 1995] the keyword vectors of the documents which were judged as relevant are fed through a GA that kept comparing every vector with the other vectors until an optimal 'common denominator' for that set of documents was found. This vector was then used as the new query. Chen reported the GA's as performing better than neural networks on the same task.

7.2.6 Similarity functions

The above describes a number of techniques to select the most promising keywords from the documents and/or remap them in a smaller set of features. The next step necessary for successful retrieval or classification is the application of one or more similarity functions to score the individual documents against the training examples or the query.

- Boolean set operations. The most general similarity function in production systems is the Boolean set operation that orders the documents in *relevant – not relevant* according to the occurrence of keywords in it. Although its practice is widely spread, its performance has been severely criticized by, among others, [Blair, 1996], [Blair and Maron, 1985]. Also attempts to replace the weak ordering by more elaborate ordering functions such as the application of fuzzy logic have not caused the all-over performance of the Boolean model to be improved.

- Rule induction. This family is characterized by their representation of acquired knowledge as decision trees. The systems are presented with a set of cases relevant to the classification task at hand and develop a decision tree guided by frequency information in the examples.

- Vector space comparisons. This model has received much attention, because it offers straightforward way to compare keyword vectors and to rank such vectors on similarity to a target vector.

7.3 Methodology and experiments

7.3.1 Evaluation issues

Following usance in information retrieval, it has become tradition also in classification systems research to use the precison ratio and the recall ratio as measures for the performance of a two-class document classification system.

Given a universe of documents $\{A, B, C, D\}$. A, B are the documents that are classified as belonging to class X; C, D its complement (all documents that classified as not-X).

In $\{A, B\}$ (the documents classified as X) A is the set of documents that in fact do belong to class X; B the set of documents that were erroneously classified as X.

In $\{C, D\}$ (classified as not-X) C is the set of documents belonging to X that were not recognized by the system; D is the set of not-X documents that were correctly classified as not-X.

The Precision ratio is $A/A + B$ and the Recall ratio is $A/A + C$. A third measure is the Fallout $B/B + D$.

The breakeven point

When a single measure is needed for the performance of retrieval or classification experiments, the breakeven point may be calculated from a number of precision and recall scores for the same experiment.

The breakeven point is generally arrived at by linear intrapolation. It is defined as the point on a precision-recall curve that has the same value for precision and recall. Thus if the two points that bracket this point are known to be $< fp, fr >$ and $< sp, sr >$, the breakeven point is $< b, b >$, where

$$b = \frac{sr * fp - fr * sp}{sr - fr + fp - sp}$$

If $fp = fr$ or $sp = sr$, then the breakeven point is on the curve, not just bracketed by two points.

A drawback of this method is that the algorithm under consideration must have some parameter that governs the 'willingness' of the algorithm to assign categories to documents or, in other words, govern the trade-off between precision and recall.

The breakeven point is not without its detractors. In a personal communication[2], Lewis, who himself first published this measure, stated that it had serious shortcomings:

1. Interpolation gives values not achievable by the system. Although plotting of recall against precision typically gives a smooth, monotonically decreasing curve, more precise plots for single classes display a less smooth and not even monotonic curve.

2. Recall=Precision is not a desirable or informative target. A system tuned for an optimal breakeven point is in a rather extreme state, where precision and recall are at their minimum and this does not necessarily reflect the preferences of the user.

3. An average over diverse categories is of dubious value.

Be this as it may be, there is an obvious relation between a breakeven point and the performance of a classification or retrieval system and where it may not be much better than other measures, it certainly is not worse for comparison purposes.

The harmonic mean

Sometimes the two outcomes of precision P and recall R are combined in one single figure by taking the harmonic mean F of the two:

$$F = \frac{2}{\frac{1}{P} + \frac{1}{R}}$$

The magnitude of F varies from 0, when no relevant documents are retrieved, to 1, when all and only the relevant documents are retrieved. Moreover F is strongly weighted towards the lower of the two values P and R; therefore this measure can only be high when both P and R are high. [Shaw et al., 1997b] use the harmonic mean as peformance measure in the computation of the base line performance for IR (see below).

[2] Also in the mailinglist dlbeta@research.att.com of 11 Sept. 1997

	Macro eval.		Micro eval.		
	Pr.	Rec.	Pr.	Rec.	Accuracy
avg. nnn	0.344	0.144	0.672	0.088	0.42
avg. atc	0.499	0.229	0.817	0.550	0.47
avg. glob.atc	0.683	0.457	0.834	0.664	0.67
avg. 10-length	0.446	0.222	0.780	0.386	0.47
avg. 20-length	0.503	0.260	0.778	0.430	0.51
avg. 66-length	0.578	0.348	0.766	0.487	0.60
avg. bin-vectors	0.496	0.290	0.768	0.457	0.57
avg. int-vectors	0.510	0.264	0.799	0.426	0.51
avg. real-vectors	0.521	0.276	0.757	0.420	0.51

Table 7.1: Precision and recall for classification with C4.5

Accuracy

Precision and recall are measures from the discipline of information retrieval and therefore they focus on only one of the two classes: the class *relevant*. There is no interest in measuring the degree in which the system does *not* retrieve irrelevant instances. In classification experiments we are interested in both sides of the coin and that calls for different measures.

[Weiss and Kulikowski, 1991] use a basic accuracy measure that is simply defined as the ratio of correctly assigned items over the total of items. This, of course, is very much like the recall, but now for all classes combined.

When we applied this measure to our data we found for all experiments an average accuracy of between 95% and 99%, which is obviously not conform the actual performance. Further inspection showed that there was a big difference between scoring on positive and on negative instances of the two classes: the negative instances (by far the biggest class) displaying an accuracy of almost 100% whereas the positive class had an accuracy of 37% in the rule induction (C4.5) experiments from Table 7.1 and 19% in the centroid/cosine experiments (VSM) from Table 7.2.

Weiss and Kulikowsky suggest adjustement of the accuracy measure by the

experiment	macro	micro	accuracy	prec.	recall
avg. nnn	0.241	0.077	0.76	0.10	0.66
avg. atc	0.316	0.503	0.79	0.09	0.71
avg. glob.atc	0.497	0.562	0.90	0.21	0.85
avg. 10-length	0.357	0.361	0.79	0.15	0.69
avg. 20-length	0.366	0.388	0.81	0.13	0.74
avg. 66-length	0.410	0.427	0.84	0.13	0.79
avg. bin-vectors	0.366	0.380	0.83	0.11	0.78
avg. int-vectors	0.334	0.375	0.78	0.11	0.68
avg. real-vectors	0.432	0.422	0.83	0.22	0.73

Table 7.2: Breakeven points and accuracy with associated precision and recall for VSM

introduction of cost factors, which essentially is a way of biasing decisions in different directions, as if there were more or fewer instances in a given class. We therefore used a bias that was proportional to the relative size of the two classes so that the net effect was of making both classes equal in size.

In the VSM experiments a threshold was varied over the values 0.10 - 0.90 and the accuracy was computed for all nine steps, keeping the step with the highest accuracy. The optimal threshold varied with the cost, so that the individual values for the accuracy of positive and negative instances and for precision and recall varied too. As the C4.5 did not use a threshold, these values remained constant.

Micro- and macro evaluation

The ratios mentioned above are computed over a single classification action. When the results of several such actions have to be combined in single measures for recall and precision the averages may be computed in two ways, called the micro- and the macro evaluation (see [Fuhr, 1995] for a discussion of these concepts).

In the macro evaluation the individual values for precision or recall are computed first and afterwards averaged. This can cause problems when one or more classification experiments in the series yield no positive results, but has as advantage that every individual classification attempt has the same weight in the final outcome, i.e. that the result is not biased towards the big classes. The macro evaluation for the precision is computed as:

$$p_{macro} = \frac{1}{N}\sum_{i=1}^{n}\frac{\| REL_i \cap RET_i \|}{\| RET_i \|}$$

The other way in which to compute an all-over measure for the performance of a classification system is to first compute the averages of the components above and below and divide afterwards. This circumvents the problem of empty sets and causes every individual document to have an equal influence on the result. The micro evaluation for precision is computed as:

$$p_{micro} = \frac{\sum_{i=1}^{n} \| REL_i \cap RET_i \|}{\sum_{i=1}^{n} \| RET_i \|}$$

The literature on classification systems prefers the micro evaluation but one must be aware of the fact that a few big classes can bias the outcome considerably. Therefore the differences between the two types of evaluation also may be considerable.

7.3.2 A baseline

The baseline, or low performance standard of a IR system is its performance if the selection of retrieved records happened on no other criterion than pure chance. This is equivalent to the blind selection of balls from an urn that have

experiment	macro	micro
nnn.10	0.135	0.063
nnn.10.int	0.130	0.058
nnn.10.real	0.134	0.061
nnn.20	0.156	0.069
nnn.20.int	0.136	0.055
nnn.20.real	0.160	0.060
nnn.66	0.149	0.064
nnn.66.int	0.137	0.045
nnn.66.real	0.167	0.065
atc.10	0.276	0.499
atc.10.int	0.255	0.466
atc.10.real	0.253	0.466
atc.20	0.292	0.520
atc.20.int	0.258	0.503
atc.20.real	0.252	0.499
atc.66	0.243	0.394
atc.66.int	0.296	0.389
atc.66.real	0.209	0.377
glob.atc.10	0.481	0.659
glob.atc.10.int	0.428	0.621
glob.atc.10.real	0.452	0.635
glob.atc.20	0.448	0.634
glob.atc.20.int	0.389	0.604
glob.atc.20.real	0.406	0.607
glob.atc.66	0.326	0.472
glob.atc.66.int	0.332	0.478
glob.atc.66.real	0.313	0.471
avg. nnn	0.145	0.060
avg. atc	0.259	0.457
avg. glob.atc	0.386	0.556
avg. 10-length	0.283	0.392
avg. 20-length	0.277	0.395
avg. 66-length	0.241	0.306
avg. bin-vectors	0.278	0.375
avg. int-vectors	0.262	0.358
avg. real-vectors	0.261	0.360

Table 7.3: Base line performance of local dictionaries with cosine

two colours: white and black. The number of white and black balls equals the number of documents in the database; the white balls signify documents that are relevant to a query. As [Shaw et al., 1997b] state: 'The low performance standard is based on identifying the highest level of retrieval effectiveness an exceedingly patient searcher can produce by such an random process'.

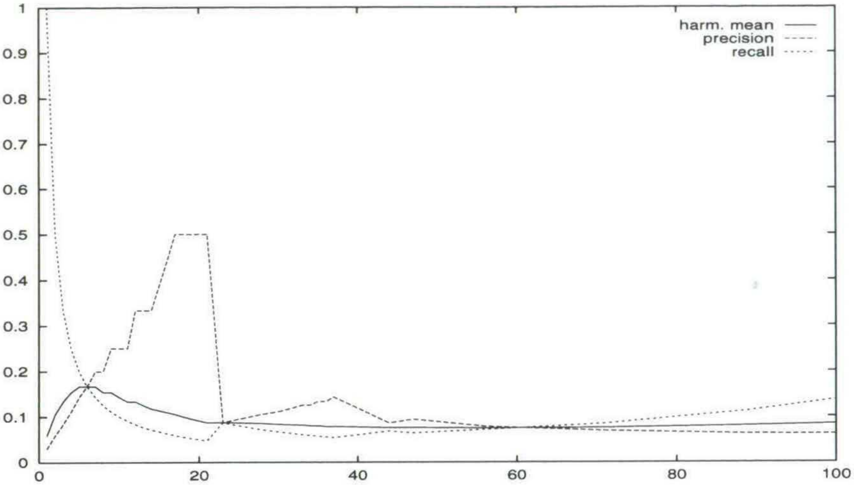


Figure 7.1: Hypergeometric distribution for 0-100 relevant records.

For a collection with N documents and R relevant documents, the probability of retrieving exactly r relevant documents by selecting randomly n documents from the database is given by the hypergeometric distribution:

$$Pr(N,R,n,r) = \frac{\begin{pmatrix} R \\ r \end{pmatrix}\begin{pmatrix} N-R \\ n-r \end{pmatrix}}{\begin{pmatrix} N \\ n \end{pmatrix}}$$

where the bracketed expressions on the right side represent the binomial coefficients. Now the number of documents N in the database and the number of relevant documents R for each class are known. Therefore it is possible to vary the number of records retrieved n and the number of relevant records in that set r to find the r and n for a given probability threshold.

Figure 7.1 displays the low performance standard for the Reuters test collection of 3299 documents for the threshold of 0.01 and associated precision and recall levels for queries with 1-100 relevant documents in the database. We observe a peak in the effectiveness (harmonic mean) of 0.16 when five to seven documents would be relevant for the query and a subsequent decrease to an effectiveness of 0.075 by 56 relevant records. The performance then rises slowly to an effectiveness of 0.36 for 719 relevant records and 0.50 for 1087 records (outside the graph).

As we know the number of classes in our corpus and for every class the number of relevant records in the test database, we can state that the mean of positive instances to be expected in every category is about 40. About half of the classes has less than 10 positive instances. Thus the low performance standard of this collection would be an effectiveness of 0.07 all over or 0.15 for the 43 categories with less than ten positives. The corresponding precision and recall can be gauged from the figure.

7.3.3 The database

For our experiments we used the Reuters-21578 collection[3]. From this we made a selection to use as train- and test collections, trying to keep as close as possible to the train- and test-selections as used by Apté, Damereau and Weiss (the so-called MODAPTE split. For details we refer to the README included in Reuters-21578). This gave a training set of 9603 documents and a test set of 3299 documents.

A great advantage of the Reuters data is that they have been categorized in a closely observed setting, therefore the assigned categories display a degree of agreement between man and machine that is much higher than in other experiments that have been published [Lewis, 1992]. In the CONSTRUE system, developed by Carnegie Group Inc. for Reuters, precision and recall rates of 92% and 89% respectively have been reported ([Hayes and Weinstein, 1990]), but one should take in account the bias caused by micro-averaging the results: in this score 10% of the categories accounted for 59.3% of the category assignments on the CONSTRUE test set.

A different an perhaps more serious uncertainty connected with this particular corpus is the fact that the categories were developed in close cooperation with research in mechanical categorization and that definities of categories were adapted to make this mechanical categorization easier.

Much care had been taken by the composers of the database during the preparation of the training and test sets to ensure an equal class distribution over both sets, so that, for instance, documents classified as belonging to the class 'rubber' or 'wheat' occurred in equal proportions in both sets. However, because Lewis and other investigators wanted to keep as close as possible to an operational setting, they used a chronological split of the documents so that the training set appeared in time before the test set. For our experiments we will use the same sets, or at least the same division between them, to facilitate comparisons with other research.

On both test and training sets the features were obtained from all fields from "TITLE" downwards, but with exclusion of "AUTHOR" and "DATELINE". General statistics, such as *tf.idf*, were computed over all documents from the complete training set.

[3] Avaliable as reuters21578.tar.gz from http://www.research.att.com/~ lewis.

nnn.10	bhp tvx war coin compan consolid norgold nbh asamer
atc.10	feet grad corp ton resourc reserv assay expect produc
glob.atc.10	gold ounc mine ton ore silf grad assay coin

Table 7.4: Nine highest scoring features for gold according to weighting method

Programs

The programs we used for the experiments were partly written by others, partly by ourselves. In the former category are SMART, written by Salton, Buckley and Voorhees, C4.5 by Quinlan [Quinlan, 1993] and Genesis by Grefenstette [Grefenstette, 1990]. We also used WEKA3[4] as a frontend for C4.5 and other ML programs. Also the program 'svdinterface' by Schuetze, itself an adaptation of programs from the SVDPACK[5].

In the latter category falls the program suite "Paai's text utilities"[6]: a series of Unix scripts and utilities that perform tasks such as computing similarities between vectors, discrimination values, average sentence weights, etcetera.

All indexing was done by SMART, using the list with common words, or stopwords' that comes with this program. This list was some hundred words longer that the list used by Apté. We also used the built-in stemmer of SMART to reduce the number of concepts.

7.3.4 The experiments

The experiments we did on the Reuters database were aimed at obtaining a good impression of the performance of various combinations of features and similarity functions, with an emphasis on local dictionaries. To this goal, we varied three methods of term weighting with three different vector lengths and three measures. Taking Table 7.3 as an example, we first observe the three main groups of variations with the identifiers *nnn*, *atc* and *glob.atc*. This refers to the method that was used to weigh the individual words in the positive training documents to obtain the local dictionaries. *nnn* refers to plain term frequency, identical to Apté, Damereau and Weiss. The atc is a much used variant of the tf.idf measure (for a detailed discussion of these and similar techniques see e.g. [Salton and McGill, 1983] or [Salton, 1989]).

As the document frequency (i.e. the number of documents in the database in which the term occurs) is one of the factors in the computation of the *atc*, it makes a difference if the weight is computed over the positive examples only or over the complete training database. Therefore the *atc* in the table refers to the weight computed over the positive examples only and the *glob.atc* is the same measure, but now computed over all documents in the trainingset. In Table 7.4 we present as an example the ten word stems that scored highest in

[4] Available from http://www.cs.waikato.ac.nz/ ml

[5] Available from http://www.netlib.org/svdpack

[6] Available from from http://pi0959.kub.nl:2080/Paai/Public

each of the three methods for the class 'gold': note that no words occur in all three lists, that only one word ('coin') in the *nnn* group overlaps with *glob.atc* and that two ('ton' and 'assay') occur in both *atc* and *glob.atc*.

The next variation, indicated by 10, 20 and 66, refers to the length of the vector, i.e. the n topranking keywords according each of the three scoring methods described above.

Finally the final notation of the weight is indicated. If no suffix is added after the length-indicator it means that binary weights are used - one for occurrence, zero for no occurrence. The suffix *.int* means that the words are scored according to frequency and the suffix *.real* indicates that the atc-weight is used. This of course is the global atc-weight, because all documents have to be indexed.

To get a 'base line' performance of these words as classifiers, we created for each of the documents in the testset corresponding vectors and compared the local dictionaries to the documents using the cosine as similarity function. In Table 7.3 the performance of these local dictionaries as queries, c.q. classification rules is displayed, using the cosine as similarity function. As expected, the words selected by the global atc method scored best, but the fact that the vectors of only ten or twenty words long score better than the longer vectors comes as a surprise. We also see that the use of word weights in the vector does not add to the performance but instead decreases it.

Rocchio

To be able to put the other outcomes into perspective, we then used the well-published Rocchio's algorithm as a classifier, which here is implemented as the centroid or mean of the positive examples. As a similarity function we used the cosine, which is often used in this kind of experiments:

$$Sim(d_j, d_k) = \frac{\sum_{i=1}^{m} d_{ji} \bullet d_{ki}}{\sqrt{\sum_{i=1}^{m} d_{ji}^2 \bullet \sum_{i=1}^{m} d_{ki}^2}}$$

where d_j and d_k are document (or document and query) vectors and m is the length of the vecor (number of features). Noreault and his collegues collected and tested a great number of these functions ([Noreault et al., 1981]). Of the 24 functions they presented in their article the best scoring was the Pearson Product Moment Correlation that is rather similar to the cosine (the cosine itself was not in their list) but in our initial experiments this last measure scored consistently lower than the plain cosine and in later experiments it was discarded.

A document would be judged as belonging to the class when the similarity between the centroid and a vector was above a certain threshold. By varying this threshold we could trade precision for recall and thus compute the breakeven point. In Table 7.2 the results are shown for both the normal cosine and the Pearson variation as the similarity functions.

Here we do observe that the performance increases with the vector length.

Learning method	Dictionary	Text Representation	Breakeven (%)
Optimized Rule Induction	local	Frequency+Titles	80.5
		Frequency	78.9
		Boolean	78.5
	universal	Frequency	78.0
		Boolean	75.5
Decision Tree			67.0
Probabilistic Bayes			65.0

Table 7.5: Breakeven points for Reuters English Data

Also, the real-weighted vectors (*tf.idf*) do better than the integer (frequency-weighted) vectors.

Rule induction

The next series of experiments continued in the same vein as the work of Apté, Damereau and Weiss, but with a different Machine Learning program. Where these other authors used SWAP-I, we used C4.5. Also we greatly varied the way in which the local dictionaries were obtained, as described in detail above, because we wanted to compare different strategies of constructing those local dictionaries. The results are in Table 7.1.

Apté and his co-researchers used in their experiments both 'local dictionaries' and entropy-based feature selection methods based on the complete or 'universal' dictionary (local dictionaries here are dictionaries that are composed from documents in the training set that were assigned to a certain category). The features were stored either as Boolean values or as the frequency of the word in the document. On the evidence of their experiments they concluded that the local dictionaries scored better than the universal dictionary by two or three percent and the same was true for the frequency-weighted features compared to the Boolean features (see Table 7.5, taken from [Apté et al., 1994a]).

Our first step consisted of the creation of local dictionaries for each class under consideration. This was where two of the three variations were introduced: the vector length and the ranking method. Apté and his collegues used the 150 most frequent words in every local dictionary and subtracted stopwords from a brief universal list of stopwords (427 words) from them, keeping 80-100 words from the original list. Our experiments introduced the variations as described. Although we did some experiments with a vector length of 100 features, we found that the improvements over shorter vectors were marginal at best. We then scored the words in the local dictionaries according to occurence (binary), frequency (integer) and weight (real); the results are displayed in Table 7.1.

Genetic algorithms

We already mentioned the experiments of Chen in using a genetic algorithm as learning algorithm for relevance feedback [Chen, 1995]. Given a initial set of

	macro	micro
avg. nnn	missing	missing
avg. atc	0.278	0.486
avg. glob.atc	0.381	0.560
avg. 10-length	0.359	0.548
avg. 20-length	0.341	0.536
avg. 66-length	0.304	0.489
avg. bin-vectors	0.312	0.523
avg. int-vectors	0.285	0.492
avg. real-vectors	0.407	0.558

Table 7.6: Breakeven points for using a GA-produced query.

retrieved documents, he selected the positive examples as the starting point of a population, taking the document vector as the chromosome and the individual keywords as gens. The fitness function that he applied was the similarity of each document vector to all other vectors from the selection using Jaccard's coefficient.

We reasoned that in the same vein GA's could be applied to find the optimal weights for classes. Therefore we ran the positive examples from the training set through Grefenstette's Genesis program, using the same similarity and fitness-function as Chen had used, obtaining an optimalized vector. This then was compared with all documents from the test set similar to the centroid experiments described above, using again the cosine as similarity function. The results are displayed in Table 7.6. Please note that the averages are computed without nnn-values.

Singular value decomposition

The last series of experiments were done in reducing the feature set using SVD as described above. Again we used the index prepared by Smart, with usage of stopwords and stemming and as weights the atc-weights computed over the training set and test set together, which gave a total of 22213 different word stems. We decided to lump training and test set together, again because of the document frequency that was needed in the atc-weight. After that both sets again were separated in the matrices Train = (TrainDoc : Terms) and Test = (TestDoc : terms).

The Singular Value Decomposition then was applied to the trainingset, obtaining three new matrices from the training matrix: $D = (TrDoc : Sing)$, $S = (Sing : Sing)$ and $T = (Sing : Term)$. The new training set TrD then was created by multiplying D with S and a new test set by multiplying Te with T , obtaining $TeD = (TeDoc : Sing)$.

From the training set TrD the centroids for the classes were computed and compared to the test set TeD. The results are displayed in Table 7.7.

length	records	classes	macro	micro
50	3708	84	0.383	0.662
100	3708	84	0.422	0.656
150	3708	84	0.450	0.661
200	3707	83	0.462	0.655
50	3748	93	0.346	0.655
100	3748	93	0.381	0.649
150	3748	93	0.407	0.654
200	3748	93	0.413	0.648

Table 7.7: Results for classification on SVD with cosine

7.4 Conclusions

The experiments were meant to give a common ground of comparing several variations of document representations and classification strategies. Although perhaps not all strategies have been tuned to top performance and the comparison of performance measures remains problematical, some conclusions may be drawn.

7.4.1 The local dictionaries

To begin with it is clear from the tables that for the local dictionaries (Tables 7.3, 7.2, 7.1 and 7.6) the words chosen by ranking the global *atc* values performed better than local *atc* values and dramatically better than the *nnn* values. The difference between global *atc* and local *atc* would be 20% of the lower (atc) value; the difference between *nnn* and local *atc* would be 700% or more (reckoned over the micro-evaluation). This was true regardless the actual strategy applied, either Rocchio, C4.5 or GA.

The differences between shorter and longer vectors were less outspoken. In half of the experiments the shorter vectors outperformed the longer ones (for instance in Tables 7.3 and 7.6).

Also the performance of binary weights as compared with frequencies (int) or atc-values (real) was not so clear as with the *nnn*, *atc* and *glob.atc* experiments. Here as with the vector lengths the real-weighted vectors sometimes scored better, sometimes worse, so that no clear conclusion could be drawn from the results.

The overall conclusions to be drawn from these results is that local dictionaries are best constructed using as a ranking measure a weight that contains word frequency information from the *complete* database as the *glob.atc* does, rather than word frequency information from the class examples only as the *atc* and *nnn* do. The length of the vector or the actual word weights in the vector have less impact on the result.

7.4.2 The strategies

When we try to compare the strategies, we are of course faced with the problem that the separate precision and recall values that were returned by the C4.5 experiments cannot be compared directly with the breakeven points that are used to score the other experiments. A very rough comparison can be made when we assume the breakeven point to lie somewhere between precision and recall. This does not conflict with the results of Apté, Damereau and Weiss and confirm that rule induction performs best (Table 7.1).

The Rocchio centroid and the GA score on average in the 50% range with a peak for Rocchio in *glob.atc.10.int* (0.628) and for the GA in *glob.atc.10.real* (0.633). But as the fitness function that we applied in the GA (the similarity between the vectors of the population) causes the vectors to home in on something very much like the centroid this would be expected.

But when we compute the accuracy for the VSM comparisons of Table 7.2 and the rule induction classification of Table 7.1 we see a peculiar tendency. In both tables the colums that contain the accuracy, display a good correlation with the corresponding recall column but the C4.5 table has an accuracy and a recall that is approximately half that of the cosine comparisons of Table 7.2 (taking the *micro evaluation*). However, the precision for both tables is drastically different, the cosine table having a very low precision and the C4.5 table a very high one.

The SVD experiments also stand on their own, because no local dictionaries were used. In Table 7.7 we see the average score, both with zero results taken in account (lower half) and omitted from the averages (upper half of the table). With a highest micro-evaluation of 0.662 this method scores reasonably well, although perhaps not so well as expected (compare [Dumais, 1994]). The only difference with the Dumais experiment seems to be that here not the *atc* but the *ltc* variant of the *tf.idf* was used (see Table 7.8).

7.4.3 Further work

The experiments in this article have been aimed at creating a general framework in which documents could be categorized by several strategies, but under similar circumstances. A major problem was and still is finding a measure for the performance that can be applied to the outcome of any strategy. So far we have not been able to reconcile the single-threshold figures of, e.g. C4.5 with the varying-threshold figures of the Rocchio type comparisons and this should be the next problem to tackle.

```
The value of the parameter is a 3 character code (eg "atc"):
    First char gives the term-freq procedure to be used
    Second char gives the inverted-doc-freq procedure to be used.
    Third char gives the normalization procedure to be used.

There are three possible conversion that can be performed on each vector:
    1. Normalize term-freq component - most often the tf component is
                  altered by dividing by the max tf in the vector
    2. Alter the doc weight, possibly based on collection freq info.
                  Note that this is done individually on each term
    3. Normalize the entire subvector - most often to "sum of squares" of
                  terms = 1. Alternative is sum of terms = 1

Weighting schemes and the desired weight-type parameter
Parameters are specified by the first character of the incoming string
   1.  "none"     : new-tf = tf
                    No conversion to be done  1 <= new-tf
       "binary"   : new-tf = 1
       "max-norm" : new-tf = tf / max-tf
                    divide each term by max in vector  0 < new-tf < 1.0
       "aug-norm" : new-tf = 0.5 + 0.5 * (tf / max-tf)
                    augmented normalized tf.  0.5 < new-tf <= 1.0
       "square"   : new-tf = tf * tf
       "log"      : new-tf = ln (tf) + 1.0

   2.  "none"     : new-wt = new-tf
                    No conversion is to be done
       "tfidf"    : new-wt = new-tf * log (num-docs/coll-freq-of-term)
                    Usual tfidf weight (Note: Pure idf if new-tf = 1)
       "prob"     : new-wt = new-tf * log ((num-docs - coll-freq)
                                                          / coll-freq))
                    Straight probabilistic weighting scheme
       "freq"     : new-wt = new-tf / n
       "squared"  : new-wt = new-tf * log(num-docs/coll-freq-of-term)**2

   3.  "none"     : norm-weight = new-wt
                    No normalization done
       "sum"      : divide each new-wt by sum of new-wts in vector
       "cosine"   : divide each new-wt by sqrt (sum of(new-wts squared))
                    This is the usual cosine normalization [...]
       "fourth"   : divide each new-wt by sum of (new-wts ** 4)
       "max"      : divide each new-wt by max new-wt in vector
```

Table 7.8: Variations on the computation of the tf.idf

Chapter 8

Word repetition as a discriminator for authorship attribution[1]

Abstract: *This paper describes some of experiments with the automated attribution of authorship. Lexical cohesion in combination with machine learning techniques are used as a method to compare texts of different authors. A methodology is described to create 'stylistic fingerprints'.*

8.1 Introduction

The attempts at automated authorship attribution described here are a secondary result of a line of research that is aimed at the identification of information-rich passages in texts: the so-called 'gravity wells of meaning' [Paijmans, 1994], [Paijmans, 1997]. The hypothesis underlying these 'gravity wells' is that passages in texts not only differ in content or topicality, but also in the degree to which that content is emphasized: the 'gravity' of the passage. Identification of such passages, then, should lead to the construction of information-rich document surrogates that in turn may serve as nuclei for information retrieval activities.

Following the example of earlier research by [Hearst and Plaunt, 1993], who used lexical cohesion as a discriminator for topical differences we included in our experiments a number of features that quantify various measures of text cohesion (see also [Morris and Hirst, 1991]). While it is not yet clear whether such features can be used to measure the *gravity* of passages as meant above, they presented themselves as potential factors in the recognition of *style*. We decided therefore to apply the tools that we had collected to the problem of authorship attribution, keeping in mind that, as [Burrows, 1992] observes, *the*

[1] This chapter appeared under the title "Discriminators for Authorship Attribution" in the South African Computer Journal, vol.23 no.7, July 1999. p. 30-41 ([Paijmans, 1999]). The bibliography entries have been collected in the bibliography at the end of this book.

manner in which stylistic differentiae are interlocked enables them to register even on defective instruments.

8.2 Authorship attribution

8.2.1 Some notes on terminology

First a few notes on terminology: it should mostly be clear from the context when we mean the author of a disputed text or the author in the sense of a scholar or scientist whom we cite for some reason or another. When doubt can arise, we will sometimes use the term 'writer' when we refer to the author of a disputed text. We will also use the term *target author* when we try to establish the authorship of a known author and the term *target text* for a corpus of positively attributed texts of that author. In the same way, we will use *control authors* and *control texts* as the complement of target author and target texts, i.e. authors that are positively identified as *not* being the target author and texts that are positively *not* written by the target author.

Methods by which attribution of texts to authors may be attempted, should be considered in the broader perspective of the analysis of style in general. In this context we may define 'style' as the various ways in which an author can allow himself freedom of expression inside the more or less fixed structure of rules and conventions that are necessary for transmitting a written message. Also, we assume that most of these are measurable; that is, we will not concern ourselves with variations that cannot be objectively identified and measured.

The three obvious dimensions in which texts can differ, then, are those of 'genre', 'content' and style. We briefly consider each of these in turn.

8.2.2 Genre classification

Examples of attempts at text classification are the work of [Pieper, 1979] for German texts and [Biber, 1989] for English texts. We mention these authors because they both use statistical methods to identify the types or genres in the respective languages. However, they approach their typologies from opposite directions. The earlier work, by Pieper, first forms hypotheses on a number of genres in the German language, called *clines* to emphasize their gradual transition from one class into another. She then tries to identify them by a multivariate analysis of linguistic characteristics, such as the ratio of nouns or finite verbs. Biber, on the other hand, starts out by defining dimensions of difference in terms of linguistic characteristics and uses objective, statistical methods (factor analysis) to create groups of texts that are maximally different on all dimensions. He introduces the word *register* for such a group [Biber, 1993].

Where this research is aimed at the creation of classification systems in which complete texts may be positioned, the next step is to identify properties of text that may lead to a classification system of *parts* of the text. Again

	Unstemmed	Stemmed
Scheme	Correct-%	Correct-%
Human-average	89	not perf
TFIDF	84	85
PPMC	64	65
GZIP	47	52
COMPRESS	23	26
ZEROWORD	52	45
FIRSTWORD	48	41
WORDAP	61	65
...		
C4.5/10c/b/lR	68	not perf
...		

Table 8.1: Classification results on stemmed and unstemmed articles (from Littin, 1995).

several researchers from many different disciplines have applied themselves to this task, e.g. [Kieras, 1985], [Dijk, 1980] and others.

8.2.3 Content analysis

A rather different classification of texts is that accordig to content: aptly called 'content analysis' [Krippendorf, 1980]. Note that the word 'content' in this context not only refers to what the text is about, but also to emotional, rhetoric or other categories. For instance, the German sociologist [Ertel, 1976] classified texts according to dogmatism by counting words like 'always', 'whenever' or 'never', which indicate a dogmatic state of mind in the writer, or 'often', 'sometimes' and 'occasionally' as indicators of a more tentative state of mind.

A different approach of content classification is found in the field of information retrieval. [Littin, 1995] describes an application of machine learning to text categorization. A number of schemes, including human judgment, tf.idf weighting, Quinlans C4.5 and even the standard Unix compression utilities are used to classify 1600 articles from ten Usenet newsgroups (tf.idf and C4.5 are explained later). As expected, humans performed best, categorizing 89% of the articles. *tf.idf* came in a very good second (84%). The best C4.5 variation scored 65% and the gzip compression utility scored a surprising 47%. Table 8.1 shows part of the results.

8.2.4 Comparing authors

In this section a short survey is given of work pertinent to the problems of author recognition. Two main approaches may be distinguished to the attribution of texts to authors. The first is almost as old as literary criticism itself and is based on literary or historical evidence, i.e. other evidence than that furnished by quantitative properties of the texts. Of course these literary and historic properties are of central importance for the scholar and the connaisseur, but they often are far from unambiguous. Therefore, additional proof is sought in the quantitative or statistical study of the texts under consideration; an activity that is sometimes called 'stylometry'. *The stylometrist looks for a unit of counting which accurately translates the 'style' of the text, where we may define 'style' as a set of measurable patterns which may be unique to an author* [Holmes, 1994]). More detailed surveys of the state of the art may be found here and in [Forsyth and Holmes, 1996, Forsyth and Holmes, 1995].

The beginning of this discipline is to be found in the last century, in the work of de Morgan and Mendenhall. These scholars concentrated on features like sentence length and word frequency and such measures are still used in modern stylometry. Nevertheless the methodology of considering an author's writings as random samples of his/her own fixed frequency distribution of word-lengths is nowadays considered unreliable, when works of various literary genres or different eras are compared.

After the second world war the statistical approach received a tremendous boost by the development of the modern computer, not only because of the greatly improved methods to compute statistics, but in more recent years also because of the ready availability of huge machine-readable corpora and sophisticated tools for analysing texts such as stemmers, taggers, and weighted indexing systems.

In 1962 [Ellegard, 1962] already used the frequency of function words and synonym pairs, but perhaps the most influential and certainly the most cited work is the study by Wallace and Mosteller of 1964 ([Mosteller and Wallace, 1964]) on the *Federalist papers* (see also [Francis, 1966]), where they proposed to attibute texts on grounds of synonym preferences of the potential authors. As synonym-pairs were few in number in the papers under consideration, in the end they selected certain function words and compared the frequencies with which these were used by the two authors.

A somewhat different approach was adopted by scholars, such as Tallentire, Baker and several others. They also worked on the assumption that every writer favours some words more than others, and that this preference can be detected in differences in the frequency profile of the word types used. The type-token ratio presented itself as a potential measure [Tallentire, 1976], [Baker, 1988], this ratio has the drawback that it is not stable over samples of different sizes as the number of tokens in increasingly large samples will show a growth-rate that is different from that of the type-dictionary. As it is generally possible to use samples with a fixed size this will rarely be a problem.

Syntactic categories are more difficult to identify than the lexical features on which most of the research mentioned above is based. [Yngve, 1961] proposed to use the depth of nesting of syntactic categories as a measure, but his suggestion has not been followed by later researchers. However, more recently Dutch researchers have looked into the discriminatory potential of syntactic rewrite rules for authorship attribution, with promising results [Baayen et al., 1996].

A rather different approach was adopted by [Matthews and Merriam, 1994] and [Matthews and Merriam, 1997], where a neural network was used.

Hence a brief and not exhaustive inventory of features that have been tried as discriminators is as follows:

- general statistics, such as average word and sentence length
- synonym preferences
- distribution of part of speech categories like nouns, verbs, or articles
- conditional clauses and phrases
- type-token ratio
- depth of nesting in sentences
- syntactic categories

Most experiments mentioned so far have been conducted in situations where the group of target authors was very small, typically one or two. As already noted by [Forsyth and Holmes, 1996], there are very few records of attempts to apply different methods on the same corpus.

8.3 Stylistic fingerprints

The *modus operandi* of attributing authorship on the basis of the texts he (or she) has written, may be contrasted with that of indentifying an individual on the basis of his fingerprints. We can hope for the emergence of a single pattern, lexical or otherwise, that uniquely binds every author to his texts, analoguous to the use of fingerprints in forensic proceedings. However, it is highly improbable that such a pattern can be found. A writer can deliberately change his style in an attempt to remain anonymous, to mimic a different writer, or for any other reason, but the changing of one's fingerprints is less lightly undertaken. Most authors on the subject prefer the use of patterns that are, as much as possible, beyond the conscious control of the writer. The problem with this assumption is that many features that define the style of an author certainly are under conscious control. We therefore prefer to postulate a 'cooperative attitude' of the writer in that he does not wilfully disguise his style.

On the other hand there is no law of nature, other than that of probability, that says that every man has to have unique fingerprints: it is just that the number of potential combinations of all features in a fingerprint are so great that even with five billion living individuals, the possibility that two individuals have exactly the same fingerprint may be discarded. Perhaps we may also accept after all the concept of 'stylistic fingerprints' in the sense of a combination of several standardized features, i.e. combinations of measurable textual features that identify the author of the text beyond reasonable doubt.

We want to emphasize the phrase *standardized features*. If the texts of only a few authors have to be attributed, it is feasible to search all attributed texts for some heuristic feature that may be used to attribute the unknown texts. In the case of the Federalist papers, for example, initially so-called *marker words* were selected, such as 'while' and 'whilst', that differentiated between the two authors. Such features are too narrow to differentiate between several authors. For 'stylistic fingerprints' we would want to use features that do apply to all authors under consideration and that are, moreover, easy to measure; in short: standardized features. Therefore we will assume the following conditions:

- Most important is the availability of sufficient data. This means that we presuppose at least three texts: a text to be classified, a text or group of texts that is positively identified as being written by the author under consideration (the *target author* and the *target group*) and a text or group of texts that certainly are not written by that author (the *control group*, written by the *control authors*). For instance, in the case of the Federalist papers an as yet unattributed paper may be hypothesized to be by Hamilton. In that case Hamilton is the target author; the papers that are positively attributed to be by Hamilton are called the target group and the papers positively attributed to Madison (and to Jay) form the control group.

- Next we may assume a cooperative attitude. As already indicated we assume that the writers under consideration did not take measures to disguise their style to prevent detection or, if they did, that these measures can be recognized and separated from the features used for classification. This goes for both the target author and the control authors. In fact, this assumption would have to be relaxed in cases as those of the Federalist papers, where all authors wrote under the same 'nom de plume' and may consciously have tried to mimick each others stylistic idiosyncrasies.

- Ceteris Paribus: it is also important that the texts to be compared resemble each other in as many respects as possible. If the target author was a 16th century playwright and the text to be attributed is a play too, we should select 16th century plays in the control group. There may be circumstances when this is not possible, e.g. when all positively identified texts of the target author are poems and the disputed text is a letter.

- Standardized features: the features that we use to compare the texts should be chosen such that they apply to all texts that could possibly come under consideration, both in the target group and in the control group. For authorship recognition that has general validity, we should avoid the use of ad hoc features and concentrate on general features.

So when do 'stylistic fingerprints' in fact become a feasible goal? First, the set of texts written by the authors between which to differentiate is large enough to make it a non-trivial subset of all texts written in that language and also that a text is easily classified as to its membership of this subset. The second condition would be that the discriminating features are easily recognized and quantified in the texts themselves.

8.4 Lexical cohesion

Looking back on the effort that has already gone into author attribution, it is perhaps amazing that no attempts have been reported at using so-called *lexical cohesion* as a measure for discriminating between authors. It is a measure that depends on the identification of word tokens and, to a lesser degree, of sentences; tasks that are typically very easily performed in automated text processing.

Also, the avoidance or repetition of words that have occurred earlier in the text is a stylistic act that is performed almost consciously: most of us will recognize the repetition of the same word within too short a distance as an unaesthetic figure of speech unless the author has very good reasons to do so, for example for rhetoric emphasis.

Occurrences of a non-function word type therefore have a tendency to cluster because they are relevant to the local focus of a text, but an opposing tendency also exists in that this clustering cannot be too tight, because that would sin against an aesthetic, indeed a stylistic, principle. The repeated use of function words may be governed by their role as a placeholder for a non-function word (anaphora) to avoid unaesthetic repetitions, but it can of course also be influenced by a host of other factors. We will comment later on the differences between function words and non-function words, when used in the context of author attribution.

The subject of lexical cohesion has been addressed by several authors in the field of linguistics and computational linguistics such as [Morris and Hirst, 1991]. They identify five classes of lexical cohesion:

1. Reiteration with identity of reference.

   1 Mary bit into an *apple*.
   2 Unfortunately the *apple* was not ripe.

2. Reiteration without identity of reference.

   1 Mary bit into an *apple*.
   2 She likes *them* very much.

3. Reiteration by means of super- or subordinate terms.

   1 Mary bit into an *apple*.
   2 She likes *fruit* very much.

4. Systematic semantic relation

1 Mary bit into a *red* apple.
2 She likes *green* ones too.

5. Nonsystematic semantic relation.

1 Mary went into the *orchard.*
2 She took an *apple.*

The first three classes depend on reiteration of the concepts involved; not necessarily of the same lexical term, but also of anaphora or direct thesaural relations such as broader terms, narrower terms and synonyms. Classes four and five depend on other relations than the repetition of (a reference to) the concept. The relations exemplified by class four, the systematic semantic relations, still may be solved relatively easily by a thesaurus or other knowledge representations; the references in class five, the nonsystematic relations, are often very difficult to solve by a formal system.

According to Morris and Hirst, lexical cohesion fulfills two roles: (**a**) that of word interpretation in context and (**b**) that of cohesion and discourse structure. They give the example of how the (narrower) meanings of the words *drink* or *wave* are defined by the context {*gin, alcohol, sober*} and {*hair, curl, comb*}. The second function of lexical cohesion, then, is that of identifying units in discourse structure and connecting such units over gaps of several sentences, and it is this last function that is pressed in service to help in attributing texts to authors.

8.4.1 Text Tiling

So far we have mentioned two different approaches to the identification of such units: that described in Morris and Hirst and extended into a computational system by [Kozima and Furugori, 1993], and secondly that applied by [Hearst and Plaunt, 1993],[Hearst, 1993b]. There is an interesting difference between the two approaches: Morris and Hirst, and by extension Kozuma, concentrate on the semantics of the words by looking up possible related words in a thesaurus, whereas Hearst applies frequency-based weights to identify stretches of sentences that are connected by the occurence of identical word tokens; a technique she calls *text tiling.*

To achieve this she first computes weights for the words in the text in the following manner. First the text in the document is divided into blocks of a heuristically chosen length of 3-5 sentences. Then every word-block combination is weighted with the *tf.idf* measure, which gives a greater weight to the word-block combination when there are more occurrences of the word in the block and fewer in the complete document (see [Salton and McGill, 1983]). The algorithm then walks through the blocks, computing the similarity between each pair of blocks by application of e.g. the cosine formula. After the application of a smoothing algorithm to lessen the effect of local fluctuations the similarities are plotted and the valleys in the graph are pronounced to be the places where tile boundaries occur.

Hearst mentions the possibility of using her algorithm not on logical sentences but on text windows of varying sizes. This was taken up by [Callan, 1994] in experiments in which he tried various ways of breaking up long texts for IR purposes. It was found that text windows of a fixed number of words performed better than passages that were based on textual discourse units (sentences or paragraphs). We decided to use this windowing technique as one of the parameters in our own experiments.

8.4.2 Measures for Lexical Cohesion

From the above, it will be gleaned that there are a number of different approaches to the computation of lexical cohesion. For our experiments we used a combination of both the chaining method of Morris and Hirst, with some refinements, and the so-called 'text-tiling' of Hearst.

1. Following Morris and Hirst, we first wrote a program that counted for each sentence the number of active word chains. An active word chain is the reoccurrence of a word token within a certain number of sentences or words; if two subsequent occurrences of the token are further apart than this threshold value, the chain is considered broken and a new chain starts when again two occurrences of that word within the threshold are detected. The obvious parameter was the size of the threshold itself; we added the possibility to include or exclude certain word categories, and whether the distances were measured in logical sentences (i.e. something that starts with a capital and ends with a point) or in non-overlapping windows of a fixed number of words.

2. Hearst's method was changed in that we did not compute the similarity between consecutive blocks of text, but between consecutive sentences. The weights were computed for blocks of approx. 2000 words. Again we added variations by including or excluding word categories or by using different ways to divide the texts in sentences or in windows of a fixed number of words.

8.5 Machine learning

The recognition of authors or even text genres depends on a great number of noisy and interacting features. Problems of this type have often been solved satisfactorily by machine learning techniques such as neural networks or instance based learning. Success was reported by [Matthews and Merriam, 1997] in recognizing two authors, Fletcher and Shakespeare, in a number of plays attributed to Shakespeare and even in discriminating between passages of those authors in the same play.

Training consisted of presenting the frequencies of the function words *are, in, no, of* and *the* of the training sets to the input layer of a three-layered neural network (Figure 8.1), putting the correct author on the output layer ((0,1) for

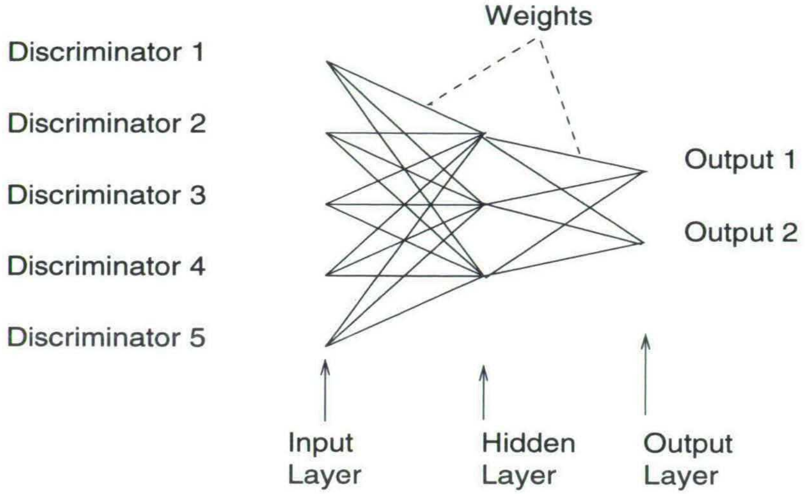


Figure 8.1: Topology for a stylometric neural network (Matthews and Merriam)

Fletcher and (1,0) for Shakespeare. After being trained in this way the neural net was able to correctly attribute each of ten remaining plays. However, it must be noted that these words were suggested as a result of unrelated (i.e. not related to neural network) research [Horton, 1987].

A different tack is the Instance Based Learning approach. This approach is based on the assumption that the simplest form of learning is memorisation. In computer terms, this means storage of the features of an object in a table, together with the identification of the object. If a new object is considered, this table is searched for either an object with the same features or for objects that most strongly resemble that object. But what does 'resemble' mean in this context?

If the features of the objects in the table consist of a single real number, there is no problem; if object A has the feature value 10.3 and object B has 15.2, it is clear that object C with the value 14.1 *in this respect* is most like object B.

However, objects are generally defined by more features than one; these features are often of different classes that are difficult to compare and some or all of these features may influence each other in a number of ways. This is where statistics can play a role: after all, this discipline was developed to make sense of data. Statistical analysis generally is confirmatory: a pattern is hypothesised to exist in the data and the analysis confirms or denies its presence. Machine Learning, on the other hand, is a tool to explore the data and to report existing patterns in a way that is relatively easy to understand.

Our experiments were inspired by the availability of the Waikato Environment for Knowledge Analysis (WEKA [2]), essentially a user interface giving a standardized way to perform a number of machine learning schemes, such as C4.5 [Quinlan, 1993], K* [Cleary and Trigg, 1995] and the IBL variations [Aha et al., 1990], among others. Preliminary experiments indicated that one of the so-called 'lazy learning' algorithms (IBL4) published by [Aha, 1990] and the Kolmogorov (K*) algorithm published by Cleary and Trigg performed best on our data.

8.5.1 IBL4

The IB1, IB2, IB3 and IB4 algorithms are four variations on simple instance-based nearest-neighbour classifiers. IB1 just computes Euclidean distances between the new object and the objects already in the database and assigns to it the class of the nearest neighbour:

$$similarity(x,y) = \frac{1}{\sqrt{\sum_{i \in P} Attr\_diff(x_i, y_i)}}$$

where P is the number of attributes and

$$Attr\_diff(x_i,y_i) = \begin{cases} (x_i - y_i)^2 & i\ is\ numerical \\ x_i \neq y_i & otherwise \end{cases}$$

Every new instance becomes part of the *partial concept description* which in IB1 is the set of stored instances. As IB1 is therefore rather wasteful of storage space, an improvement was made in that only incorrectly classified instances were stored to become part of the database (IB2). This drastically reduces the storage requirements but is less noise-tolerant than IB1. Therefore IB3 was introduced, which also maintains a record of correct and incorrect classification attempts for each instance stored in the partial concept description. In this way the fitness of every instance as a classificator is determinded. This strengthens the noise tolerance and keeps down the storage requirements.

IB4 adds an important improvement on IB3. In the other algorithms it was assumed that the attributes carry equal weight in the predictions of a class. This is often not the case and when instances were described by many irrelevant attributes the older algorithms IB1 - IB3 had no way to detect the less relevant features. The similarity function in IB4 is defined as

$$similarity(x,y,t,P) = \frac{1}{\sqrt{\sum_{i \in P} w_t * Attr\_diff(x_i, y_i)}}$$

where w_t is attribute *i*'s weight when predictions are requested for target concept *t*. When two instances are compared the weight may differ depending on the target concept. As Aha says: 'For example: the similarity of a tiger and a cat is higher if the task is to predict whether they are animals than whether they are potential pets.'

[2] http://www.cs.waikato.ac.nz/ ml

8.5.2 Kolmogorov*

The other learning algorithm that performed well on our data besides IB4 is the K* classifier. This algorithm assumes that, if two instances resemble each other, then there is a high probability of one instance transforming into the other by some accumulation of small mutations. By assigning probabilities to these mutations, a measure can be computed to calculate a distance between one instance and the other. K* incorporates all possible transformation paths in its similarity function and takes as the distance measure the sum of the probabilities of all possible transformation programs, rather than the shortest path.

Positive properties of the K* classifier (which it shares with IBL4) are the fact that attributes of different types, such as reals and symbolic values, can be dealt with within the same framework. The computation of distances between instances that have more than one attribute is straightforward.

The ultimale goal of using a classification system is that the system is trained on a dataset with known classes and that subsequently the class of new, unknown cases is decided on using the results of the training set. By contrast, when we want to study the performance of an algorithm or of selected features, we perform the second test run on data the class of which is already known. The percentage of correct predictions is then used as an indicator of the performance of the algorithm or the suitability of the features. In [Weiss and Kulikowski, 1991] several procedures are described to test the validity of such assumptions, of which the so-called 'tenfold cross-validation' is considered to be the most stringent test of the performance. According to this procedure the data is divided into ten equal parts and that every partition is tested against the nine other partitions. The figures we will quote in the tables are averages computed over the results of tenfold cross-validation, unless explicitly stated otherwise.

8.6 Methodology and experiments

Our main concern here is to establish the performance of lexical cohesion measures as authorship indicators whenever a text fragment has to be attributed to either of two target authors (i.e. authors of whom a sizable text is already available). A secondary goal was to establish the potential of lexical cohesion to uniquely identify an author between *all* authors of a particular genre or group.

We use two different ways of computing lexical cohesion. The first, straightforward procedure is that of counting re-occurring words in sentences and measure their distances in number of sentences or words (chains). The second is that applied by Hearst: sentences are considered as vectors of word weights and lexical cohesion is computed as the sentence-sentence similarity. We decided to create feature databases with variations on both features in the hope that they would reinforce each other. Although we subsequently established that the lexical cohesion as expressed in the number of active chains per sentence carries most of the weight, we decided to keep both features in the database.

8.6.1 Preparation of the texts

We used three small corpora, C-I, C-II and C-III. The first consists of the J-category of the LOB corpus, including thirty fragments of scientific writings. A drawback of this corpus is that only two fragments are by the same author and that the fragments are very short (2000 words). The second corpus contains three books of Jane Austen and one, Wuthering Heights, by Emily Brontë. These writings were collected from the Internet. From each book a 'text' was selected consisting of ten fragments of approx. 2000 words each from the beginning of the book, except for Wuthering Heights, where we extracted two of such 'texts' (the second of which immediatly followed the first). The third corpus and *piece de resistance* was formed by thirty from the eighty-odd federalist papers[3].

First the texts were normalized and enriched by attaching the word category to each word in the text. To obtain the word categories we used the Brill-tagger ([Brill, 1994]), except for C-I as the LOB corpus already has tags attached.

The attachment of the word category has two purposes. First it reduces the problem of homographs in those cases where the same token could be reduced to two or more word categories. Second, and more importantly, it allows us to introduce the word category as a variable in our experiments.

8.6.2 Creation of the databases

For each experiment we created for every text a number of databases in which for every sentence the following information was collected:

1. number of active chains,
2. sentence-sentence similarity using the $atc-$weights (see below),
3. mean of the $atc-$weights of the words in that sentence,
4. for each of these three attributes, the difference between two subsequent values.

In the sections below we will give a more detailed description of the manner in which this information was collected.

[3] Details on corpora and authors are to be found in the appendix.

General details

The programs that extracted the information from the texts were designed with a number of general options, that applied to all programs, and options, which were used to control the paramneters of specific programs. General options included:

- limiting the processing of the data to the function words or to the non-function words,
- minimum wordlength,
- the option of using all word categories in the text or only one or two categories (nouns and/or adjectives),
- whether sentences were used or windows consisting of a fixed number of words,
- the option of using n-grams instead of word tokens.

These options concern mainly various parts that can be filtered out from the text before the actual processing starts or how the text is divided into parts. The individual programs also had options to influence the processing proper:

- for word chains: the maximum length of a chain before it is considered 'broken';
- for word weights: the particular way in which the word weight was obtained (*atc*);
- for vector comparisons: the exact similarity measure (Jaccard, Dice or cosine).

Details of the *atc*-weight

For the tagged files we computed the *tf/df* or *tf.idf* weight of each word-fragment combination. The *tf* or term frequency is the number of occurrences of a certain word in the fragment, and the *df* or document frequency is the number of fragments in which that word occurs. A popular variation is the so-called *atc*-weight, that was also used in the Hearst experiments. It calculates the *tf.idf* in three steps. The first step creates the value *new_tf* for the term-frequency (*tf*) as

$$new\_tf = 0.5 + 0.5 * \frac{tf}{max\_tf}$$

where *max_tf* is the frequency of the term with the highest frequency in the fragment. Then the weight *new_wt* is calculated as

Austen	Brontë	classified as
346	189	a Austen
61	396	b Brontë
		Correct: 74%
Sense	Pride	classified as
319	166	s Sense & sensibility
248	263	p Pride & prejudice
		Correct: 58.4 %

Table 8.2: Confusion tables from K* (class E)

$$new\_wt = new\_tf * log\frac{N}{D_t}$$

where as before N is the number of fragments and D_t the document frequency of term t. Finally the cosine normalization is applied by

$$new\_wt = \frac{new\_wt}{\sqrt{\sum_{i=1}^{T} new\_wt_i^2}}$$

where T is the length of the document vector, i.e. the number of unique terms in the database.

For a detailed discussion of these and similar techniques see, for example, [Salton and McGill, 1983] and [Salton, 1989].

Preparing the data

The results were organized in databases consisting of the features of every sentence for a text fragment with both the author and the fragment as potential classes.

The next step consisted of running two of the ML algorithms, IB4 and K*, that were included with WEKA, on those databases.

Before we applied the machine learning algorithms directly, we first tried to gauge the performance of both algorithms in some more detail.

This was done by concatenating two databases into a new database with randomized order, splitting the new database in two parts, training the algorithm on one part and then making it classify the second, unseen part. The performance is measured in percentages of correctly classified cases and if we have two classes, a random attribution would cause 50% of the cases to be classified correctly. This extreme would mean that the ML algorithm performed badly or that the cases were very similar or both. On the other hand, a score of 100 percent would mean that the algorithm worked very well and that the cases differed strongly between the classes.

Therefore, the precision with which the ML algorithm was able to classify the unseen part of this database was taken as a measure for the *dissimilarity* between the two original databases: a high performance in assigning the

	sm	fw	wnd	ng
A	1	.	.	.
B	5	.	.	.
C	5	.	20	.
D	5	1	.	.
E	5	1	20	.
F	5	2	.	.
G	5	2	20	.
H	5	.	.	3

Table 8.3: Variations

sentences (cases) to the correct texts (classes) meant that the two texts were different from each other; bad classification performance indicated a high similarity between the texts.

The hypothesis to be tested was that texts of *different* authors displayed big differences, i.e. good scores in the classification by the ML algorithms, whereas texts by the *same* author, even from different works, should be difficult for the ML program to classify and therefore approach the 50% mark.

Trying K* and IB4: it was found that our datasets were best classified by IB4, although the other algorithm also performs satisfactorily.

As an example, we provide in Table 8.2 two confusion tables from the output of the K* algorithm. In the upper part of the table, class *a* refers to sentences from a text by Austen; class *b* to sentences from a text by Brontë. The first column displays the number of lines from Austen that are recognized as belonging to the Austen text and those wrongly assigned to Brontë, the second column gives the attributions for the Brontë lines. In this particular test almost 75% of the sentences is classified correctly, with (in this particular case) an as yet unexplained bias towards wrongly recognizing lines from Brontë as coming from Austen.

In the lower part we see the result when classification is attempted over two fragments from different books, but from the same author (Jane Austen). The number of correctly classified instances is now only 58%.

8.6.3 The search space

With all possibilities and variations the experiment space had grown rather large and we did not have the opportunity to exhaustively test all possible variations to find the optimal combination of databases and algorithms. In Table 8.3 we have aligned the variations that we tried. The first column refers to the identifiers given to the datasets. The second column, *sm*, gives the smooth-factor, i.e. the number of sentences we averaged over to smooth local fluctuations. The third column, *fw*, indicates whether the list with function-words was omitted (1) or whether the processing was limited to the function words in the text (2). The column *wnd* indicates the window size when windows of a fixed number of words were used in stead of grammatical

exp	same	other	diff	exp	same	other	diff
	no smoothing						
A	57.26	57.23	-0.02				
	smooth n=5				smooth n=10		
B	64.89	66.91	2.01	I	74.26	75.04	0.78
C*	60.95	69.11	8.16	J*	69.56	76.00	6.44
D*	66.86	72.74	5.88	K*	75.31	81.06	5.75
E*	65.27	79.86	14.58	L*	73.83	86.44	12.6
F*	66.82	71.11	4.29	M*	74.51	77.64	3.14
G*	65.52	71.87	6.35	N*	74.07	79.27	5.20
H	63.36	63.80	0.44	O	68.50	69.59	1.09

Table 8.4: Differences between the experimental databases classified by IB4. Asterisks indicate the .99 confidence level (T-test).

sentences. If the last column, *ng* is filled, it refers to the length of the n-grams, if used. A second collection of files, identified by the capitals I-O was similar to the files B-H, but with a smoothing factor of ten. This group is not shown in the table.

In Table 8.4 the results of the experiments are displayed. Columns 1 and 5 indicate the character associated with the experiment as defined in Table 8.3. The columns *same* show the classification results for two fragments from the same author (Austen). The columns *other* for two fragments from different authors (Austen and Brontë). Experiments E and L (with features that were computed over text windows, using a list of function words to be ignored) display great differences in classification accuracy, and so, to a lesser degree, do the experiment pairs C-J, D-K, F-M and G-N. The asterix attached indicates differences on the 99% level as computed by the T-test.

It was found that tri-grams performs badly; the same is true to a lesser degree for the features measured over the logical sentences (A, B and I).

Austen versus Brontë

As we have noted, the effectiveness of lexical cohesion as an author recognizer depended on the accuracy with which the ML algorithm was able to classify the sentences of different texts. It should be significantly more difficult to classify sentences from two texts by the same author than those of two texts by different

	pride	sense	nabby	wuther	awuth
pride	.	61.7	57.9	75.8	71.7
sense	59.1	.	59.4	70.1	67.8
nabby	54.4	59.9	.	73.2	70.7
awuth	75.2	72.2	74.5	.	55.6
wuther	73.9	67.4	71.9	56.8	.

Table 8.5: Cross-table of three fragments by Austen and two by Brontë, showing averages from a ten-fold classification test. Method: E.

	pride(A)	wuth1(B)	pride(A)	wuth(B)
	C		I	
sense(A)	0.64	0.36	0.65	0.35
nabby(A)	0.64	0.36	0.66	0.34
wuth2(B)	0.39	0.61	0.34	0.66
	D		K	
sense(A)	0.42	0.58	0.41	0.59
nabby(A)	0.51	0.49	0.52	0.48
wuth2(B)	0.28	0.72	0.23	0.77
	E		L	
sense(A)	0.51	0.49	0.58	0.42
nabby(A)	0.62	0.38	0.68	0.32
wuth2(B)	0.26	0.74	0.30	0.70
	F		M	
sense(A)	0.61	0.39	0.66	0.34
nabby(A)	0.60	0.40	0.68	0.32
wuth2(B)	0.41	0.59	0.41	0.59
	G		N	
sense(A)	0.57	0.43	0.57	0.43
nabby(A)	0.58	0.42	0.63	0.37
wuth2(B)	0.41	0.59	0.39	0.61

Table 8.6: Results of direct classification in K* of two fragments by Austen and one by Brontë after training on Pride and Wuth1

authors. In Table 8.5 we see a cross-tab of five texts, three from different books by Austen and two collections of fragments from Wuthering Heights by Brontë. The upper left and lower right segments display the percentages of correctly classified sentences of texts by the same author; upper right and lower left for different authors. Again it is clear that the program performs far better on texts by different authors than on texts by the same author.

Finally we applied the ML algorithm K directly, training on two fragments of Austen and Brontë respectively and then leaving it to the algorithm to classify the test fragment (see Table 8.6).

The LOB corpus

As already indicated, we applied this method to three corpora. The tables displayed above all were taken from experiments on the Austen-Brontë corpus (C-II). In the next corpus to consider, the LOB texts, the situation is rather different in that not two texts were compared but thirty and that, moreover, the texts were much shorter (2000 versus 20,000 words per fragment). Also, only two texts (23 and 24) were by the same author. We first did a tenfold cross-validation, comparing text 23 with all other texts, including itself. As text 23 and 24 were the two texts by the same author one would expect, if lexical cohesion was a sound author discriminator, that text 23 would score lowest, followed by text 24, with a sizable gap between 24 and all other texts.

As a matter of fact this was not always the case. Over the experiment

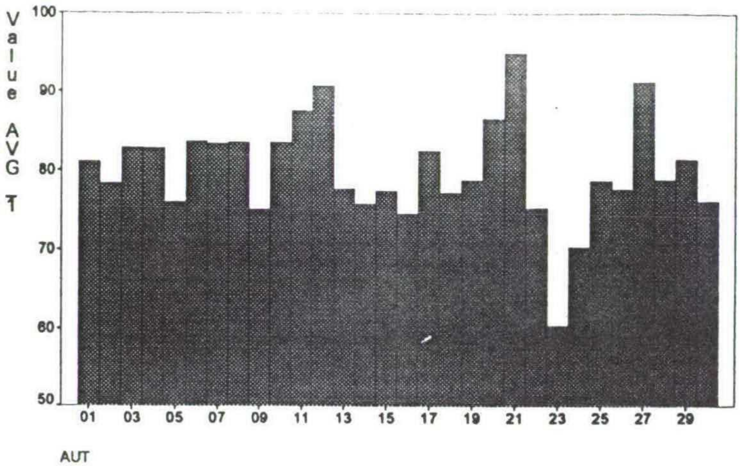


Figure 8.2: Average accuracy on comparing frag. 23 with all other fragments

	K*				IB4			
	ham	mad1	mad2	disp	ham	mad1	mad2	disp
ham	.	87.4	81.1	80.7	.	84.3	80.6	75.8
mad1	85.5	.	64.3	71.4	83.4	.	58.7	72.1
mad2	84.0	63.0	.	73.2	81.3	59.5	.	70.6
disp	80.9	72.0	73.0	.	76.9	70.9	73.3	.

Table 8.7: Tenfold, KS, IB4 federalist.

classes C-G and J-N, text 24 scored consistently low, but in every run one or two other texts would score even lower, so that 24 never would come out lowest. But when we take the averages over all experiment classes (see Figure 8.2) text 24 still does show up as closest to text 23. It is possible that the differences between the authors would have been more pronounced if the available fragments had been longer, but we never expected that lexical cohesion in itself would suffice to discriminate between any two of authors.

The Federalist papers

The third group of papers that we used for our experiments were the Federalist papers. The initial experiments, conducted on the individual papers, showed disppointing results: no real differences between Madisons and Hamiltons papers were visible. As these papers also were relatively short (anything between 80 and 175 lines) we decided to combine several papers of the two protagonists to larger texts each consisting of four or five original papers. We did the same with the disputed papers and proceeded to compare these groups in the same manner as used with the other two corpora.

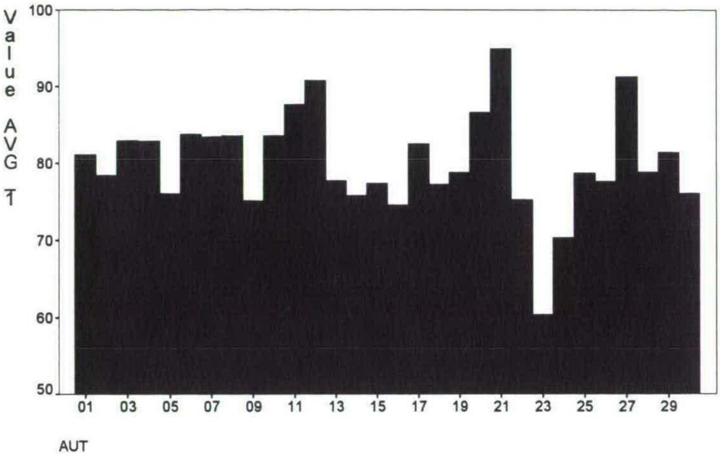


Figure 8.2: Average accuracy on comparing frag. 23 with all other fragments

	K*				IB4			
	ham	mad1	mad2	disp	ham	mad1	mad2	disp
ham	.	87.4	81.1	80.7	.	84.3	80.6	75.8
mad1	85.5	.	64.3	71.4	83.4	.	58.7	72.1
mad2	84.0	63.0	.	73.2	81.3	59.5	.	70.6
disp	80.9	72.0	73.0	.	76.9	70.9	73.3	.

Table 8.7: Tenfold, KS, IB4 federalist.

80 and 175 lines) we decided to combine several papers of the two protagonists to larger texts each consisting of four or five original papers. We did the same with the disputed papers and proceeded to compare these groups in the same manner as used with the other two corpora.

Table ?? shows that according to our *modus operandi* and using the K* algorithm, the disputed papers lie between Hamilton and Madison: the Madison-Madison classifications score 64%, the Hamilton-Madison classifications between 87% and 81%, the Hamilton-disputed score is 80% and the Madison-disputed score is 72-73%. In other words: the disputed papers are almost as dissimilar from Hamilton as the Madison papers, but compared with the Madison groups they lie between Hamilton and Madison. The IB4 algorithm did not so well here; the disputed papers are shown to be different from Hamilton, but still nearer to Hamilton than to Madison.

Now Mosteller and Wallace conclude that the disputed papers are probably written by Madison. Our method so far indicates that they are probably *not* written by Hamilton. The discrepancy that still exists between the figures for Madison and those of the disputed group might be caused by the fact that the writer of the disputed papers consciously tried to change his natural style to

Table 8.7 shows that according to our *modus operandi* and using the K* algorithm, the disputed papers lie between Hamilton and Madison: the Madison-Madison classifications score 64%, the Hamilton-Madison classifications between 87% and 81%, the Hamilton-disputed score is 80% and the Madison-disputed score is 72-73%. In other words: the disputed papers are almost as dissimilar from Hamilton as the Madison papers, but compared with the Madison groups they lie between Hamilton and Madison. The IB4 algorithm did not so well here; the disputed papers are shown to be different from Hamilton, but still nearer to Hamilton than to Madison.

Now Mosteller and Wallace conclude that the disputed papers are probably written by Madison. Our method so far indicates that they are probably *not* written by Hamilton. The discrepancy that still exists between the figures for Madison and those of the disputed group might be caused by the fact that the writer of the disputed papers consciously tried to change his natural style to conform as much as possible to the 'group style' of the papers.

8.7 Conclusions

We have tried to show that lexical cohesion is a computationally cheap way of comparing the style of authors. As expected it does perhaps not suffice in itself to discriminate between *any* two authors, but it certainly is a candidate for inclusion in the set of standardized features necessary to obtain 'stylistic fingerprints'.

Three lines of further research suggest themselves at this point.

- First it could be useful to continue the line of experiments described here. As we have seen only a small part of the experiment space relating to lexical cohesion has been explored. For instance, the maximum chain length was rather arbitrarily fixed at six grammatical sentences, respectively artificial windows of twenty words. Also, we did not limit chains to selected word categories, such as nouns or verbs. By doing this, we could probably get a better combination of lexical cohesion features and the various procedures to obtain them from the original text files.

- Another matter is the quantity of texts that is needed to obtain enough data to train the algorithm on. The best results were obtained when the databases were typically a thousand records (sentences) or more (Austen-Brontë). The LOB-corpus and the Federalist papers generally have no more than two hundred records (sentences) per text. When we combined the papers of Hamilton and Madison in two big texts, the results did improve, but tot to the point that they could be compared to the results of Wallace and Mosteller.

- A third and rather promising line of research would consist of collecting other features of texts that also perform well as author discriminators, and combine them in a standard set, using the *modus operandi* as described above to recognize individual authors.

8.8 Appendix

List of texts used in the comparisons:

8.8.1 Corpus I

The first thirty texts of the LOB-corpus, section J (scientific writings). Each section contains approx. 2000 words. Number 23 and 24 are from the same author (K. Lovell).

8.8.2 Corpus II

Five fragments from four books, downloaded from Internet, the Gutenberg project.

pride: Jane Austen, Pride and Prejudice, first 20,000 words.
sense: Jane Austen, Sense and Sensibility, first 20,000 words.
nabby: Jane Austen, Northanger Abbey, first 20,000 words.
wuth1: Emily Brontë, Wuthering Heights, first 20,000 words.
wuth2: Emily Brontë, Wuthering Heights, words 20,000 - 40,000.

8.8.3 Corpus III

The Federalist Papers, numbers 40-70 as found on the CD-rom 'Bookshelf Compendium', Medialine, Holland, 1996. This group includes nine papers attributed to Madison (40-48), five attributed to Hamilton (64-69), one attributed to Jay (64) and thirteen contested papers (49-63).

Chapter 9

Conclusions

The title of this thesis, *Explorations in the Document Vector Model of Information Retrieval*, could suggest to some readers that the document vector model or DVM is 'terra incognita' and that the author has entered it at great risk to bring back new and unheard-off marvels. As we have seen, this is not exactly true. Information retrieval has used vector representations and vector space operations for thirty years and more, and has for the most part been relatively successful in applying these concepts to improve performance and efficiency. So why the need for a new exploration when there may not be much left to explore?

There are many answers to this question, not the least of which the fact that although the original discoverers of this 'terra incognita' took great pains to document their findings and develop sometimes very accurate and detailed maps, the area never has been really open to the public. Indeed, we have noted the reluctance of libraries and other documentation services to embrace vector-based alternatives for the ubiquitous Boolean model of information retrieval [Paijmans, 1996]. It therefore seems useful to make another map of the area, this time using a different projection and placing emphasis on features that until now have been underdeveloped. The different projection that we used is that of the document vector model as a unifying structure for a large family of IR models, and we shifted the emphasis from retrieval to other applications, such as recognitions of 'gravity wells of meaning', text classification, and author recognition.

The reason for this search for new vistas lies in the recognition of the fact that information retrieval between 1970 and 1980 had reached a point of stasis; all models and submodels described in this thesis had been well established by that time, and while automated indexing proved more efficient than manual indexing [Keen, 1992] [Cleverdon, 1991], in terms of precision and recall it was not a quantum leap forward. This culminated in the wry paradoxes of IR formulated in chapter 1 of this thesis. When discussing the evaluation of IR systems, in chapter 3.2, we described how Shaw and others compared the performance of the cluster model and the vector space model with a baseline of random effectiveness, and found that the effectiveness of clustering was not outside the boundaries of randomness [Shaw et al., 1997a], and that

	Unstemmed	Stemmed
Scheme	Correct-%	Correct-%
Human-average	89	not perf
tf.idf	84	85
...		
C4.5/10c/b/lR	68	not perf
...		
GZIP	47	52
...		

Table 9.1: Classification results on stemmed and unstemmed articles (from Littin, 1995, abridged).

[?]. [?] concluded that "The keywords approach with statistical techniques has reached its theoretical limit and further attempts for improvement are considered a waste of time".

On the other hand, it was shown repeatedly, e.g. by Littin ([?]), that a weighting scheme based on document vectors, the *tf.idf*, performs very well in at least *some* IR-related tasks, such as text categorization (see chapter **??**: part of the table showing the performance of human categorization and various other techniques is reproduced here as Table **??**). Note that categorization using *tf.idf* weights here performs almost as well as human categorization. Lewis too in [?] used probabilistic similarity functions, based on document vectors, to categorize documents, and the inductive learning method of [?], again firmly based on the DVM, did the same.

The difficulty of drawing general conclusions on IR with only the traditional set of models as references, is demonstrated by Blair [?] when ten years after his famous experiment [?], in which the shortcomings of the Boolean model are shown conclusively, he tries to assess the reasons for the continuing popularity of the Boolean model. He describes how the access to documents can be divided into *physical* access and *intellectual* access and identifies the main concern of IR to lie with this "intellectual access". He then divides the (commercial) IR-systems again in a large group that is based on the Boolean model and a much smaller group of advanced 'conceptual' IR systems that are based on statistical or semantic associations between the various terms that have found their way into the document representation. He draws the conclusion that

> "...in spite of the central importance of intellectual acces, commercial [...] developers have applied their resources much more vigorously to the problems of physical access [...] The principal reason for this is that improvements in physical access [...] are relatively easy to measure. Advances in intellectual access are much more difficult and costlty to estimate. A harsh reality of commercial investment is that venture capital flows towards success."

Be that as it may, a division in Boolean models versus conceptual models, where 'conceptual' is in effect almost identical with 'statistical', is not very

	Unstemmed	Stemmed
Scheme	Correct-%	Correct-%
Human-average	89	not perf
tf.idf	84	85
...		
C4.5/10c/b/IR	68	not perf
...		
GZIP	47	52
...		

Table 9.1: Classification results on stemmed and unstemmed articles (from Littin, 1995, abridged).

even the vector space model (which was and is considered the best of the extant models) at best was barely above that baseline [Shaw et al., 1997b]. [Sembok and van Rijsbergen, 1990] concluded that "The keywords approach with statistical techniques has reached its theoretical limit and further attempts for improvement are considered a waste of time".

On the other hand, it was shown repeatedly, e.g. by Littin ([Littin, 1995]), that a weighting scheme based on document vectors, the *tf.idf*, performs very well in at least *some* IR-related tasks, such as text categorization (see chapter 8: part of the table showing the performance of human categorization and various other techniques is reproduced here as Table 9). Note that categorization using *tf.idf* weights here performs almost as well as human categorization. Lewis too in [Lewis, 1992] used probabilistic similarity functions, based on document vectors, to categorize documents, and the inductive learning method of [Apté et al., 1994a], again firmly based on the DVM, did the same.

The difficulty of drawing general conclusions on IR with only the traditional set of models as references, is demonstrated by Blair [Blair, 1996] when ten years after his famous experiment [Blair and Maron, 1985], in which the shortcomings of the Boolean model are shown conclusively, he tries to assess the reasons for the continuing popularity of the Boolean model. He describes how the access to documents can be divided into *physical* access and *intellectual* access and identifies the main concern of IR to lie with this "intellectual access". He then divides the (commercial) IR-systems again in a large group that is based on the Boolean model and a much smaller group of advanced 'conceptual' IR systems that are based on statistical or semantic associations between the various terms that have found their way into the document representation. He draws the conclusion that

> "...in spite of the central importance of intellectual acces, commercial [...] developers have applied their resources much more vigorously to the problems of physical access [...] The principal reason for this is that improvements in physical access [...] are relatively easy to measure. Advances in intellectual access are much more difficult and costlty to estimate. A harsh reality of commercial investment is that venture capital flows towards success."

Be that as it may, a division in Boolean models versus conceptual models, where 'conceptual' is in effect almost identical with 'statistical', is not very satisfying. As we have seen in chapter 4, the Boolean models are based on document vectors, but so are most models that Blair considers to be conceptual, especially those that use statistical co-occurence techniques. Of course, there is more to conceptual indexing than statistical patterns; as Blair and others remarked, semantic aspects also have to be considered. But one of the goals of our thesis was to explore the borders between the different models and therefore we will reserve the term 'conceptual IR' for models that go *beyond* the document vector model.

Therefore in the first three chapters we have given a survey of existing models in Information Retrieval, both outside the DVM and inside. The general model of IR was mentioned in chapter 1 and described in detail in the chapters 4 and 4.3.3. In chapter 4 we also covered a number of sub-models that describe ways to arrive at the vectors and we surveyed the most important models with which to compare such vectors. Although properly speaking the evaluation of IR systems is not bound to these models, for completeness we decided to dedicate a few sections in chapter 3.2 to it.

9.1 Synopsis of results

The chapters 5, 6, 7 and 8 consist of four articles that have been published elsewhere. They all cover aspects of IR and text categorization based on document vectors, even if the categorization in the last article is aimed not at recognition of topical classes, but on the recognition of the author of a text.

In this section we will give a short synopsis of the results reported in each of these chapters in the order in which they have been published and we will show how they are related, more specifically, how they can be seen as consecutive stages on the way to a better understanding of IR and text classification, and of the role that the document vector model plays in these areas.

9.1.1 Clarit-Topic

Chapter 5 was concerned with comparing the performance of two 'conceptual' IR-systems: CLARIT, the experimental system from Carnegie-Mellon, and RUBRIC that was later commercialized as TOPIC. The central problem here was that the two systems use different document representations. Although both are based on the document vector model, both systems went on to build additional knowledge representations on that basis. CLARIT does so by parsing the document for noun phrases and extending the dictionary by overlapping windows of collocations within noun phrases. It then applies a rather complicated system of weighting the terms in this dictionary by comparisons with various corpora. The final document representation of CLARIT looks very much like a document vector, but the keyphrases are divided in three groups: exact terms, general terms and novel terms, which does not fit in with the two-dimensional document vector model.

TOPIC introduces world knowledge in the form of a hierarchy of concepts, the occurrence of which in a document adds weight to the hypothesis that this document is about higher concepts. A major drawback of this approach is that this knowledge has to be coded by hand.

For our study of the performance of these systems, we had to translate the final document representations of both systems to a common denominator. We could have used the initial document vectors that were built by both systems, but both systems are very similar if not identical at this stage.

The subterfuge that we used was to translate the final knowledge representations of both systems back to new document vectors, which gave us the common ground needed to compare the two systems. The conclusions did conform to the all to familiar *Leitmotiv*: there were some slight differences, but no clear superiority of one system. Although we used a fundamentally different approach for our comparisons, the findings were in fact similar to the conclusions of [Gey and Chan, 1988] when they compared RUBRIC with the vector space model: that the RUBRIC system showed a slight improvement over the VSM in the recognition of documents that were *marginally* relevant. But, the authors continue, "...since improvement at the margin is what IR is all about, RUBRIC makes a contribution to advancement of the field...". This does conform to our findings, that

> "...TOPIC scores better when groups of documents are to be retrieved, that cover a broad concept, or when a concept is described using many different, but identifiable terms. CLARIT performs better when the documents display a marked terminology, because such terms are readily recognized against the background of the corpus. The very specificity that CLARIT displays, would point to a possible use in smaller document collections, or collections that limit themselves to a very specialized subject with users that know the specific terms of the trade. The TOPIC-system is more apt for big libraries that cover a rather wide spectrum of subjects." [Paijmans, 1993].

It still is an open question whether this slight advantage of TOPIC is not actually caused by the human component: after all the knowledge about related events and objects was hand-coded into the system. The manufacturer Verity itself does not seem to have much confidence in this feature: it has been quietly relegated to the background of their marketing efforts and in the documentation database of Verity there are no references to be found to RUBRIC (or even to our article cited here).

TOPIC falls within Blair's 'semantic' conceptual IR. CLARIT is a 'statistical' system and although the extraction and scoring of noun phrases follows a rather complicated scheme, the final result again is a document vector system. But as we have seen, neither has performed a quantum leap in regard to each other or to other, more traditional systems, and there is no evidence in literature that there are other conceptual systems that perform significantly better.

9.1.2 Gravity wells

The DVM model is two-dimensional. There is a keyword vector and a document vector, and on the intersection of both vectors a scalar, called 'weight', that is a measure of the relation between the two elements. In IR this relation may be called 'topicality' or 'aboutness'; as we have seen in chapter 4, this weight can be computed in a number of ways, mostly by taking frequency information into account. Indeed, much of the energy in IR has gone into finding ever more sophisticated algorithms to adjust this weight.

Of course, a document is not a monolithic object. It can be divided according to a number of theories in all kinds of structures and the resulting parts can have different topicalities, different functions in the discourse, and different typographies that reflect the functionality of those passages.

Some work has been done in IR where certain parts of the document were singled out for the extraction of keywords ([Buckley and Salton, 1991], [Salton and Buckley, 1993], [Hearst and Plaunt, 1993], [Callan, 1994]), but this always resulted in the splitting up of the original document in subdocuments, which then were treated as documents in their own right. Also, such activities were aimed at recognition of *topical* differences. In IR there have not been many attempts to systematically compare parts of documents for their information weight apart from the assumption that titles and abstracts carry more importance than the rest of the text and, again, indexing them separately. Indeed, the frase *full text* in expressions like 'full text retrieval' or 'full text indexing' for a long time has been synonymous with the use of an *abstract* or an abstract-like stucture (our *document surrogate*) rather than a complete document, and the transition when *full text* came to signify the complete text of a book or article never has been clearly marked. This is one more reason why the results of IR research in 'full text indexing' in the sixties and seventies and even the early eighties are difficult to interpret without clearing up first what each author actually means when using this term.

Curiously the transition from 'abstract indexing' to real 'full text indexing' did not bring increased interest in the structure of the complete document, and there were no further attempts to single out special parts of the document for other than topical segmentation. An exception is perhaps the work of [Kwok, 1984], who proposes to use the bibliography of scientific books and articles for indexing. Kieras [Kieras, 1985] suggests that the first and last sentences of paragraphs in texts carry extra topical emphasis, but he did not try to prove this experimentally in an IR environment.

In chapter 6 we have systematically compared parts of scientific articles, including Kieras' notion of first and last sentences of paragraphs, titles and subtitles. Aside from these positional features, we also tried some non-positional properties like cue-words (we used 'important' and 'significant') or even Part-of-Speech (POS) tags. The conclusion was that, contrary to what was expected the differences of word weights in different parts of the document are hardly, if at all statistically significant. The same is true for the two cue-words that we tested. POS-tagging, which we added as a control measure, *did* yield significant differences in information weight; not unexpectedly it was shown that nouns, adjectives and verbs carry more weight than most function words, but this may barely be called a new insight.

The document collection that we used for these experiments was very small, and we have made no attempts to recognize and test structures that belonged to, e.g. the rhetorical structure theory of Thompson and Mann [Mann and Thompson, 1987] or to the goals, plans and intentions model of Grosz and Sidner [Grosz and Sidner, 1986], the reason being that such structures could not be recognized automatically in a text. Nevertheless the results of the experiments strongly suggests that at least the position of a keyword in a text, or its nearness to a cue-word, does not add much to its information weight.

9.1.3 Local dictionaries

If there is a lesson to be drawn from the work described in the two preceeding sections, it is that traditional IR based on the document vector model has reached the point of diminishing returns. This is not quite unexpected; it has been suggested before that the actual document representation has ony a minor effect on the performance of the system as a whole ([Croft, 1987], [Lewis, 1992]). The same is true for the similarity function that is used. Although Noreault, McGill and Koll found an improvement of 20% effectiveness over random ranking, they also concluded that "While some algorithms were bad, most produced very similar results." [Noreault et al., 1981]. Even with enhancements built upon it as with CLARIT and TOPIC, or with looking into the individual document for positional clues as to which keywords carried more weight than other keywords, these conclusions remain valid.

So far we have only searched the individual document for features that might be helpful to decide on its topicality *vis-a-vis* a user query. There is a different approach, that also uses document vector representations, but that adds existing relevance information: relevance feedback. This is in essence a kind of supervised learning: after an initial query, the user points out the relevant documents in the returned set, and the query subsequently is adjusted by the system towards the relevant documents and away from the irrelevant ones. Relevance feedback, when applicable, is considered one of the most successful techniques ([Croft and Das, 1990]) in IR.

An early application of this is the Rocchio formula for relevance feedback [Rocchio, 1971]; other examples of relevance feedbackcan be found in e.g. [Chen, 1995], among which an interesting application of genetic algorithms. All examples have in common that they are firmly based on the document vector representation. The relevance feedback model has many similarities with techniques that are used in an area that is closely related to IR: text classification. Text classification is concerned with the automatic assignment of classes to texts, or documents. If these classes are known beforehand, the term *text categorization* is preferred. In chapter 1 we aready touched on text classification in non-topical classes ([Biber, 1989], [Pieper, 1979]) and in the next section we will take this up again, but here we are interested in classification in topical classes.

In 1994 Apté, Damerau and Weiss published a study on the categorization of the Reuters database [Apté et al., 1994a] and reported very good categorization results with what they called 'local dictionaries'. These local dictionaries consist of a relatively small number of keywords (80-100) that are selected from the document set, on the criterion of scoring highest on some property in the *relevant* documents. Apté, Damereau and Weiss for some unexplained reason used the plain *term frequency*, that is, without normalizations for e.g. document length, of the words. Also, they decided to use the breakeven point as a measure of performance. Even if we disregard the fact that the breakeven point is not considered a very useful measure anymore (see chapter 3.2), it has the additional drawback that its use is dependent on the ability of the classification system to change the threshold for relevance over the experiments. For these reasons we felt the need to repeat these experiments, using other measurements of performance, varying dictionary sizes, and above all varying approaches to the ranking of the keywords.

The overall conclusion from our own experiments is that local dictionaries are best constructed using as a ranking measure a weight that mixes local information with word frequency information from the *complete* database, as is the case in $tf.idf$ and its variations, rather than word frequency information from the class examples only. The length of the vector or the actual word weights in the vector has less impact on the result. The shortest vector in our experiments was ten keywords, which still did not show a much lower performance than the 20-, 66-, or even 100-word vectors (the results of the 100-word vectors were not published).

The learning program that was used by [Apté et al., 1994a], SWAP-1, is proprietary and not available to us. As both were based on rule induction, we used C4.5 instead [Quinlan, 1993]. A drawback of C4.5 is that the treshold for including a document in a set cannot be varied, and therefore a breakeven point could not be computed. Thus we had to revert to precision and recall and related measures for our comparisons.

9.1.4 Author recognition

To a man with a hammer perhaps everything looks like a nail, and this may have brought us to apply the techniques that we first applied to pure IR and later to topical text categorization, to the classification of text according to author.

We already noted that document vectors do not necessarily have to contain keywords, even in IR. In chapter 4.3.3 we have described how long vectors can be 'condensed' into relatively few singular values by means of statistical methods; the emphasis in IR literature on keywords sometimes makes us forget that there are other data that can be collected, and other uses that it can be put to.

The article included in this thesis as chapter 8 is not about IR at all, but

	pride	sense	nabby	wuther	awuth
pride	.	61.7	57.9	75.8	71.7
sense	59.1	.	59.4	70.1	67.8
nabby	54.4	59.9	.	73.2	70.7
awuth	75.2	72.2	74.5	.	55.6
wuther	73.9	67.4	71.9	56.8	.

Table 9.2: Cross-table of three fragments by Austen and two by Brontë, showing averages from a ten-fold classification test.

about author recognition. The vectors that we used describe not the keywords themselves but variations on lexical cohesion. Of course lexical cohesion itself is a function of the (re)occurence of words and we used techniques from IR to select the most promising descriptors.

Our main concern here is to establish the performance of lexical cohesion measures as authorship indicators whenever a text fragment has to be attributed to either of two target authors (i.e. authors of whom a sizable text is already available). A secondary goal was to establish the potential of lexical cohesion to uniquely identify an author between *all* authors of a particular genre or group.

We used two different ways of computing lexical cohesion. The first, straightforward procedure is that of counting re-occurring words in sentences and measure their distances in the number of sentences or words (chains), what Morris and Hirst call "Reiteration with identity of reference"[Morris and Hirst, 1991]. The second is that applied by [Hearst and Plaunt, 1993]: sentences are considered as vectors of word weights and lexical cohesion is computed as the sentence-to-sentence similarity. We decided to create feature databases with variations on the two features in the hope that they would reinforce each other. We found that lexical cohesion as expressed in the number of active chains per sentence carries most of the weight.

The results of the computation of lexical cohesion according to the various methods were organized in databases, where every tuple describes the lexicon cohesion features of a sentence, both in terms of recurring tokens and in sentence-to-sentence similarity. Several variations were tried, including 'sentences' consisting of n-grams of words, filtering out function words, smoothing, and even n-grams of characters. The n-grams of characters were found to perform poorly; the best performance was found for 'sentences' of 20 words and the filtering out of function words.

Finally we used two supervised learning algorithms, K from Cleary and Trigg [Cleary and Trigg, 1995]) and IB4 from Aha and Kibler [Aha et al., 1990]. We performed two kinds of experiments. One consisted of the straightforward technique of training the learning algorithm on two authors and then leaving it to the algorithm to recognize an unknown text. We also used a technique in wich every experiment consisted of mixing up sentences from two texts and training the algorithm to recognize the text from which every sentence originated. The assumption was that it should be significantly more difficult to classify sentences from two texts by the same author than those of two texts by different authors. In Table 9.2 we see a cross-tab of five texts, three from different books by Austen and two collections of fragments from Wuthering Heights by Brontë. The upper left and lower right segments display the percentages of correctly classified sentences of texts by the same author; upper right and lower left for different authors. The program clearly performs far better on texts by different authors than on texts by the same author.

9.2 Main findings

When we started our research in IR, there was no consistent taxonomy of models in IR that satisfied our need for a coherent description of at least the traditional keyword-based models. We sought to provide a basis for such a taxonomy and have found it in the document vector model. The virtue of this model is that it encompasses all traditional models, including the Boolean models, the vector space model and the probabilistic models in a coherent group that is easily defined by the document representation. These models describe different approaches to the way in which documents are represented and compared, where the objects that are actually compared, are not the documents themselves but the document (and query) vectors. Although this is sometimes forgotten, it is the content of the vector that forms the most important factor in the performance of the system.

At the same time the DVM excludes all models with a document representation that is based on hierarchies or networks, such as RUBRIC. Implementations of such structured models may be added to the system, e.g. in the form of a hierarchical system of subjects in the user interface of a document vector system, but this has no consequences for the document representation as such.

Thus the research in this thesis has been based squarely on document vector representations and on methods to compute the optimal weight for the relation between the document and the feature. Mostly this feature was a keyword, but in the case of author recognition, measures of lexical cohesion were choosen. Clearly, lexical cohesion is itself a function of the (re)occurrence of words; we used techniques from IR to select the most promising descriptors.

Looking back at the results reported in this dissertation, and notwithstanding the foreboding remarks of [Sembok and van Rijsbergen, 1990], we are more than ever convinced of the potential still inherent in the vector representation of documents. This potential will not be realized in traditional, 'direct' IR, where the system has to match query and documents instantaneously, but there is still much that can be done with document vectors in relevance feedback and for classification purposes, both in IR and in other linguistically motivated research.

nnn.10	bhp tvx war coin compan consolid norgold nbh asamer
atc.10	feet grad corp ton resourc reserv assay expect produc
glob.atc.10	gold ounc mine ton ore silf grad assay coin

Table 9.3: Highest scoring features for documents from the class 'gold' according to three different weighting methods

9.3 Future work

According to [Lawrence and Giles, 1998], the Internet consisted of over 320 million 'pages' in 1997 and it is anybody's guess what size it will be at the change of the millenium. Of course many of these 'pages' are irrelevant or even worthless as potential containers of information, but the sheer volume of potentially relevant pages does offer a great problem for the IR specialist if only to separate chaff from wheat. One of the solutions may have to be found in a system of pre-classification, based on machine learning techniques, but it should be realized that faced with these problems, some Internet agencies have reverted to *human* indexing (e.g. Yahoo, or Netscape's 'Open Directory' project). However, these efforts are not aimed at assigning keywords, but rather at the creation of hierarchies.

In any case, the search engines will be stuck with the DVM because, when all is said and done, it is a simple and computationally rather cheap model. So the (topical) relation between the keyword and the document will be expressed in a single weight. Much of traditional research in IR has been aimed at improving this set to some ideal 'optimal' solution. But of course there is no such 'optimal' set, because if nothing else, "...all users of information retrieval systems are not created equal..." [Borgman and Belkin, 1987].

Still, there may be some mileage left in the traditional query-matching methods. As we already noted, the creation of document vectors is relatively cheap, and adding variations often is even cheaper because the data already have been collected. Now one of the interesting features of the weights and similarity functions that we have described here is the great number of equivalent strategies to arrive at a set of documents that match a particular query. If we consider the example of Table 9.3 where the highest scoring words for a certain class of documents are represented, we see how little overlap actually exists, even if the indexing methods are very much related. In this example, we used the frequency (nnn), a local *tf.idf* (atc) and a global *tf.idf* (global atc; for a more precise description see chapter 7). Matched with a query, each document representation will rank the documents differently and return a different set of documents above a certain threshold. At this moment we must assume that the differences between the ranking of keywords according to different algorithms are caused by the 'internals' of the ranking algorithms and that there is no direct mapping from the different algorithms to different groups of users or even different 'shades' of topicality. However, this could relatively easy be investigated by, for example, performing experiments that systematically index multi-class documents that have a single class in common, using different algorithms, and then try to differentiate the documents according to ranking differences between the methods used.

If and when in this way a mapping can be established between the indexing and comparison algorithms on one side, and groups of users or even (sub)topics on the other side, it should be possible for a search engine to offer different combinations of indexing and comparing to the user. As the intrinsics of such combinations would be above most users, the system could present different types of interface, 'personae', masks or personalities that handle the interaction between the system and the user according to different strategies. The user then would associate these ‘personae’ with more or less successful searches, and over time select one or two as his favorite medium.

Hoofdstuk 10

Samenvatting

Dit proefschrift is voortgekomen uit onvrede met de traditionele indeling van information retrieval in een aantal los van elkaar staande modellen, welke indeling weinig innerlijke samenhang vertoont. Het is de gewoonte geworden om te spreken over het Booleaanse model, het vectorruimte model of het probabilistische model, zonder dat zich rekenschap wordt gegeven van een algemeen model waarop al deze varianten gebaseerd zouden kunnen worden. Wij hebben geprobeerd in deze leemte te voorzien door een nieuw model te introduceren dat een aantal van de bestaande modellen in zich verenigt, of er tenminste de basis voor vormt: het document vector model.

Het eerste deel van deze dissertatie beschrijft de geschiedenis en de methoden van information retrieval en vooral die van de geautomatiseerde IR en tekst categorisatie, met sterke nadruk op de traditionele, op vector representatie gebaseerde modellen die in de zestiger en begin zeventiger jaren waren ontwikkeld. Aan het einde van de zeventiger jaren trad er stagnatie op in de ontwikkeling van de information retrieval. Alle belangrijke modellen waren tot in details uitgewerkt en hoewel geautomatiseerde methoden voor het indexeren van documenten even goed of beter bleken te presteren dan de manuele [Keen, 1992] [Cleverdon, 1991], waren de verschillen tussen al deze methoden toch nauwelijks doorslaggevend te noemen [Noreault et al., 1981]. In de tachtiger jaren werden er experimentele systemen gebouwd die waren gebaseerd op ideeën uit de kunstmatige intelligentie, zoals frames en scripts, maar met de opkomst van krachtiger computers en grote machine-leesbare corpora tekende zich ook een terugkeer naar quantitatieve en statistische methoden af die de jaren zestig en zeventig hadden gekenmerkt. Opnieuw bleek echter dat er een onzichtbare limiet voor de prestaties van IR-systemen scheen te bestaan (zie hoofdstuk 3.2 en [Shaw et al., 1997a]) die ook met krachtiger algoritmen niet doorbroken kon worden. Dit leidde tot de formulering van de wrange paradoxen die in hoofdstuk 1 worden beschreven. Om met [Sembok and van Rijsbergen, 1990] te spreken: "The keywords approach with statistical techniques has reached its theoretical limit and further attempts for improvement are considered a waste of time".

Aan de andere kant wordt steeds opnieuw vastgesteld (zie bijvoorbeeld [Littin, 1995]) dat algoritmen gebaseerd op document vectoren, zoals de $tf.idf$, in ieder geval voor een aantal IR-gerelateerde taken goed voldoen en bij taken als bijvoorbeeld tekst categorisatie bijna even goed presteren als de mens.

Tegen deze achtergrond zijn de vier artikelen geschreven die het tweede deel van deze dissertatie beslaan. Zij behandelen allemaal aspecten van IR en tekst categorisatie. We zullen hieronder elk van deze artikelen kort samenvatten en duidelijk maken hoe de bevindingen het belang van de document vector en zodoende van het document vector model onderstrepen.

10.1 Clarit-Topic

Het eerste artikel (hoofdstuk 5, [Paijmans, 1993])behelst de vergelijking tussen twee IR-systemen, CLARIT en TOPIC die nogal van elkaar verschillen. Beiden gaan uit van de document vector, maar bouwen vervolgens twee heel verschillende kennisrepresentaties op die basis. CLARIT ontleedt de tekst om er noun phrases uit te halen en breidt de zo ontstane dictionary uit door deze noun phrases vervolgens te permuteren tot allerlei collocations. Vervolgens wordt een ingewikkeld stelsel van vergelijkingen met allerlei corpora gehanteerd om deze collocaties te wegen en het eindresultaat is een driedelige lijst, waarin de collocaties zijn gegroepeerd naar exacte overeenkomsten, nieuwe termen en algemene termen. TOPIC daarentegen introduceert kennis in het systeem door middel van een door de mens samen te stellen hierarchie van concepten, waarbij elk lager concept een bepaalde bewijskracht heeft voor het voorkomen van het naasthogere concept. De hogere concepten hoeven dan niet letterlijk in de tekst voor te komen; de eindknopen echter zijn letterlijk voorkomende trefwoorden.

Het is duidelijk dat deze twee systemen niet direct met elkaar kunnen worden vergeleken. Dit is opgelost door de uiteindelijke document representaties van beide systemen terug te vertalen naar eenvoudige document vectoren, waarna de vergelijking alsnog betrekkelijk eenvoudig kon worden uitgevoerd. De validiteit van deze methode werd bevestigd door het feit dat de uiteindelijke bevindingen conformeerden aan het beeld dat in de literatuur van beide systemen bestond.

10.2 Gravity wells

Het volgende artikel (hoofdstuk 6, [Paijmans, 1997]) ontleent zijn wat raadselachtige titel aan een term uit de natuurkunde waarmee objecten in het heelal worden beschreven als plaatsen die tegelijk fungeren als een *bron* (van zwaartekracht) en een *put* (die door de zwaartekracht andere objecten naar zich toe trekt). Na het lezen van [Kieras, 1985] en diens aanname dat bepaalde aanwijsbare plaatsen in documenten rijker aan informatie waren dan andere, vroegen we ons af of zulke informatie-rijke passages wellicht ook, analoog aan de *gravity well* uit de natuurkunde, 'informatie-rijke' woorden naar zich toe trokken.

In de literatuur worden diverse methoden beschreven om documenten te ontleden in passages [Hearst and Plaunt, 1993], [Grosz and Sidner, 1986], [Mann and Thompson, 1987], maar theorieën als die van Grosz en Mann zijn nog niet omgezet in bruikbare algoritmen, terwijl de passages van Hearst worden onderscheiden naar onderwerp en niet naar het informatiegehalte. Volgens Kieras echter zijn het vooral de eerste en laatste zinnen van alineas, die rijk zouden zijn aan informatie en deze passages zijn natuurlijk gemakkelijk te herkennen.

Als maat voor het informatiegehalte van een woord namen we de op de document vector gebaseerde $tf.idf$ en daarmee kon het onderzoek worden teruggebracht tot een relatief eenvoudige bepaling van de correlatie tussen een hoge $tf.idf$ van een woord en het voorkomen ervan in de begin- of eindzin van een alinea. Na het berekenen van deze correlaties konden we vaststellen dat deze niet statistisch relevant waren.

10.3 Local dictionaries

Een van de problemen bij het gebruik van document vectoren is dat het aantal keywords in een niet-triviale database kan oplopen tot tien- of twintigduizend, terwijl er in het individuele document slechts enkele tientallen voorkomen. In hoofdstuk 7, [Paijmans, 1998] onderzoeken we methoden om deze lange vectoren terug te brengen tot kortere, waarbij we uitgaan van de door [Apté et al., 1994a] beschreven 'local dictionaries'. Deze local dictionaries zijn korte (honderd woorden of minder) lijsten met termen, die zijn opgesteld op grond van de analyse van reeds geklassificeerde documenten. Aan elke klasse van documenten wordt zodoende een eigen local dictionary toegekend en hiermee worden dan nieuwe, onbekende documenten gecategoriseerd. Hoewel de resultaten van Apté, Damereau en Weiss veelbelovend waren, gebruikten ze een maat voor de prestaties van hun systeem, die moeilijk was om te zetten in een meer gangbare metriek en we haden daarenboven het vermoeden dat hun experimenten geen optimale combinatie van weeg- en leermethoden gebruikten.

Onze bevindingen waren dat de door Apté en zijn collega's gevonden vectorlengte van 80–100 termen nog aanzienlijk kon worden ingekort door de toepassing van $tf.idf$ waarden voor de woordgewichten, waarna ook vectoren met een lengte van zestig, twintig en zelfs tien termen nog een bevredigende prestatie leverden.

10.4 Auteursherkenning

Het laatste hier als hoofdstuk 8 opgenomen artikel, [Paijmans, 1999], onderscheidt zich van de andere drie doordat de documenteigenschappen, welke in de vectoren zijn opgenomen, geen trefwoorden zijn, maar scores van de lexicale cohesie van zin tot zin. De individuele vectoren beschrijven dan ook geen documenten, maar zinnen of passages. Niettemin is het eindresultaat de categorisatie van teksten, zij het niet naar onderwerp maar naar auteur, en de methodes die wij hiervoor hebben ontwikkeld passen in de context van tekst categorisatie en het document vector model.

In [Paijmans, 1997] hadden we kennis gemaakt met de verschillende methoden om een tekst te naar onderwerp te splitsen in verschillende passages, met name door het werk van Hearst. Deze beriep zich op haar beurt op het werk van [Morris and Hirst, 1991] op het gebied van lexicale cohesie en zij beschreef enkele relatief eenvoudige methoden om de inhoudelijke samenhang van opeenvolgende zinnen of passages te bepalen. Dit gebeurde niet alleen door middel van de lexicale samenhang, maar ook door het berekenen van de relatieve informatiewaarde van opeenvolgende zinnen met gebruik van de $tf.idf$ waarde van de woorden als maat voor hun informatiewaarde. Zij toonde aan hoe een 'dip' in de informatiewaarde van opeenvolgende zinnen dikwijls de overgang naar een ander onderwerp markeerde.

Wij hebben beide metrieken in combinatie met 'supervised learning' algoritmes toegepast om te zien of deze metrieken een betrouwbare maat waren om teksten van verschillende auteurs van elkaar te onderscheiden. Hiertoe werden databases gebouwd met voor telkens twee auteurs de scores per zin voor beide metrieken. Een aantal deze supervised learning technieken werd dan gebruikt om na training op een deel van de database het overblijvende deel correct te herkennen. Het bleek dat met name de lexicale cohesie van zin tot zin een betrouwbare maat was om twee verschillende auteurs van elkaar te onderscheiden.

10.5 Conclusies

In het begin van deze samenvatting noemden we de traditionele modellen van IR: het Booleaanse model, het vectorruimte model en het probabilistische model. Tijdens het hierboven beschreven onderzoek viel steeds opnieuw op dat de eigenschappen van al deze modellen, die werkelijk van belang bleken te zijn in de oorspronkelijke beschrijvingen een ondergeschikte plaats innamen. Met name was dat de manier waarop de document vector werd afgehandeld, die toch voor alle modellen, zelfs voor het Booleaanse, de constante basis vormt. We hebben in deze leemte voorzien door een taxonomie van modellen op te stellen, waarbinnen de document vector duidelijk is onderscheiden als de basis voor alle handelingen die in het kader van IR en tekst classificatie worden uitgevoerd.

Bibliography

[ANSI, 1979] ANSI (1979). *American national standard for writing abstracts.* ANSI Z39. 14-1979.

[IBM, 1976] IBM (1976). *Storage and Information Retrieval System/Virtual Storage (STAIRS/VS): General Information.* IBM Program Product 5740-XR1.

[Aha, 1990] Aha, D. W. (1990). A study of instance-based algorithms for supervised learning tasks: Mathematical, empirical and psychological evaluations. Technical report, University of California, Irvine.

[Aha et al., 1990] Aha, D. W., Kibler, D., and Albert, M. K. (1990). Instance-based learning algorithms. *Machine Learning*, 7:37–66.

[anon., 1990] anon. (1990). *TOPIC Retrieval Technology: a technical overview.* Verity corporation.

[Appelbaum and Tong, 1988] Appelbaum, L. A. and Tong, M. (1988). Conceptual information retrieval from full text. In Lichnerowicz, A., editor, *Conference proceedings of RIAO88.*

[Apté et al., 1994a] Apté, C., Damerau, F., and Weiss, S. M. (1994a). Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12:233–251.

[Apté et al., 1994b] Apté, C., Damerau, F., and Weiss, S. M. (1994b). Toward language independent automated learning of text categorization models. In *zSIGIR94.* To appear.

[Baayen et al., 1996] Baayen, H., van Halteren, J., and Tweedie, F. J. (1996). Outside the cave of shadows: Using syntactic annotation to enhance authorship attribution. *Literary and Linguistic computing*, 11 (3):121–131.

[Baker, 1988] Baker, J. C. P. (1988). A test of authorship based on the rate at which new words enter an authors text. *Journal of the association for Literary and Linguistic Computing*, 3(1):36–39.

[Bantzer and Toussaint, 1987] Bantzer, P. and Toussaint, C. (1987). Information retrieval - relational? *Nachrichten fuer Dokumentation*, 38:351–360.

[Barr and Feigenbaum, 1989] Barr, A. and Feigenbaum, E. (1986,1989). *The handbook of artificial intelligence (four volumes).* Addison Wesley.

[Belew, 1989] Belew, R. K. (1989). Adaptive information retrieval. In *SIGIR 1989: 12th international conference on Research and development in information retrieval.* Grenoble, France 1988.

[Belkin and Croft, 1987] Belkin, N. and Croft, W. (1987). *Retrieval techniques*, volume 22, pages 109–145. Elsevier Science Publishers B.V.

[Belkin and Croft, 1992] Belkin, N. J. and Croft, W. B. (1992). Information filtering and information retrieval: two sides of the same coin. *Communications of the ACM*, 35(12):29–38.

[Biber, 1989] Biber, D. (1989). *A typology of english texts.* Linguistics, Vol. 27-1. p. 3-44. jan.

[Biber, 1993] Biber, D. (1993.). *Using Register-Diversified Corpora for General Language Studies.* Computational linguistics, Vol. 19, no. 2, pp. 219-241, June.

[Blair, 1980] Blair, D. C. (1980). *Searching biases in large interactive document retrieval systems.* Journal of American Soc. for Information science, 31:4.

[Blair, 1996] Blair, D. C. (1996). STAIRS redux: thoughts on the STAIRS evaluation, ten years after. *Journal of the American Society for Information Science*, 47(1):4–22.

[Blair and Maron, 1985] Blair, D. C. and Maron, M. E. (1985). An evaluation of retrieval effectiveness for a full-text document retrieval system. *Communications of the ACM*, 28(3):280–299.

[Bookstein et al., 1991] Bookstein, A., Chiaramella, Y., Salton, G., and Raghavan, V. V., editors (1991). *Proceedings of the 14th international conference on research and development in information retrieval.* Association for Computing Machinery.

[Bookstein and Swanson, 1975] Bookstein, A. and Swanson, D. R. (1975). A decision theoretic foundation for indexing. *JASIS*, 26(1):45–50.

[Boorstein, 1983] Boorstein, D. J. (1983). *The Discoverers.* Harry Abrams Inc. New York.

[Borgman and Belkin, 1987] Borgman, C. L. and Belkin, N. (1987). Distributed expert-based information systems : An interdisciplinary approach. *Information Processing and Management (Oxford)*, 23:395–409.

[Brill, 1994] Brill, E. (1994). Some advances in transformation-based part of speech tagging. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94).*

[Britton and Black, 1985] Britton, B. K. and Black, J. B., editors (1985). *Understanding expository text: a theoretical and practical handbook for analyzing explanatory text.* Hillsdale, NJ; Lawrence Erlbaum ass. - 405 pp.

[Buckley and Salton, 1991] Buckley, C. and Salton, G. (1991). Automatic text structuring and retrieval - experiments in automatic encyclopaedia searching. In [Bookstein et al., 1991], pages 21–30.

[Bunt and Extra, 1997] Bunt, H. C. and Extra, G., editors (1997). *De informatiemaatschappij en de multiculturele samenleving.* Katholieke Universiteit Brabant, Tilburg.

[Burrows, 1992] Burrows, J. F. (1992). *Computers and the Study of Literature*, pages 167–204. Oxford: Blackwell.

[Callan, 1994] Callan, J. P. (1994). Passage-level evidence in document retrieval. In Croft, W. B. and van Rijsbergen, C. J., editors, *SIGIR '94; Proceedings of the 17th annual international ACM-SIGIR conference on research and development in Information Retrieval*, pages 302–310. Springer Verlag.

[Chen, 1995] Chen, H. (1995). Machine learning for information retrieval: Neural networks, symbolic learning and genetic algorithms. *Journal of the American Society for Information Science*, 46(3):194–216.

[Chiera, 1960] Chiera, E. (1960). *They wrote on Clay.* The University of Chicago Press. Dutch translation Baarn 1960.

[Chudacek, 1984] Chudacek, J. (1984). *Niet-grammaticale verwerking van natuurlijke talen in computers*, volume 8. Informatie (Deventer).

[Cleary and Trigg, 1995] Cleary, J. G. and Trigg, L. (1995). K*: an instance-based learner using an entropic distance measure. In *Proceedings of the International Conference on Machine Learning.* Morgan Kaufmann.

[Cleverdon, 1984] Cleverdon, A. W. (1984). *Optimizing convenient on-line access to bibliographic databases.* Inf. Serv. Use 4 pp.

[Cleverdon, 1991] Cleverdon, C. W. (1991). The significance of the cranfield tests. In [Bookstein et al., 1991], pages 3–12.

[Cleverdon and Keen, 1966] Cleverdon, C. W. and Keen, E. M. (1966). *Aslib-Cranfield research project.* Cranfield institute of technology, Cranfield, England.

[Conklin, 1987] Conklin, J. (1987). *Hypertext: an introduction and survey.* Computer, September 1987, p. 17.

[Cooper and Maron:, 1978] Cooper, W. and Maron:, M. E. (1978). Foundations of probabilistic and utility-theoretic indexing. *Journal of the ACM*, 25(1):67–80.

[Crestani, 1994] Crestani, F. (1994). Comparing probabilistic and neural relevance feedback in an interactive information retrieval system. In *Proceedings of the 1994 IEEE International Conference on Neural Networks*, pages 3426–3430.

[Croft, 1987] Croft, W. B. (1987). Approaches to intelligent information retrieval. *Information Processing and Management*, 23:249–254.

[Croft and Das, 1990] Croft, W. B. and Das, R. (1990). Experiments with query acquisition and use in document retrieval systems. In Vidick, J., editor, *SIGIR '90; Proceedings of the 13th international conference on research and development in information retrieval*, pages 349–365. Brussel 1990.

[Crouch, 1988] Crouch, C. (1988). An analysis of approximate versus exact discrimination values. *Information Processing and Management*, 24(1):5–16.

[Cutting et al., 1992] Cutting, D., Kupiec, J., Pedersen, J., and Sibun, P. (1992). A practical part of speech tagger. In *Proceedings of the third conference on Applied Natural Language Processing*, pages 133–140. ACL.

[Date, 1983] Date, C. J. (1983). *An introduction in Data-Base Systems.* Addison-Wesley, 3th ed.

[Davies, 1986] Davies, R. D., editor (1986). *Intelligent information systems; progress and prospects.* Horwood ltd. Chicester.

[de Burgh, 1959] de Burgh, W. G. (1959). *Legacy of the ancient world.* MacDonalds and Evans Ltd. London.

[Deerwester et al., 1990] Deerwester, S., Dumais, S. T., Landauer, T. K., Furnas, G. W., and Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the Society for Information Science*, pages 391–407.

[DeJong, 1982] DeJong, G. (1982). *An overview of the FRUMP system.* Erlbaum, London.

[DELPHI, 1992] DELPHI, D. . (1992). *Content based retrieval systems.* A market & technology overview developed by DEC and Delphi.

[Dewey, 1876] Dewey, M. (1876). *A classification and subject index for cataloguing and arranging the books and pamphlets of a library.* published anonymously.

[Dijk, 1980] Dijk, T. V. (1980). *Macrostructures: An interdisciplinary study of global structures in discourse, interaction, and cognition.* Lawrence Earlbaum Associates. Hillsdale, NJ.

[Dumais, 1994] Dumais, S. (1994). Latent semantic indexing (lsi): Trec-3 report. In *Overview of the Third Text Retrieval Conference (TREC-3)*, pages 219–230, Gaithersburg, Maryland.

[Earl, 1970] Earl, L. (1970). Experiments in automatic extracting and indexing. *Information storage and retrieval*, 6:1970, 313–334.

[Edmundson, 1969] Edmundson, H. P. (1969). New methods in automatic abstracting. *Journal of the ACM*, 16:1969, 264–285.

[El-Hamdouchi and Willett, 1988] El-Hamdouchi, A. and Willett, P. (1988). An improved algorithm for the calculation of exact term discrimination values. *Information Processing and Management*, 24(1):17–22.

[Ellegard, 1962] Ellegard, A. (1962). *A Statistical Method for Determining Authorship: the Junius Letters, 1769-1772*. Gothenburg: the University of Gothenburg.

[Ertel, 1976] Ertel, S. (1976). Dogmatism: an approach to personality. In Deichsel, A. and Holzenscheck, K., editors, *Maschinelle Inhaltsanalyse, Materialien I*, pages 34–44. Hamburg University.

[Evans et al., 1991] Evans, D., Handerson, S. K., Lefferts, R., and Monarch, I. (1991). *A summary of the CLARIT project.* Laboratory for computational linguistics. Carnegie-Mellon University, 12 Sept. 1991.

[Farradane, 1966] Farradane, J. E. L. (1966). *Report on research into Information Retrieval by relational indexing.* London, City University.

[Forsyth and Holmes, 1995] Forsyth, R. S. and Holmes, D. I. (1995). The federalist revisited: New directions in authorship attribution. *Literary and Linguistic computing*, 10(2):111–126.

[Forsyth and Holmes, 1996] Forsyth, R. S. and Holmes, D. I. (1996). Feature finding for text classification. *Literary and Linguistic computing*, 11(4):163–174.

[Foskett, 1982] Foskett, A. C. (1982). *The subject approach to information.* London 1969, 4th edition.

[Fox and Koll, 1988] Fox, E. and Koll, M. (1988). *Practical enhanced boolean retrieval: experiences with the SMART and SIRE systems.*, volume 24. Information processing & management.

[Frakes and Baeza-Yates, 1992] Frakes, W. and Baeza-Yates, R. (1992). *Information Retrieval: Data Structures and Algorithms.* Prentice-Hall, Upper Saddle River, New Jersey. - 503 pp.

[Francis, 1966] Francis, I. (1966). An exposition of a statistical approach to the federalist dispute. In Vincent, H. P., editor, *The Computer and Literary Style*, pages 38–78. Kent State University.

[Fuhr, 1995] Fuhr, N. (1995). Information retrieval, skriptum zum vorlesung im ss 93.

[Gey and Chan, 1988] Gey, F. and Chan, W. (1988). *Comparing vector space retrieval with the RUBRIC expert system.* ?? december 1988.

[Ginther-Webster et al., 1991] Ginther-Webster, K., Hart, M., and Evans, D. (1991). Automatic indexing using selective nlp and first-order thesauri. In Lichnerowicz, A., editor, *RIAO91. Conference proceedings of RIAO 91. Intelligent text and image handling*, pages 624–643. Center for Advanced Study of Information Systems - 998 pp.

[Gordon and Kochen, 1989] Gordon, M. and Kochen, M. (1989). Recall-precision trade-off: a derivation. *Journal of the American society for information science*, (40):145–151.

[Grefenstette, 1990] Grefenstette, J. J. (1990). A user's guide to genesis, version 5. Computer Program (AI CD-ROM).

[Grosz and Sidner, 1986] Grosz, B. and Sidner, C. L. (1986). Attention, intentions and the structure of discourse. *Computational linguistics*, 12.

[Hahn, 1990] Hahn, U. (1990). *Topic parsing: accounting for text macro structures in fulltext analysis-*. Information Processing and Management vol. 26, p. 135-170.

[Hahn and Reimer, 1987] Hahn, U. and Reimer, U. (1987). Knowledge based text analysis in office environments: The text condensation system topic. In Lamersdorf, W., editor, *Office Knowledge : Representation, Management, and Utilization : Selected Full Papers Based on Contributions to the IFIP TC 8/ WG 8. 4 International Workshop : Toronto, Ontario, Canada, 17-19 August,*, pages 197–215. Amsterdam [etc.] : North-Holland. 278 pp.

[Harter, 1975] Harter, S. (1975). A probabilistic approach to automated keyword indexing: Part ii. an algorithm for probabilistic indexing. *JASIS*, 26(4):280–289.

[Hayes and Weinstein, 1990] Hayes, P. J. and Weinstein, S. P. (1990). CONSTRUE/TIS: a system for content-based indexing of a database of news stories. In *Second Annual Conference on Innovative Applications of Artificial Intelligence.*

[Hearst, 1993a] Hearst, M. (1993a). *A quantitative approach to discourse segmentation.* ACL 1993.

[Hearst, 1993b] Hearst, M. A. (1993b). Cases as structured indexes for full-length documents. In *Proceedings of the 1993 AAAI Spring Symposium on Case-based Reasoning and Information Retrieval, Stanford CA*. Stanford CA.

[Hearst, 1995] Hearst, M. A. (1995). Tilebars: visualization of term distribution information in full text information access. In Katz, I., Mack, R., and

Marks, L., editors, *Human factors in computing systems*, pages 59–66. New York: ACM SIGCHI.

[Hearst and Plaunt, 1993] Hearst, M. A. and Plaunt, C. (1993). Subtopic structuring for full-length document access. In [Korfhage et al., 1993], pages 59–68.

[Hiemstra and Kraaij, 1999] Hiemstra, D. and Kraaij, W. (1999). Twenty-one at trec-7: Ad-hoc and cross-language track. In *Proceedings TREC 7*.

[Holmes, 1994] Holmes, D. I. (1994). Authorship attribution. *Computers and the Humanities*, 28:87–106.

[Horton, 1987] Horton, T. B. (1987). *The effectiveness of stylometry of function words in discriminating between Shakespeare and Fletcher*. PhD thesis, University of Edinburg.

[Inc., 1990] Inc., I. D. (1990). *A Clarification of Claims made by Verity Corp. vis-a-vis the Capabilities of BASISplus*. Information Dimensions, Inc. La Jolla, California, may 1990.

[Irwin, 1968] Irwin, R. (1968). *Ancient and medieval libraries*, pages 399–415. Volume 1 of [Kent, 1968].

[Jardine et al., 1971] Jardine, N., van Rijsbergen Information Storage, C., and Retrieval, 7, .-. . (1971). The use of hierarchic clustering in information retrieval. *Information Storage and Retrieval*, pages 217–240.

[Jolley, 1968] Jolley, J. L. (1968). *Information Handling: Einfuehrung in die Praxis der Datenverarbeitung*. Fischer Taschenbuch Verlag.

[Jones, 1973] Jones, K. S. (1973). Does indexing exhaustivity matter? *Journal of the American Society for Information Science*, 24:313–316. specificity, exhaustivity.

[Keen, 1992] Keen, M. (1992). Term position ranking: some new test results. In Belkin, N., editor, *SIGIR '92; Proceedings of the 15th international conference on research and development in information retrieval*, pages 66–75. New York, ACM press - 353 pp.

[Kent, 1968] Kent, A., editor (1968). *Encyclopedia of Library and Information Science*. Marcel Dekker, Inc. , New York.

[Ketelaar, 1997] Ketelaar, F. C. J. (1997). *Vertrouwen van Informatie?*, chapter 1, pages 12–25. In [Bunt and Extra, 1997].

[Kieras, 1985] Kieras, D. E. (1985). *Thematic processes in the comprehension of technical prose*, chapter 4, pages 89–108. Volume 1 of [Britton and Black, 1985].

[Korfhage et al., 1993] Korfhage, R., Rasmussen, E., and Willet, P., editors (1993). *Proceedings of the 16th annual international ACM-SIGIR conference on research and development in Information Retrieval.* New York, ACM press - 361 pp.

[Kozima, 1994] Kozima, H. (1994). Text segmentation based on similarity between words. *Literary and Linguistic Computing*, 9:13–19.

[Kozima and Furugori, 1993] Kozima, I. and Furugori, T. (1993). Similarity between words computed by spreading activation on an english dictionary. In *Proceedings of the 6th Conference of the European Chapter of the Association for Computational Linguistics*, pages 232–239.

[Krippendorf, 1980] Krippendorf, K. (1980). *Content analysis. An introduction to its methodology.* Sage publications, London.

[Kwok, 1984] Kwok, K. L. (1984). A document-document similarity measure based on cited titles and probability theory and its application to relevance feedback retrieval. In van Rijsbergen, C. J., editor, *Research and development in Information Retrieval*, pages 221–223. Cambridge, - 433 pp.

[Lancaster, 1977] Lancaster, B. C. (1977). *The measurement and evaluation of library services.* Washington DC.

[Lancaster, 1968] Lancaster, F. W. (1968). *Information Retrieval Systems: characteristics, testing, and evaluation.* Wiley, New York.

[Lancaster, 1976] Lancaster, F. W. (1976). *Vocabulary Control for Information Retrieval.* Information Resources Press. Washington D. C.

[Lawrence and Giles, 1998] Lawrence, S. and Giles, C. L. (1998). Searching the world wide web. *Science*, 280(5360):98.

[Lebowitz, 1983] Lebowitz, M. (1983). *Generalization from natural language text.* Cognitive science, Vol 7, 1983, p. 1-40.

[Lebowitz, 1986] Lebowitz, M. (1986). *An experiment in intelligent information systems: RESEARCHER.* In [Davies, 1986].

[Lehnert, 1981] Lehnert, W. G. (1981). Plot units and narrative summarization. *Cognitive Science*, 5(4).

[Lewis, 1991] Lewis, D. D. (1991). *Representation and Learning in Information Retrieval.* PhD thesis, University of Massachusetts.

[Lewis, 1992] Lewis, D. D. (1992). *Representation and Learning in Information Retrieval.* PhD thesis, Computer Science Dept. ; Univ. of Massachusetts; Amherst, MA 01003. Technical Report 91–93.

[Lewis and Ringuette, 1994] Lewis, D. D. and Ringuette, M. (1994). A comparison of two learning algorithms for text categorization. In *Symposium on Document Analysis and Information Retrieval*, pages 81–93, Las Vegas, NV. ISRI; Univ. of Nevada, Las Vegas.

[Littin, 1995] Littin, J. N. (1995). Applications of machine learning in information retrieval. Technical report, Computer Science Department, University of Waikato, Hamilton.

[Luhn, 1958] Luhn, H. P. (1958). The automatic creation of literature abstracts. *I. B. M. Journal of research and Development 2(2)*, pages 159–165.

[Mackenzie Owen, 1998] Mackenzie Owen, J. (1998). *Kennis in veelvoud; Rede uitgesproken bij de aanvaarding van het ambt van hoogleraar in de Documentaire Informatiewetenschap aan de Universiteit van Amsterdam.* Vossiuspers AUP.

[MacLeod, 1990] MacLeod, I. A. (1990). Storage and retrieval of structured documents. *Information Processing and Management*, 26. 2:197–208.

[MacLeod and Reuber, 1987] MacLeod, I. A. and Reuber, A. R. (1987). The array model: a conceptual modeling approach to document retrieval. *Journal of the American Society of Information Science*, 38:162–170.

[Mann and Thompson, 1987] Mann, W. and Thompson, S. (1987). *Rhetorical structure theory: a theory of text organisation. reprinted from 'the structure of discourse'.* Information Sciences institute. USC.

[Maron, 1965] Maron, M. E. (1965). Mechanized documentation: the logic behind a probabilistic interpretation. *Statistical Associoation Methods for Mechanized Documentation*, pages 9–13.

[Matthews and Merriam, 1994] Matthews, R. and Merriam, T. (1994). Neural computation in stylometry i: An application to the works of shakespeare and marlowe. *Literary and Linguistic computing*, 9(1):1–6.

[Matthews and Merriam, 1997] Matthews, R. A. J. and Merriam, T. V. N. (1997). Distinguishing literary styles using neural networks. In Fiesler, E. and Beale, R., editors, *Handbook of Neural Computation*, chapter 8. IOP publishing and Oxford University Press.

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

[McCune et al., 1985] McCune, B. P., Tong, R., Dean, J., and e. a. (1985). *RUBRIC, a system for rule-based information retrieval.* IEEE transactions on software engineering.

[Minsky, 1981] Minsky, M. (1981). *A framework for representing knowledge.* Mind Design, p. 95-128. MIT press 1981.

[Morris and Hirst, 1991] Morris, J. and Hirst, G. (1991). Lexical cohesions computed by thesaural relations as an indicator of the structure of text. *Computational linguistics*, 17(1):1991, 21–48.

[Mosteller and Wallace, 1964] Mosteller, F. and Wallace, D. L. (1964). *Inference and Disputed Authorship: The Federalist.* Reading, Mass. : Addison Wesley.

[Mosteller and Wallace, 1984] Mosteller, F. and Wallace, D. L. (1984). *Applied Bayesian and Classical Inference - the case of the Federalist Papers.* Springer Verlag.

[Mulders et al., 1992] Mulders, M., Raaijmakers, S., and Verschuur, L. (1992). *DRUIDE, documentstructuur als zoeksleutel.* MMC preprint No. 3, ITK, Tilburg.

[Noreault et al., 1981] Noreault, T., McGill, M., and Koll, M. B. (1981). *A performance evaluation of similarity measures, document term weighting schemes and representations in a Boolean environment*, pages 57–76. Butterworths.

[Olle, 1980] Olle, W. T. (1980). *The Codasyl Approach to Data Base Management.* John Wiley & Sons Chichester.

[Paice, 1990] Paice, C. D. (1990). Constructing literature abstracts by computer: techniques and prospects. *Information processing and management*, 26(1):171–186.

[Paijmans, 1999] Paijmans, J. (1999). Discriminators for authorship attribution. *South African Computer Journal*, 23:30–41.

[Paijmans, 1993] Paijmans, J. J. (1993). *Comparing the document representations of two IR-systems.* IASIS vol. 44, no. 7, august 1993. p. 383-392.

[Paijmans, 1994] Paijmans, J. J. (1994). Relative weights of words in documents. In Noordman, L. G. M. and de Vroomen, W. A. M., editors, *Conference proceedings of STINFON*, pages 195–208. StinfoN.

[Paijmans, 1996] Paijmans, J. J. (1996). Some alternatives for the boolean model in information retrieval. In *TICER Summer school.*

[Paijmans, 1997] Paijmans, J. J. (1997). Gravity wells of meaning: detecting information-rich passages in scientific texts. *Journal of Documentation*, 53(5):520–536.

[Paijmans, 1998] Paijmans, J. J. (1998). Text categorization as an information retrieval task. *South African Computer Journal*, (21):4–15.

[Paijmans and Verrijn Stuart, 1982] Paijmans, J. J. and Verrijn Stuart, A. A. (1982). *A new approach to automated museum documentation*, volume 16. Computers and humanities.

[Pieper, 1979] Pieper, U. (1979). *Ueber die Aussagekraft statistischer Methoden fuer die linguistische Stilanalyse.* Gunter Narr Verlag Tuebingen,.

[Quinlan, 1993] Quinlan, J. R. (1993). *C4. 5: programs for Machine Learning.* Morgan Kaufman.

[Ranganathan, 1967] Ranganathan, S. R. (1967). *Prolegomena to Library classification.* Asia Publishing House, 3 edition.

[Rau and Jacobs, 1990] Rau, L. F. and Jacobs, P. (1990). *SCISOR: extracting information from on-line news.* Communications of the ACM: Vol. 33, no 11. nov. 1990 p. 88-97.

[Rau and Jacobs, 1988] Rau, L. F. and Jacobs, P. S. (1988). Natural lanuage techniques for intelligent information retrieval. In *SIGIR 1988: 11th international conference on Research and development in information retrieval.* Grenoble, France 1988.

[Rau et al., 1989] Rau, L. F., Jacobs, P. S., and Zernik., U. (1989). Information extraction and text summarization using linguistics knowledge acquisition. *Information Processing and Management (Oxford) 25 (1989) nr. 4 p. 419-428 (20 refs.).*

[Rayward, 1975] Rayward, E. B. (1975). The universe of information: the work of paul otlet for documentation and international organisation. In *All-Union Institute for Scientific and Technical Information.* Federation International de Documentation.

[Robertson, 1977] Robertson, S. E. (1977). The probability ranking principle in ir. *Journal of Documentation*, 33(4):292–304.

[Robertson and Jones, 1976] Robertson, S. E. and Jones, K. S. (1976). *Relevance weighting of search terms*, volume 27. Journal of american society for information science.

[Rocchio, 1971] Rocchio, J. J. (1971). *Relevance Feedback in Information Retrieval*, pages 313–323. Volume 1 of [Salton, 1971].

[Roes, 1992] Roes, H. (1992). Topic toepassingen aan de kub. In Paijmans, J. J. and Weigand, H., editors, *Artificial Intelligence and Information Retrieval.* Institute for Language and Technology, Tilburg University.

[Salton, 1968] Salton, G. (1968). *Automatic Information Organization and Retrieval.* McGraw-Hill, New York.

[Salton, 1971] Salton, G., editor (1971). *The SMART retrieval system; experiments in automatic document processing.* Prentice-Hall, Englewood Cliffs, N. J. , 556 pp.

[Salton, 1989] Salton, G. (1989). *Automatic text processing: the transformation, analysis and retrieval of information by computer.* Addison Wesley, - 530 pp.

[Salton and Buckley, 1993] Salton, G. and Buckley, C. (1993). Approaches to passage retrieval in full text information systems. In [Korfhage et al., 1993], pages 49–58.

[Salton et al., 1983] Salton, G., Fox, E. A., and Wu, H. (1983). Extended boolean information retrieval. *Communications of the ACM*, 26:1022–1036.

[Salton and McGill, 1983] Salton, G. and McGill, M. J. (1983). *Introduction to Modern Information Retrieval.* McGraw-Hill New York [etc.] - 448 pp.

[Salton and Yang, 1973] Salton, G. and Yang, C. S. (1973). On the specification of term values in automatic indexing. *Journal of Documentation*, 29:351–372. specificity, exhaustivity.

[Salton et al., 1975] Salton, G., Yang, C. S., and Yu, C. T. (1975). A theory of term importance in automatic text analysis. *Journal of the American Society for Information Science*, 26:33–44.

[Salton and Zhang, 1986] Salton, G. and Zhang, Y. (1986). Enhancement of text representations using related document titles. *Information Processing and Management*, 22:1986, 385–394.

[Schank and Abelson, 1977] Schank, R. and Abelson, R. (1977). *Scripts, plans, goals and understanding.* Hillsdale, New York.

[Sembok and van Rijsbergen, 1990] Sembok, T. M. T. and van Rijsbergen, C. J. (1990). Silol: a simple logical-linguistic document retrieval system. *Information Processing and Management*, 26:111–134.

[Shaw et al., 1997a] Shaw, W. M., Burgin, R., and Howell, P. (1997a). Performance standards and evaluations in ir test collections: Cluster-based retrieval models. *Information Processing and Management*, 33(1):1–14.

[Shaw et al., 1997b] Shaw, W. M., Burgin, R., and Howell, P. (1997b). Performance standards and evaluations in ir test collections: Vector space and other retrieval models. *Information Processing and Management*, 33(1):15–36.

[Smith and Warner, 1984] Smith, L. C. and Warner, A. J. (1984). A taxonomy of representations in information retrieval design. *Journal of information science*, 8:113–121.

[Sparck-Jones and Kay, 1973] Sparck-Jones, K. and Kay, M. (1973). *Linguistics and information science.* Academic press, New York.

[Sprague de Camp, 1961] Sprague de Camp, L. (1961). *The ancient engineers.* Ballantine.

[Tallentire, 1976] Tallentire, D. R. (1976). Confirming intuitions about style using concordances. In Jones, A. and Churchouse, R. F., editors, *The Computer in Literary and Linguistic studies.* University of Wales press.

[Teufel and Schmidt, 1988] Teufel, B. and Schmidt, S. (1988). *Full Text Retrieval Based on Syntactic Similarities,* volume 13. Information Systems (Elmsford, NY).

[Thompson, 1968a] Thompson, L. S. (1968a). *Monastic Libraries,* pages 233–263. Volume 18 of [Kent, 1968].

[Thompson, 1968b] Thompson, L. S. (1968b). *Roman and Greek libraries,* pages 3–40. Volume 26 of [Kent, 1968].

[van de Waal, 1955] van de Waal, H. (1955). Some principles of a general iconographical classification. In *Actes du XVIIme congres international d'esthetique,* pages 601–606.

[van Rijsbergen, 1979] van Rijsbergen, C. J. (1979). *Information Retrieval.* Butterworths, sec. edition. 208 pp.

[Weiss and Kulikowski, 1991] Weiss, S. M. and Kulikowski, C. A. (1991). *Computer Systems that Learn.* Morgan Kaufmann.

[White and Griffith, 1987] White, H. D. and Griffith, B. C. (1987). *Quality of indexing in online data bases,* volume 23. Information processing and management.

[Willett, 1985] Willett, P. (1985). An algorithm for the calculation of exact term discrimination values. *Information Processing and Management,* 21(3):225–232.

[Witty, 1958] Witty, F. J. (1958). The pinakes of callimachus. *Library Quarterly, 28, 2,* pages 132–136.

[Yngve, 1961] Yngve, V. (1961). A model and hypothesis for language structure. In *Proceedings of the AMerican Philosophical Society,* volume 104, pages 444–466.

Index

A

AIR 60

Alexandria 25, 27

analet 66

archives 23

Aristotle 25, 27

artes liberales 27

Assurbanipal 25, 26

authentication 24

author recognition . . 6, 12, 151, 179

auxiliaries 30

B

Benedict, Saint 28

binary cosine 81

Boolean model 4, 15, 20, 43, 66, 77, 92, 112

Boolean operators 15, 38, 43, 58, 67

bottom-up processing 52

breakeven point 46, 179

C

Callimachus 26, 28

Cassiodorus 28

chaining 112, 159, 180

CLARIT 91, 92, 96, 175

classical model 18, 63

cline 7

cluster model 51, 82, 173

codex 27

complete link comparison 83

compound terms 37, 94

Conceptual Dependency 16

controlled dictionary 19, 31

CRANFIELD 100

cue method 111, 131

D

DBMS 14, 65

Dewey 27, 30, 94

Dice coefficient 81, 164

discourse passages 112

discrimination value . . . 20, 71, 115, 116, 131

document representation 13, 35, 65, 93

document surrogate . 13, 35, 64, 93, 177

document vector . . 3, 64, 80, 84, 89, 132, 174

document vector model . . 4, 16, 19, 63, 66, 79, 87, 89, 173, 178

E

Euclidean distance 81, 161

Euripides 25

exhaustivity 48

expert systems 65

expository text 6

F

factor analysis 7, 152

feature reduction 76, 129, 134

feature selection 76, 134

field control 9

first mention theory 111

flat clustering 82

frames 12, 52, 93

Freebase 92

FRUMPS 52

futility point 67, 68

G

genetic algorithms . 51, 85, 86, 135, 145

genre 6

group average comparison 83

H

hard clustering 83

harmonic mean 47, 137
hierarchical clustering 82

I
ICONCLASS 31
incipit 26, 29
index language . . 16, 33, 64, 92, 130
indexing, assigned 41, 66, 93
indexing, derived . . . 41, 66, 94, 133
internet 22, 129, 163, 182
inverted file . 15, 16, 41, 58, 94, 111

J
Jaccard coefficient 81, 164

K
KWIC index 37, 39
KWOK index 39

L
Lafontaine, Henri-Marie 30
latent semantic indexing 76, 89, 129, 134
lexical cohesion 3, 157, 180
LOB corpus 6, 7, 123, 163
London Lund corpus 7

M
macro evaluation 45, 139
matching coefficient 81
micro evaluation 45, 139

N
n-grams 19, 66, 89, 167
neural networks 60, 135, 155
Niniveh 26

O
OCR 34
on-line document 13, 35, 65, 93
Otlet, Paul 30

P
papyrus 27
parchment 27
parser 97
Peek-a-boo system 38
Pinakes 26, 28
Poisson distribution 69
Porphyry 27
positional methods 111, 131
postcoordination 37, 66, 91, 94
precision . 44, 67, 91, 109, 111, 129, 136
precoordination 37, 40, 91, 94
probabilistic model 41, 83, 91
proximity operators 58, 95, 131

Q
quadrivium 27

R
recall . 44, 67, 91, 109, 111, 129, 136
register 7
relevance feedback model 5, 18, 60, 65, 83, 85, 132
RESEARCHER 52
Rocchio algorithm 85, 129, 134
rotation 37
RUBRIC 55, 58, 95, 175

S
scalar product 81
SCISOR 17, 41, 53, 65
scroll 27
semantic passages 112
semantic similarity 77
similarity functions 65, 80, 131, 135
single link comparison 83
singlular value decomposition . . . 76
singular value decomposition . . . 146
SMART 79, 100, 120, 143
soft clustering 83
spanning 103
specificity 48, 94
STAIRS 8, 17, 58, 95
stemming 19
sum of products 81

T
text summarization 56
tf.idf . . 20, 68, 73, 84, 115, 158, 164
thesaurus 19, 31, 33, 94
tilebars 113
top-down processing 52
topic (concept) 96

TOPIC (German) 14, 42, 53, 55, 65
TOPIC (Verity) . 42, 51, 58, 91, 92, 95, 175
topicality 111
TREC 21, 42, 51
trivium 27
truncation 19

U

UDC 30, 94
Uniterm system 37

V

Varro, Terentius 27
vector space model 16, 63, 79
Vivarium, the 28
volumen 27
VSM 59

W

weighted cosine 81
window passages 112
word weights, plain 66
word weights, word-document . . . 66
World Wide Web 22

Z

Zyindex 92

Pictures:

The photographs of the pictures 2.1 and 2.2 are taken from [Chiera, 1960]. The photograph of 2.3 is from [Boorstein, 1983] and that of 3.2 from [Jolley, 1968]. The picture of the ICONCLASS object is from the website at http://iconclass.let.ruu.nl/texts/icsys.htm.

Stellingen behorende bij het proefschrift
Explorations in the document vector model of information retrieval
door J.J. Paijmans

1. De overgang van de boekrol of *volumen* naar het gebonden boek of *codex* werd mede veroorzaakt door het toenemend gebruik van het boek als naslagwerk.
2. De *tf.idf* klasse van woordgewichten combineert lokale informatie van de term in een bepaald document met globale informatie van die term over de gehele database. Bij het categoriseren van documenten door middel van 'supervised learning' levert deze methode betere resultaten op wanneer die globale informatie over de hele database wordt berekend dan wanneer dit slechts over de positieve voorbeelden gebeurt.
3. Men test document retrieval en classificatiesystemen doorgaans door de set documenten die wordt teruggevonden op grond van een vraag, te vergelijken met een van tevoren vastgestelde, optimale set. Deze test houdt echter geen rekening met het feit dat zulke nauwomschreven optimale sets voor niet-triviale vragen doorgaans niet bestaan.
4. Succesvolle methoden om de auteur van een tekst te identificeren door middel van vergelijkingen tussen die tekst en een corpus mogen niet uit een enkel type vergelijking bestaan, maar uit een combinatie of *suite* van zulke methoden. Het berekenen van maten voor lexicale cohesie mag in zo'n suite niet ontbreken.
5. Het traditionele vector space model wordt zo genoemd wegens de centrale plaats die het berekenen van de afstand tussen de document- en query-vectoren hierbij inneemt. Het mist echter een systematische afbakening van die methoden en hetzelfde geldt voor de manieren waarop de woordgewichten berekend zouden moeten worden. Daarom kan het vector space model als model binnen de information retrieval beter worden vervangen door het document vector model.
6. De theorie dat in een alinea de eerste en de laatste zin meer informatie bevatten dan de rest van de zinnen wordt niet bevestigd door een significant hogere *tf.idf* van de woorden in die zinnen.
7. Information retrieval systemen zoals CLARIT, waarin termgewichten worden berekend door middel van vergelijking met een of meer corpora, zijn vooral geschikt voor kleine, gespecialiseerde collecties, waarin 'vaktermen' een belangrijke rol spelen.
8. De universiteit is een der laatste vrijplaatsen voor zonderlingen. Het belang van deze functie kan niet genoeg worden onderstreept.
9. In het belang van de vrijheid van informatie moeten de burgers van een samenleving voorkomen dat de middelen voor het electronisch vastleggen en verspreiden van informatie in die samenleving in handen komen van een enkele persoon of bedrijf.
10. Bij het publiceren van boeken is het van het grootste belang dat dit niet onder hoge tijdsdruk gebeurt. Is dit wel het geval, dan is het bijna zeker dat de auteur de wet van Murphy uit eigen ervaring zal leren kennen.

Tilburg University *Press*
P.O. Box 90153
5000 LE Tilburg
The Netherlands
http://cwis.kub.nl/~dbi/tup

ISBN 90-361-0024-0
NUGI 601