Contents

List of Acronyms **xiii**

List of Symbols **xv**

1 Introduction **1**
1.1 Knowledge Management 1
1.1.1 Knowledge 1
1.1.2 Knowledge Representation 4
1.1.3 Knowledge Management Systems 6
1.1.4 McKnow, McKnow-E and Akropolis projects 8
1.2 Statistical versus Non-statistical Approaches 10
1.3 Objectives and Scope of the Thesis 12
1.4 Structure of the Dissertation 13

I Clustering Algorithms in Knowledge Management **17**

2 State-of-the-Art in Clustering **19**
2.1 Introduction 19
2.2 Clustering Environment 20
2.2.1 Defining a clustering problem 20
2.2.2 Conditioning a document clustering problem 21
2.2.3 Dimensionality reduction 28
2.3 Ideal Clustering Algorithm 35
2.4 Traditional Clustering Techniques 36
2.4.1 Partitional or flat clustering 37
2.4.2 Hierarchical clustering 40
2.4.3 Mixture resolving or probabilistic clustering 41
2.4.4 Density-based clustering 42
2.4.5 Grid-based clustering 44
2.5 Clustering in Large Datasets 45

2.5.1 Traditional algorithms . . . 45
2.5.2 Advanced algorithms . . . 50
2.6 Discussion . . . 56
2.7 Cluster Validity Measurement . . . 57
2.7.1 External measures . . . 58
2.7.2 Internal measures . . . 61
2.7.3 Relative measures . . . 62
2.8 Statistical significance in clustering . . . 62

3 Improving Clustering Techniques 63
3.1 Incremental Tree-based Clustering Algorithm . . . 64
3.1.1 Introduction . . . 64
3.1.2 Algorithm . . . 64
3.1.3 Validation . . . 71
3.1.4 Parameter values . . . 73
3.1.5 Conclusions . . . 76
3.2 Pairwise Adaptive Dissimilarity Measure . . . 77
3.2.1 Introduction . . . 77
3.2.2 Cosine dissimilarity measure variants . . . 78
3.2.3 Proposed dissimilarity measure . . . 79
3.2.4 Validation datasets . . . 81
3.2.5 Validation experimental setup . . . 81
3.2.6 Results . . . 82
3.2.7 Conclusions . . . 93
3.3 A Zipf-based Weighting Scheme . . . 94
3.3.1 Introduction . . . 94
3.3.2 Zipf's law . . . 94
3.3.3 Motivation . . . 96
3.3.4 First approach to a mid-frequency favoring weighting scheme 97
3.3.5 Zipf-based weighting scheme . . . 99
3.3.6 Validation . . . 101
3.3.7 Conclusions . . . 104

4 Applications 105
4.1 Metadata Extraction from Document Content . . . 105
4.1.1 Introduction . . . 105
4.1.2 Document model . . . 106
4.1.3 Concept of multilayer clustering . . . 107
4.1.4 Multilayer clustering process . . . 109
4.1.5 Term network generation . . . 111
4.1.6 Ideal conceptual term network . . . 112
4.2 Identification of Document Types . . . 113
4.3 Identification of Document Properties and Property Values . . . 115

4.3.1 Model enrichment . . . 115
4.3.2 Template identification . . . 116
4.4 Validation . . . 116
4.4.1 Test cases . . . 117
4.4.2 Preprocessing . . . 120
4.4.3 Validation measure . . . 120
4.4.4 Validation procedure . . . 121
4.5 Results . . . 123
4.5.1 20Newsgroup case . . . 123
4.5.2 Research group case . . . 130
4.5.3 Quotes case . . . 135
4.6 Conclusions . . . 141

II Multi-vector Representation of Documents based on Topics 143

5 State-of-the-Art in Multi-vector Representation of Documents 145
5.1 Multiple Vector Representations for a Document . . . 146
5.1.1 Structural information based approaches . . . 146
5.1.2 Content based approaches . . . 147
5.2 Topic Identification Algorithms . . . 148
5.3 Discussion . . . 150

6 Multi-vector Representation for Documents 153
6.1 Introduction . . . 153
6.2 Topic Identification Process . . . 154
6.2.1 Coherence graph construction based on a linkage matrix . . . 155
6.2.2 Graph Laplacian construction . . . 158
6.2.3 Number of topics identification . . . 158
6.2.4 Spectral partitioning of the graph . . . 161
6.3 Validation . . . 162
6.3.1 Validation datasets . . . 163
6.3.2 Validation experimental setup . . . 164
6.3.3 Validation measure . . . 165
6.4 Results . . . 166
6.4.1 Sequential test environments . . . 167
6.4.2 Randomized test environment . . . 169
6.5 Multi-vector Representation based on Topic Identification . . . 170
6.6 Identification and Extraction of Multiple Language Sections . . . 171
6.7 Conclusions . . . 173

7 Applications **175**
7.1 Search Engine . . . 175
7.1.1 Experimental setup . . . 175
7.1.2 Datasets . . . 176
7.1.3 Validation using standardized datasets . . . 178
7.1.4 Results . . . 178
7.1.5 Conclusions . . . 181
7.2 Near-duplicate Identification . . . 181
7.2.1 Introduction . . . 181
7.2.2 Existing approaches . . . 183
7.2.3 Identification process of near-duplicate documents . . . 184
7.2.4 Validation . . . 186
7.2.5 Conclusions . . . 189

8 General Conclusions and Future Work **191**
8.1 Conclusions . . . 191
8.1.1 Cluster techniques . . . 191
8.1.2 Multi-vector representation of documents . . . 193
8.1.3 Applications . . . 193
8.2 Main Contributions . . . 194
8.2.1 Contributions on the technological level . . . 195
8.2.2 Contributions on the application level . . . 196
8.3 Future Work . . . 196
8.3.1 Part I: Clustering techniques . . . 197
8.3.2 Part II: Multiple Vector Representation of documents . . . 198

Appendices

A Clustering: Datasets Descriptions **217**
A.1 Ohsumed based test collections . . . 217
A.1.1 Ohsumed 1 . . . 217
A.1.2 Ohsumed 2 . . . 218
A.2 Reuters based test collections . . . 220
A.2.1 Reuters 1 . . . 220
A.2.2 Reuters 2 . . . 221
A.3 Banksearch . . . 222
A.4 20Newsgroup . . . 223
A.5 TREC based datasets . . . 224

B Language identification process **227**
B.1 Stopwords for language identification . . . 227

C Advances in Clustering **229**
C.1 Incremental clustering 229
C.1.1 Insertion operation for non-leaf node 229
C.1.2 Insertion operation for leaf node 230
C.1.3 Splitting operation for leaf node 231
C.2 Pairwise adaptive dissimilarity measure 231

D Multivector Representation of Documents **235**
D.1 Stopwords for language identification 235
D.2 Algorithmic descriptions 235

E Near-duplicate Identification **241**

F Incremental clustering algorithm **245**
F.1 Operations for Incremental Clustering 245
F.1.1 Insertion operation for non-leaf node 245
F.1.2 Insertion operation for leaf node 245
F.1.3 Splitting operation for leaf node 245

G SAME **249**
G.1 Case Results 249
G.1.1 20Newsgroup 249
G.1.2 Research group 255
G.1.3 Quotes 261
G.2 Algorithmic descriptions 263

Abstract

Over the last decennia the dominating economic driver shifted several times. This paradigm in the last decade has changed to a knowledge driven economy wherein a company maps out and manages its knowledge. The IT-tool that supports this objective, is called a knowledge management system. The documents in a company can be used as a source for such as system, in order to gather, organize and disseminate the available knowledge throughout the company.

In recent years the amount and size of document collections have however vastly increased, spurred by the evolutions in the domain of information technology. These large data pools provide an environment for a knowledge management system from which information and, in a later stage, knowledge can be extracted. In a collection of documents, the grouping or clustering of these documents can offer insights in the information present in the collection. Such an algorithm assigns a cluster label to similar items, while other items are labeled differently. These extracted clusters can convey information about the content of, or missing information in the collection. As the average size of the collection increases in time, the collection of available clustering techniques also needs to be updated in order to counter the emerging difficulties related to this evolution. The computational and memory complexity of these algorithms are an important property. Therefore current clustering techniques are not able to perform well on these large, and complex datasets. The creation of new or adaptations of existing technologies are required the ascertain the previously described manageability. A second important consequence of the ever increasing size of the document collections, is the retrievability of the information an end-user is searching for. As the amount of information increases, the retrievability of this relevant part of information becomes more and more complex.

The goal of the research presented in this dissertation is thus subdivided in two main research objectives

1. The development or adaptation of techniques in the domain of clustering that enables the handling of a large collection of documents.
2. An enhanced accessibility of the information a document contains in large

and complex document collections.

This dissertation is subdivided in two parts. In the first part, research efforts are described in the clustering process and its preprocessing stage. A state-of-the art overview is provided of the existing clustering techniques and its capabilities in the changing clustering world.

Three contributions in the domain of clustering are described in the dissertation:

1. Incremental clustering algorithm
2. Pairwise adaptive dissimilarity measure
3. Zipf based weighting scheme

In the validation process indications are provided proving the significance of these techniques. Especially the dissimilarity measure has shown promising results.

The second research objective results in a novel representation format of documents, which is described in Part II. Currently every document is presented as a vector in a space spanned by all unique features that appear in a document collection. The new multi-vector representation represents every topic in a document as a separate vector. The validation procedure indicated that the presented technique is able to identify the number of, and the topics itself by means of the lexical chains and the application of the spectral theory. This novel representation of documents also offers promising results for enhanced information retrievability.

For both of these research parts, new and adapted applications in a knowledge management system are described. The research in the domain of clustering resulted in a technique to semi-automatically identify metadata based on the content of a document. This metadata is captured by so-called generic document models, which are associated with a set of document properties.

Based on the new vector representation format, two applications were further developed:

- Integration in a search engine
- Near-duplicate identification

The near duplicate identification technique is based on the pairwise comparison of the lexical chains of both documents. The validation process indicated that the usage of lexical chains are a promising research track in the near-duplicate identification process. Experiments were conducted in the context of search engines, meta-data identification and the near-duplicate detection of documents, providing promising results.

Nederlandse samenvatting

Gedurende de laatste deccenia is de economische drijfveer van de ondernemingen meermaals gewijzigd. Sinds een aantal jaren is het besef gegroeid dat kennis een belangrijke drijfveer is in de economische evolutie van ondernemingen.

De IT-toepassing om die aanwezige kennis in een onderneming te kunnen gebruiken, is een kennisbeheersysteem. Eén van de bronnen van een dergelijk systeem zijn documenten die in een onderneming kunnen voorkomen, zoals handleidingen en verslagen van vergaderingen. Door de sterke technologische evolutie van de afgelopen decennia zijn het aantal en de omvang van dergelijke documentverzamelingen echter sterk toegenomen. De bestaande technieken die informatie en kennis uit deze documentverzamelingen trachten te extraheren, zijn veelal niet geschikt voor dergelijke complexe collecties. Een voorbeeld van dergelijke technieken zijn cluster algoritmes, wat in essentie algoritmes zijn die gelijkaardige items eenzelfde clusterlabel toewijzen, en de overige items andere clusterlabels toekennen. De complexiteit, zowel op computationeel als ook op het vlak van geheugencapaciteit, is een belangrijke eigenschap van dergelijke algoritmes. Deze algoritmes kunnen toegepast worden op documentverzamelingen met als oogmerk overeenkomstige documenten te kunnen groeperen in een grote, ongestructureerde documentverzameling.

Een tweede belangrijk gevolg van deze toenemende hoeveel informatie, is de opvraagbaarheid van de nuttige informatie die een eindgebruiker wenst te bekomen. In een steeds toenemende verzameling van informatie wordt het moeilijker om de relevante informatie te identificeren die een eindgebruiker wenst te bekomen.

Deze thesis beschrijft bijgevolg een aantal contributies in twee doelstellingen:

1. Het ontwikkelen en aanpassen van technieken in het clusterdomein die het mogelijk maken om grote, complexe datasets te kunnen beheren.

2. Het beter opvraagbaar maken van informatie in grote, complexe datasets.

Voor de eerste doelstelling zijn drie contributies besproken in het eerste deel van de thesis:

1. Incrementeel clusteralgoritme
2. Paarsgewijze dynamische afstandsmaat
3. Zipf-gebaseerde gewichtsschema

De validatie heeft aangetoond dat deze contributies veelbelovende onderzoekstrajecten zijn.

De tweede onderzoeksdoelstellling, beschreven in het tweede deel, richt zich op het ontwikkelen van een topic identificatietechniek gebaseerd op informatie uit het document, waardoor elke topic door een vector kan voorgesteld worden. Tot op heden wordt elk document voorgesteld als een vector in een ruimte omspannen door alle unieke termen die voorkomen in de documentverzameling.

Door validatie werd vooreerst aangetoond dat het identificeren van het aantal thema's en van de documentonderdelen die horen bij die thema's kan bepaald worden op basis van de ontwikkelde technieken. Het converteren naar een meervoudige vectorvoorstelling leidt bijgevolg tot een verbeterde opvraagbaarheid van informatie in een zoekmachine.

Tenslotte werden voor deze twee algemene doelstellingen ook toepassingen ontwikkeld. Voor de clusterdoelstelling is dit de semi-automatische metadata extractie. Het uitgangspunt voor deze techniek is de voorstelling van een documentverzameling door een aantal generieke documentmodellen, waarbij een set van documenteigenschappen horen.

De meervoudige vectorvoorstelling leidde tot de ontwikkeling van twee toepassingen:

- Integratie in een zoekmachine
- Identificatie van gelijkaardige documenten

De identificatie van gelijkaardige documenten is gebaseerd op gemeenschappelijke woordketens. Indien twee documenten een toenemend aantal woordketens gemeenschappelijk heeft, kan dit een indicatie zijn voor gelijkaardigheid van deze documenten.

Verschillende experimenten werden uitgevoerd waarbij de ontwikkelde technieken toegepast werden in de beschreven toepassingen, wat veelbelovende resultaten opleverden.

Voorwoord

Een goede vijf jaar geleden werd de start gegeven aan de onderzoeksweg die geleid heeft tot de dissertatie die nu voor u ligt. Die vijf jaren waren niet altijd de eenvoudigste jaren, de weg werd meermaals gekruist door momenten van opluchting en frustratie. Maar toch, die mijlpaal, waarbij alles definitief in tekstvorm gegoten wordt, is genomen.

Op de weg doorheen al die jaren, ben ik veel mensen tegengekomen die ik hierbij wil bedanken. Vooreerst wil ik mijn promotoren bedanken, prof. Joost Duflou en prof. Dirk Cattrysse, voor de mogelijkheid om onderzoek te kunnen en mogen verrichten aan het Centrum voor Industrieel beleid. Joost en Dirk, zonder die vele uren voor stuurgroepvergaderingen, de text mining meetings, seminaries, en voor het talloze verbeterwerk in de laatste fase van het schrijfwerk, was het niet mogelijk geweest om dit doel te kunnen bereiken. Daarnaast wil ik ook mijn jury bedanken voor de vele bemerkingen en opmerkingen die ik ontving op weg naar dit eindresultaat: prof. dr. ir. Van Houtte, prof. dr. ir. De Moor, prof. dr. Daelemens, prof. dr. ir. Liliane Pintelon en dr. ir. Crauwels.

Speciale dankwoorden zijn zeker op zijn plaats voor de vele (ex-)collega's van het CIB: de oude garde Mark, zoals Pieter, Bart, tot de laatste generatie Wouter, Jasmine, TODO nakijken. Joris (V.), voor de begeleiding in die eerste jaren, en voor het bijbrengen van die kritische en relativerende visie op onderzoeksresultaten. Dennis verdient tevens een speciale vermelding, voor het thesiswerk rond de metadata, wat zeer nuttig gebleken is in het totale onderzoekswerk. En natuurlijk dank ik ook Prof. Vanoudheusden, of DVO, over de juiste kant van Schelde en de verschillende grondtypes die belangrijk zijn in het leven.

In het CIB heb ik mensen leren kennen waar ik het buiten het CIB ook zeer goed kan vinden. Zoals algemeen geweten kon (en kan) ik het goed vinden met het secretariaat. Al het werk dat daar geleverd wordt om ons te ondersteunen, moet zeker vermeld worden. Daarom: Arianne, Cindy en Evy: hartelijk bedankt voor al die mooie jaren! Arianne, nog eens bedankt voor het tafeltje aan het venster ;). Tenslotte, toch nog een zeer speciaal woord voor een bijzondere collega die ik hoop nog lang te mogen kennen: Paul-Armand. Jaren samen op dezelfde bureau doet

een speciale band creeren: het discussiren over een project of paper, of babbelen over zaken die het leven in het juiste perspectief plaatsen. Paul, bedankt, en ik hoop dat er zo nog vele jaren mogen volgen.

Mijn schoonouders uit het mooie West-Vlaanderen, voor de mooie aangename zondagen die me steeds doen denken aan een vakantiedag, waarbij leven, genieten en babbelen (met een glaasje) echt belangrijk zijn. Ines en Mathieu, voor de vele charcuterie verhalen en om ons wegwijs te maken in termen zoals LBD-tjes.

Mijn ouders, voor de mogelijkheden om zo ver te geraken in het leven: vanuit een warm gezin de wereld mogen verkennen, te mogen studeren en wonen in Leuven en veel steun te krijgen in Boutersem. Elke zondag mogen we weer even terugkeren naar die gezelligheid, en ik hoop dat dit nog lang mag blijven duren. Voor alle steun die we ooit ontvangden, en mogelijks nog zullen ontvangen: heel hard bedankt! Daarbij horen natuurlijk ook Pieter, Kelly en Eva als broer, schoonzus en zus waarmee we al heel wat beleefd hebben, en zeker nog zullen doen. Bedankt voor al die mooie jaren.

De laatste woorden zijn voorbehouden voor iemand waarbij ik niet genoeg mooie woorden kan vernoemen. Niet alleen in die afgelopen vijf jaren was je de rots als steun, de veilige haven in woelige tijden, of de zon in mooie tijden. Al die tijd dat we al samen zijn, in mooie en minder mooie tijden, ben je iemand speciaal geweest waarop ik kon bouwen en kon steunen. We hebben al zo veel meegemaakt, en zullen nog meemaken. Samen met jou kan ik de wereld aan, en met deze tekst is dat wel duidelijk gebleken.

List of Acronyms

AI Artificial Intelligence

BNC British National Corpus

CoP Communities of Practice

DC Document Cluster

DF Document Frequency

DLSI Differential Latent Semantic Indexing

DM Data Mining

DMS Document Management System

DTM Document-by-Term Matrix

HAC Hierarchical Agglomerative Clustering

ICT Information and Communication Technology

IN Information Need

IDF Inverse Document Frequency

IR Information Retrieval

ITF Inverse Term Frequency

KM Knowledge Management

KMS Knowledge Management System

LNRE Large Number of Rare Events

LSI Latent Semantic Indexing

MLC Multilayer Cluster

NLP Natural Language Processing

OCR Optical Character Recognition

PCA Principal Component Analysis

SAME Semi-Automated Metadata Extraction

SSI Square Segments Identification

SVD Singular Value Decomposition

TDM Term-by-Document Matrix

TF Term Frequency

TFIDF Term Frequency Inverse Document Frequency

TP Transition Point

TVSM Topic-based Vector Space Model

VSM Vector Space Model

List of Symbols

$\mathbf{A}$	Adjacency matrix of a graph
$\mathbf{B}$	Degree matrix of a graph
$\mathbf{C}$	Collection of clusters
$\mathbf{C_i}$	Cluster i
$\mathbf{C_{i,j}}$	Cluster i assigned to topic j
$\mathbf{D}$	Document-by-Term matrix (size $n \times m$)
$\mathbf{d_j}$	Vector representation of document j
$\mathbf{d_{j,a}}$	Vector representation of document j, limited to the a most important terms
$\mathbf{F_i}$	F-measure of cluster i
$\overline{\mathbf{F}}$	Average F-measure
$\mathbf{f_i}$	Total frequency of term i in document collection
$\mathbf{f_{i,j}}$	Frequency of term i in document j
$\mathbf{I_f}$	Number of terms with frequency f in the document
$\mathbf{k}$	Number of clusters
$\mathbf{L}$	Laplacian matrix of a graph
$\mathbf{l_{i,j}}$	Linkage weight between entities i and j
$\mathbf{m}$	Number of terms
$\mathbf{m_i}$	Number of terms of document i
$\mathbf{m_{SC}}$	Number of terms of subspace SC
$\mathbf{n}$	Number of documents
$\mathbf{n_i}$	Number of documents containing term t_i
$\mathbf{nt_i}$	Number of documents related to topic i
$\mathbf{s_i}$	i-th treshold
$\mathbf{S_{i,excl}}$	Number of relationships excluded from square segment i
$\mathbf{SC_i}$	Subspace i
$\mathbf{T}$	Number of topics present in a document collection
$\mathbf{TC_i}$	Term chain based on term i
$\mathbf{TC_{t,i}}$	Document entity i in the term chain based on term t
$\mathbf{W}$	Linkage matrix containing the weights of the coherence graph
$\mathbf{w_i}$	Weight of term i
$\mathbf{w_{i,j}}$	Weight of term i in document j

Chapter 1

Introduction

1.1 Knowledge Management

What is the importance of knowledge? Why do we want to manage the knowledge present in an organisation? Over the last decennia the dominating economic driver shifted several times [Platzer, 2008]. In the 70's the competitive weapon for companies was the cost. In the 80's quality and in the 90's service became the important focus point to conquer the market. The paradigm in the last decade has shifted from a service oriented to a knowledge driven economy wherein a company maps out and manages its knowledge [Brinkley et al., 2009]. In the framework of this dissertation a commonly used definition for knowledge management can be described as a set of processes directed at creating-capturing-storing-sharing-applying-reusing knowledge [Sydänmaanlakka, 2000]. The competitive advantage nowadays sought by companies is to offer innovative products to the market, and thus these processes are of vital importance to support the strategy of the organisation.

In the first section, the concept of knowledge is briefly introduced. The importance of knowledge management and knowledge management systems is introduced in the following sections, and different representation formats used in this domain are summarized. The last sections concern the projects in the framework of which this doctoral research was performed, and the derived objectives and scope of the research.

1.1.1 Knowledge

> "All men by nature desire knowledge."

More than two thousand years ago, Aristotle described with this quote the importance of knowledge. Knowledge is a tool; it enables a human being to evolve

through his life. Without the knowledge mankind possesses today, caves would still be its home seeking shelter against weather conditions. Sheltering for rain is something you experience: the rain can result in a cold. Knowledge is in fact experience, it are the skills or thoughts acquired by a person through experience or education. Some people write it down, others capture it in a different manner such as drawings or audio tapes.

Knowledge also creates power, *scientia potentia est* as stated by Francis Bacon in *Meditationes Sacrae*. You know something someone else does not know. You could share it, or you could use it for better or worse. In early centuries people fought for it, nowadays people pay for it.

What is knowledge? Defining knowledge, and distinguishing it from information, is a difficult task. In literature an uncountable number of definitions or descriptions can be found that try to capture the concept of knowledge. An extensive discussion about an elusive concept such as knowledge is beyond the scope of this dissertation. Plato's three analyses of knowledge are considered as the first written-down statements about knowledge acquired in a human's life [Plato. and Levett, 1928]. Many other, more philosophical or psychological followed. To have a workable interpretation of the concept of knowledge, reference is made to one of the most often cited definitions of knowledge [Davenport and Prusak, 1998]:

> "Knowledge is a mix of framed experiences, values, contextual information and expert insights that provides a framework for evaluating and incorporating new experiences and information"

The terms 'information' and 'knowledge' however are often used interchangeably. In [Brinkley, 2008] the difference between knowledge and information is stated as the capacity for intellectual or physical activity. Knowledge is a matter of cognitive capability and enables actors to do and reflect. Information, by contrast, is passive and meaningless to those without suitable knowledge.

This difference is further elaborated in the Knowledge or so-called DIKW pyramid, which describes the relations between Data - Information - Knowledge - Wisdom. This pyramid was presented by Rowley [Rowley, 2007] and is displayed in Figure 1.1. Data are located at the base of the pyramid and represent discrete, objective facts about events. When these data are processed to present useful answers to simple questions (what, where, when), the concept of Information is founded. When combinations are made between data and information to provide answers to 'how' questions, Knowledge is created. Capturing knowledge is finally called Wisdom. Wisdom reveals answers to 'why' questions. Several versions exist of the DIKW hierarchy, some of them downplaying the importance of wisdom.

An example of the differences between these four layers can be found in the application of sales records in a super market chain. The records itself do not

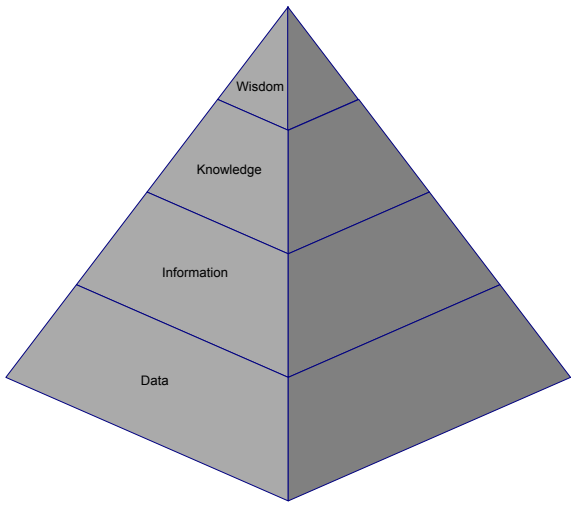


Figure 1.1: Graphical representation of the knowledge pyramid after Rowley [Rowley, 2007]

provide answers to any questions. Summarizing them creates information to answer questions as 'what is sold the most?' or 'when do sale peaks occur?'. Thoroughly analyzing them reveals when and how stock shortages occur, and the resulting wisdom explains why this is occurring. The boundaries of the four layers in the pyramid are not always clear, and as previously stated, the notions of information and knowledge are frequently interchanged.

Knowledge is often categorized in 'tacit' and 'codified' (or explicit) knowledge [Lundvall and Johnson, 1994]. Codified knowledge is knowledge that is made explicit in any form, such as written down on paper or stored in any other (electronic) medium. This type of knowledge can be shared readily among others. Tacit knowledge, however, is knowledge that resides in the mind of the individual and that is often very difficult to make explicit. This knowledge is the personal knowledge of someone and that is difficult to transfer, even if it is communicated. Extensive personal contact is needed to transfer this type of knowledge. The ability to speak a second language is an example of tacit knowledge. Codified knowledge and information are indistinguishably connected in many respects, because this knowledge is often readily available. On the other hand, due to the often personal and hidden nature of tacit knowledge, a significant difference exists between this knowledge and information.

The different types of knowledge and the manner to acquire these types are described in the SECI model [Nonaka and Takeuchi, 1995], shown in Figure 1.2. The gathering of knowledge is governed by four different processes:

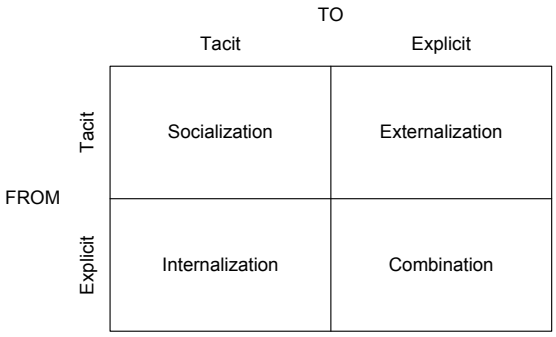


Figure 1.2: Representation of the SECI model with the four main processes [Nonaka and Takeuchi, 1995]

- **Socialization**
  This process describes the communication aiming to share tacit knowledge. An example for this process is apprenticeship.
- **Externalization**
  Converting tacit knowledge into explicit concepts is called externalization. This process is often driven by theoretical models. An example is the creation of a quality model for a product.
- **Combination**
  Combining is defined as the process of aggregating different types of explicit knowledge. Deriving product specifications from technical documents is an example of this process.
- **Internalization**
  As the opposite of the externalization process, internalization is the process of converting explicit knowledge into tacit knowledge, for example through study.

1.1.2 Knowledge Representation

How can we represent knowledge? This question is a key point in Artificial Intelligence (AI) research. Knowledge representation for AI concerns the manner to express what we formally 'think'. AI needs to represent a variety of things: objects, properties, categories and relationships between objects; but also situations, events, states and, if present, notions of time, causes and effects.

A complete description or representation of 'what exists' in a certain domain, is called an ontology. Figure 1.3 represents an example of an ontology of a part of the animal world.

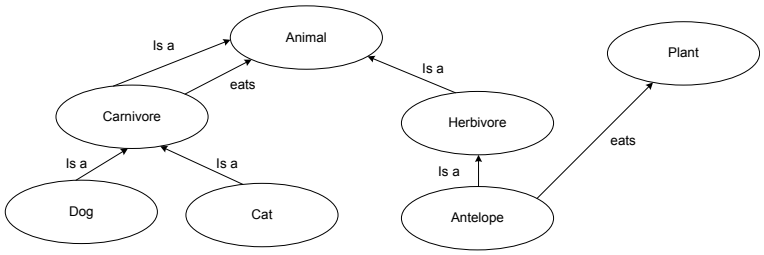


Figure 1.3: Example of an ontology

In the information science domain, the philosophical definition of an ontology was changed towards an explicit specification of a conceptualization [Waterson and Preece, 1999]. It is a representation of an existing domain with the following three properties:

- it represents the properties of the concepts in such manner that a systematic correlation exists between the representation and the related reality.
- it should be comprehensible to domain experts.
- it is codified in such manner that it allows automatic information processing.

A variety of formats exists to construct such a knowledge representation. In the current state-of-the-art, knowledge representation has become an AI research field that applies theories and techniques from several domains, such as [Grimm et al., 2007]

- Semantic Networks
- Rules
- Logic

The use of semantic networks was first introduced by Peirce in 1885 [Peirce, 1885] to capture sentences as graphical diagrams. Since then several other networks appeared, differentiating in syntax and semantics. The common goal of these semantic networks is to express the taxonomic structure of categories or objects and the relationships between them. Nowadays a semantic network is thus generally defined as a graph in which nodes represent concepts and arcs represent relations between these concepts. For a certain domain of interest, such a semantic network provides a structural representation of statements about this domain. Examples of (parts of a) semantic networks are synonyms or antonyms of a word retrieved from Wordnet.

Semantic networks are closely related to the concept of frames as knowledge representation system. Instead of a network, the application of frames results

in nested boxes representing nested concepts. Starting from the concepts and relationships from a semantic network, an ontology can be created using this network and the actual occurrences of the concepts to completely describe the domain. The particular semantic network is then used to express the state of affairs in the domain.

The concept of 'rules' is also often used to describe knowledge in a domain. This approach is more natural to the human mind as rules are used to describe the notion of 'consequence'.

A last popular manner is representing a domain with 'logic'. Logic is concerned with the truth of statements in a certain context. Logic can be seen as the formalization of the previous types of knowledge representation. A semantic network provides a graphical overview, but it remains a vague description that leaves several details undiscovered. To make knowledge computer aware, a formal description to express these models is needed. Logic and ontologies provide these required mechanisms.

The research presented in this dissertation can be mainly situated in the 'Information' layer of the DIKW pyramid. By making information more accessible, processing steps can be made towards the knowledge layer. For example, the combination of semantic networks and the Vector Space Model (VSM) with the multiple vector representation of a document creates an environment enabling knowledge extraction [Jing et al., 2009].

1.1.3 Knowledge Management Systems

IT plays an important role in knowledge management, more specifically in the context of a Knowledge Management System (KMS). Briefly summarized, these systems collect, organize and disseminate the knowledge present in organisations. The input of such systems are usually documents circulating in organisations, such as reports, specifications or memos. These documents contain, after certain filtering processes, a good overview of the basic knowledge and competencies that are present in the organization [Olui-Vukovic, 2001]. Such a system is not only a support tool for managing knowledge in an organisation, it also provides strategic tools as a guide towards new goals to be pursued in the future.

1.1.3.1 Processes in a KMS

According to Davenport and Prusak [Davenport and Prusak, 1998] a KMS covers the following four main stages of knowledge management:

- Acquisition and creation
  In the first phase the knowledge required by an organisation needs to be identified and gathered as an input to the KMS. Knowledge can be acquired in three manners [Lengnick-Hall and Lengnick-Hall, 2002]. The first type of

knowledge is the self-developed knowledge of the company, either explicit or as tacit knowledge of several employees. If the knowledge is not present in the company, the knowledge can possibly be bought (e.g. a firm take-over, buying patents or the hiring of a knowledge worker). If buying knowledge is not possible, renting the required knowledge is the third possibility. Examples of renting knowledge are licenses to software products or the integration of consultants in business processes.

- Codification
  The knowledge gathered in the previous stage is often distributed in different formats. The objective of the codification process is thus to convert this knowledge into an explicit, distributable format. Examples of this process are a description of a business process, or a developed algorithm.

- Distribution
  When the knowledge is converted to a distributable format, services need to be provided to make this knowledge available. Search engines, intranet networks or knowledge bases are a few examples of such services.

- Use
  The last stage in knowledge management is the application of the incorporated knowledge in the KMS. Other employees can consult the KMS for example to prevent the repetitive solving of already-solved problems or as input for innovative processes.

1.1.3.2 Types of Knowledge Management Systems

In literature [Borghoff and Pareschi, 1998, Wenger, 1999, Hahn and Subramani, 2000] four types of KMS are identified:

- KMS operating on unstructured data
- KMS operating on structured data
- Hybrid KMS
- Communities of Practice (CoP)

The first type of KMS uses automatically extracted metadata of the data objects to enable and facilitate search processes. The main advantage of data documented in such manner is that no additional efforts are needed to store these data. However several disadvantages also exist for this type of KMS. Searching for a document requires a full search of the repository, resulting in considerable processing times in larger systems. Also the functionalities of these systems are rather limited.

The opposite of the first type of a KMS uses a maintained knowledge structure that represents the knowledge present in the company, e.g. the operational units in the organization. This categorization is constructed by experts with the cooperation of knowledge workers of the company and is typically stored in a database environment. This underlying structure is not intended to change frequently and therefore is rather static. The deployment of such system, or the adaptation in the underlying structure, requires considerable amount of interactive effort of experts and knowledge workers.

A hybrid KMS is, in general, a KMS that operates on the input of unstructured knowledge. Applying techniques of the Natural Language Processing (NLP) or Information Retrieval (IR) domain (semi-)automatically transforms this unstructured knowledge in structured knowledge. Extraction of metadata and automatic summarization are examples of processes in a hybrid KMS.

The last type of KMS, the CoPs, is different from the first three KMS in the role of Information and Communication Technology (ICT). In the first three KMS ICT is a main actor in the knowledge process whereas in a CoP ICT plays a more supportive or enabling role. A group of knowledge workers uses this ICT tool to share their knowledge and interests among each other. Informal networks are created besides the functional or organizational structure, based on the common knowledge or interest.

1.1.4 McKnow, McKnow-E and Akropolis projects

As already stated in previous sections, the IT-component plays an important role in knowledge management. The research activities described in this dissertation, were partly performed in the context of the McKnow-E and Akropolis projects conducted in close cooperation with ICMS NV. These two projects were preceded by the McKnow project, all funded by the IWT. In the following sections these projects are briefly summarized.

1.1.4.1 McKnow

In the beginning of the 21th century the functionalities of the KMS were merely limited to the reordering of incoming documents into a structure that allowed an end-user to efficiently browse to provide an answer to his question. Disadvantages for this approach were the time consuming manual construction of this structure with several interaction phases with the knowledge workers of the involved company. A consensus structure is not necessarily appropriate for every knowledge worker, it can even hinder the query process of some knowledge workers. Another aspect missing in the generation of KMS at that time, was a functionality to capture the tacit knowledge (see Section 1.1.1).

The main objective of the McKnow project was to develop an experimental platform for new knowledge management methods and supporting algorithms that

could respond to these needs. A number of quantitative techniques were explored to investigate their usefulness for linking information and users of a knowledge management system. The most important innovative elements in the project were

- the integration of user expertise domains in a KMS
- the automation of the generation process of document and user profiles
- the dynamic adjustment of user and document related information based on feedback from the KMS

An important part in the research on supporting algorithms of this project was reserved for clustering techniques. These algorithms will be discussed in Section 2.4.

The project started in September 2001 and finished in August 2005.

1.1.4.2 McKnow-E

The promising techniques obtained from the McKnow project were the first steps towards the integration of the previously mentioned desired functionalities. However, further enhancements were needed to adapt these technologies towards the requirements of the industry. These enhancements were introduced in the follow-up McKnow-E project. The adaptations consisted of adjustments to the developed techniques from the McKnow project, but also new, complementary techniques were developed based on the achievements of the McKnow project. The two major research topics of this project were:

1. advanced user modeling
2. advanced cluster techniques

The first item describes a collection of techniques that enables the representation of knowledge, interests or preferences of the end-user of an application. Among the techniques developed in this part of the research project are the multi-vector profiles of an end-user based on experience and interests, and the automatic identification of the number of profiles.

The other research topic has a twofold function in the context of a KMS. The first function is the supportive role clustering can play in document retrieval. The clustering of documents to group these documents according to their content, can significantly improve the query response. Clustering can also create an overview of knowledge present in a document collection. Besides the supportive role, clustering itself can also provide functionalities. An example of this second function is the ability to provide a topical overview of a document collection.

A large range of clustering techniques already exists, ranging from the straightforward hierarchical clustering to the more complex neural networks. All

these techniques have advantages and disadvantages, but in the context of a KMS, the more detailed techniques are often computationally prohibitive for large and high dimensional datasets. These types of datasets however frequently occur in industrial environments where a KMS needs to prove its efficiency. One objective of the McKnow-E project therefore was the design of a clustering method that is able to operate in a Knowledge Management (KM) environment, and which is well adjusted to the requirements of an industrial environment.

The project started in September 2005 and finished in August 2007.

1.1.4.3 Akropolis

The competences developed in the first two projects led to a range of new possibilities for further functional developments for KM. The second follow-up project, Akropolis, had two main objectives:

- the development of a multi-vector representation for documents
- the semi-automatic identification of metadata for documents

The first objective aimed to obtain a higher level of detail for the representation of documents. In Chapter 6 the developed multi-vector representation reaching this objective is discussed. This representation format delivers a profound change in the capabilities of several existing functionalities and offer opportunities for new features. A search engine, for example, can return more detailed results (e.g. document sections or paragraphs) as an answer to a query. Another example is the possibility of near duplicate detection between a pair of documents on the level of document sections, based on this new representation format.

The second research goal of the Akropolis project was the semi-automatic identification of metadata based on the content of documents (see Chapter 4). In a KMS different so-called document types can be identified. These document types are generic representations of clusters of similar documents. Assigning metadata labels to these document types is a time-consuming interactive task with no exclusion of a subjective selection of labels. After these labels are identified, values for these labels need to be assigned by the creator of a document. The semi-automatic identification of these labels and values reduces the time needed to deploy a KMS. In Section 4.1 this functionality will be explained in further detail.

1.2 Statistical versus Non-statistical Approaches

The research presented in this dissertation is situated in the statistical domain of NLP, in particular the domain of AI. In this section a reasoning is formulated to motivate this choice.

In [Abney et al., 1996] an overview is provided of common problems experienced by non-statistical approaches in the context of language processing. In this section these problems are briefly summarized.

Ambiguity of the language

Ambiguity in language is present in different forms. The ambiguity of the word sense is a well-known issue. Many words have more than one meaning (polysemy), and their meaning depends on the context in which they are used. Ambiguity is also present in the grammar. In [Abney et al., 1996] an ambiguity example is given in the domain of parsing where different parse trees are possible. This problem of identifying the appropriate parse among all possibilities plays a central role in stochastic grammar in computational linguistics.

Dynamics of the language

Every living language is dynamic. Not only the meaning of some words shifts towards a new content, also new words appear and grammar rules evolve over time. These dynamics therefore need to be expressed by an update of the grammar and other semantic information.

Text segmentation

Segmenting a text to identify sections such as words, sentences or paragraphs is often not trivial. Several Asian languages such as Chinese have no single-word boundaries. The characters themselves have a meaning, so several sequences have a different meaning. Identifying the boundaries for sentences or paragraphs logically depends on the definitions of these concepts, which in literature can be found in several formats [Hearst, 1997].

Robustness

The notion of robustness concerns the behavior of an NLP system when perturbations have been made on the input data. This notion is recognized as an important issue in natural language processing [Menzel, 1995].

Scaling

Scalability is the last important issue concerning non-statistical approaches. Many language problems are based on the calculation of the relative frequencies of certain items of the documents, such as n-grams or grammatical associations. The calculation of these relative frequencies results in a large number of combinations of associations that needs to be evaluated. When the corpus size grows, it thus becomes computationally prohibitive to calculate these relative frequencies.

The statistical approach in the domain of NLP applies stochastic and probabilistic techniques on the statistical properties of a document to bypass several of the previously mentioned difficulties. An example of a statistical approach can be found in the context of disambiguation. The longer sentences are, the more ambiguous they become. Applying grammar to these sentences can result in several possible analyses. Several statistical methods for disambiguation exist, such as Markov chains [Boyd-Graber et al., 2007].

1.3 Objectives and Scope of the Thesis

The goal of the research presented in this dissertation, summarized as clustering techniques in Knowledge Management, was subdivided in two main research objectives.

The main research objectives are to develop methods and techniques in the domain of clustering that

- improve the handling of a large collection of documents. This improvement should be reached in terms of computational and spatial complexity.
- improve the accessibility of a document and the information it contains.

Based on these objectives, specific research goals were set in the context of KM.

The objectives that initiated the presented research are:

- the development of techniques in the domain of clustering that support the processing of large, incrementally updated document collections which are characterized by a high dimensional term space.
- the development of a multi-vector representation of a document based on the topics present in its content.
- the development of techniques to (semi-)automatically identify the different general document types and the related metadata in a document collection.
- to enable the adaption of existing, or the creation of new functionalities for end-users to enhance the level of detail.

These different methods and techniques are situated in the context of Knowledge Management, and thus several conditions had to be taken into account in the development phase. These conditions concerned the deployment and the

usability of the methods and techniques.

The developed methods and techniques must adhere to some prerequisites:

- The developed techniques should be readily applicable in terms of space and computational requirements.
- The construction of the document representation model should require as little effort as possible from the administrator or end-user. Ideally these techniques are entirely automated.
- The metadata extraction process should require as little effort as possible for the end-user or administrator.
- The identified metadata should enable the identification of different views on a document collection.
- The techniques should require as little effort as possible for the end-user or administrator.

The resulting methods and algorithms will use the statistical properties of a document in an organisational environment.

The context wherein these techniques should be applicable, are KMSs that are deployed in an unstructured environment (see Section 1.1.3.2).

The input of the KMS is thus limited to an unstructured collection of documents. The documents used in the research processes originate from established benchmark datasets and repositories supplied by professional organisations. Assumed is that the documents were created by so-called knowledge workers. According to Drucker [Drucker, 1969], a knowledge worker is defined as 'the man or woman who applies to productive work ideas, concepts, and information rather than manual skill or brawn.' These limitations imply that the scope is limited to the quantifiable content of documents, assembled in an unstructured collection and written by knowledge workers.

1.4 Structure of the Dissertation

In accordance with the objectives of the research, the content of this manuscript is structured in two main parts:

- Part 1: Clustering processes

- Part 2: Multi-vector representation of a document based on topic identification

The first part of this dissertation gives an overview of the current existing clustering algorithms applicable in KMS environments, summarized in Chapter 2, and describes own contributions in a clustering environment in Chapter 3. Chapter 4 describes the applications for a KMS based on these developments on the domain of clustering.

The second part of this dissertation, concerning the multi-vector representation, has a similar structure as the first part. Chapter 5 provides an overview of the current algorithms for topic identification. The new proposed approach is discussed in Chapter 6. The last chapter describes an overview of the developed applications based on the derived multi-vector representation. The last chapter of the dissertation generally concludes this dissertation while providing an overview of future work, and a summary of the contributions on technological and application level.

The structure of this dissertation is graphically summarized in Figure 1.4.

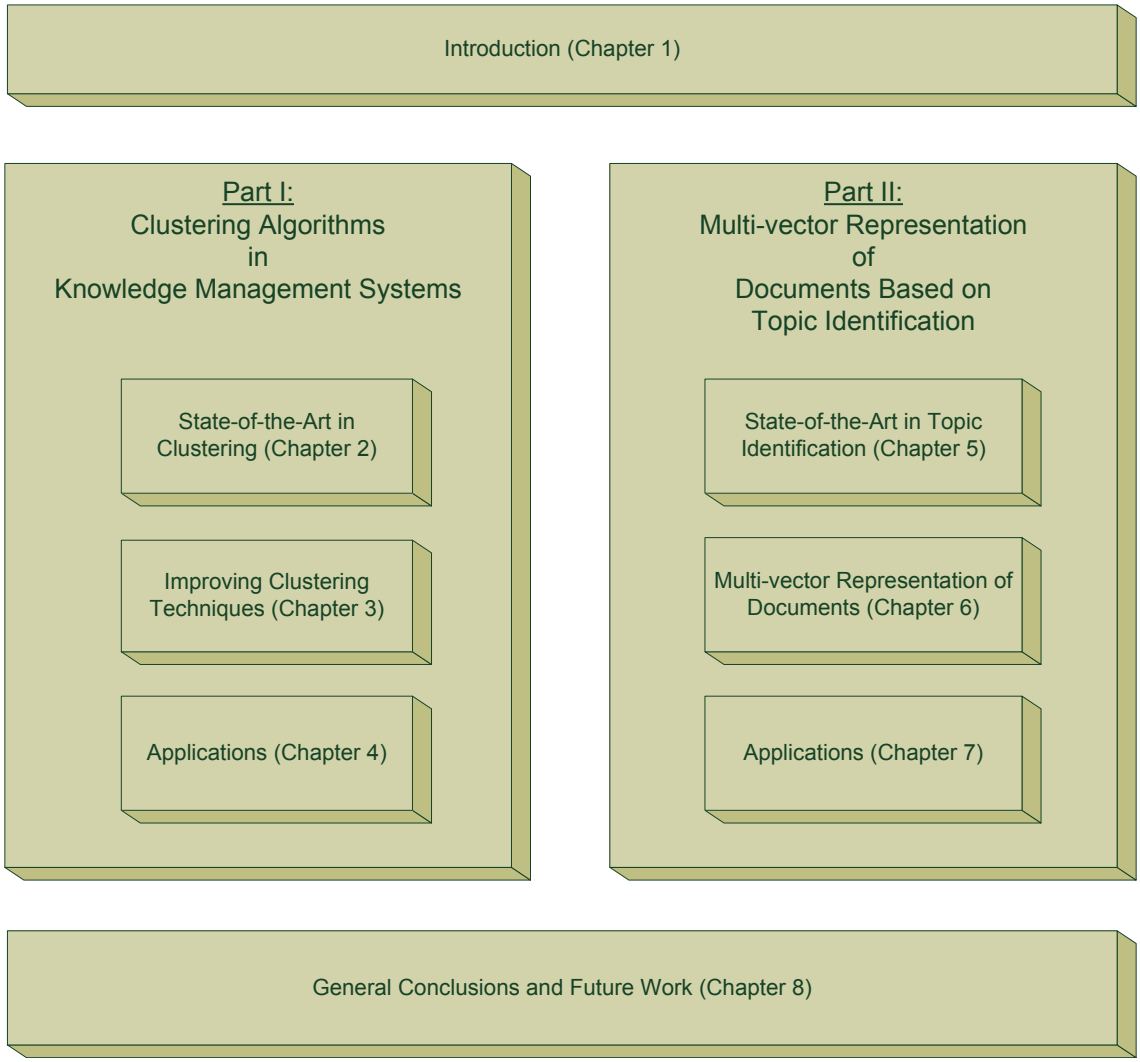


Figure 1.4: Overview of the PhD dissertation structure

Part I

Clustering Algorithms in Knowledge Management

Chapter 2

State-of-the-Art in Clustering

2.1 Introduction

Clustering has a long history as a mental process: mankind always had the tendency to cluster items to obtain a categorization. Early human civilizations e.g. tried to differentiate organic substances in edible and non-edible categories [Mcinnis et al., 1990]. Understanding the human language also relies on clustering: each noun in a language is essentially a label used to describe a class of objects or notions that share specific properties [Beeman et al., 1994].

The creation of the large collection of available cluster techniques boomed in the 1960s, driven by the research of biologists working on numerical taxonomies (i.e. the grouping of taxonomy units using numerical methods) [Sneath and Sokal, 1973]. Since then the statisticians took control, and nowadays clustering techniques are frequently used in a variety of domains such as data mining, geology and medicine.

In recent times the availability of IT infrastructure drastically increased the amounts of digital information, such as websites, e-mail and other documents. This exponential evolution in data storage capabilities makes data collection and storage cheaper and easier. Vast datasets in industrial and other environments are thus more rule than exception. To keep control of such vast collections of unstructured information, several techniques are desirable that are able to reveal one or more underlying structures. Among the possible techniques available in literature, the domain of unsupervised clustering is one possible approach to form the basis of such a classification system. This is a technique to partition data, in which the assessment of the partitions is done without prior knowledge of the underlying structure of the data. The capabilities of several of these methods have

not kept pace with the constant advance in data collection capacity [Berkhin, 2002, Kriegel et al., 2009]. Advances or new algorithms are required to unlock these large datasets or useful information hidden in these datasets will be wasted.

In the first sections of this chapter the cluster problem is defined, and several processing steps are explained that are necessary to develop a clustering environment (Section 2.2). The properties of an ideal clustering algorithm are specified in Section 2.3, followed by a short overview of traditional clustering techniques in Section 2.4. The advances towards the larger datasets are explained in Section 2.5, and the chapter is closed with an overview of measures for cluster quality.

2.2 Clustering Environment

2.2.1 Defining a clustering problem

According to Fraley and Raftery [Fraley and Raftery, 1998], Definition 1 generally describes a clustering problem:

Definition 1 *On the basis of quantitative measurements on a set of patterns, devise a method that assigns the patterns to meaningful subclasses.*

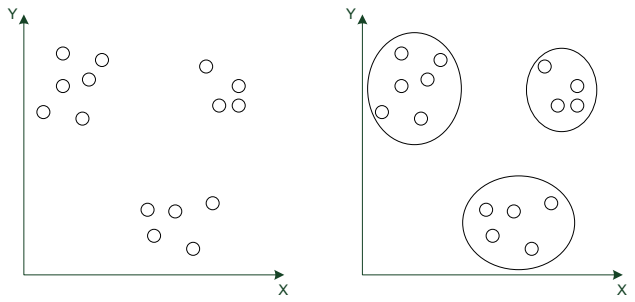


Figure 2.1: Graphical representation of a clustering problem.

A large number of definitions exists in literature that try to capture the notion of clustering. They all share a common basic approach: the action of grouping together similar items or patterns (e.g. documents) into clusters (Figure 2.1), often followed with the interpretation or labeling of these clusters to clarify their meaning. Two general types of clustering algorithms exists that try to provide an answer to the clustering problem: supervised and unsupervised methods. In contrast to supervised methods, no previous knowledge about the pattern collection is required to operate such an unsupervised algorithm. In the remainder

of this dissertation, only the unsupervised clustering algorithms are taken into consideration.

Graepel [Graepel, 1998] provides a more formal definition of a cluster method:

Definition 2 *Let $X \in \mathbb{R}^{m \times n}$ be a collection of data patterns of size n represented in an m-dimensional space $\mathbb{R}^m$. A clustering process is the discovery of the mapping $X \to C$ that partitions the collection X into groups C_K so that patterns x_K in group K are more similar to each other than to other patterns. Each of these K groups is called a cluster.*

The space $\mathbb{R}^{m \times n}$, also called the "measurement space" [Mercer, 2003], is a multidimensional space composed of quantitative variables of a continuous, discrete or categorical nature [Everitt et al., 2009].

In many environments in which a clustering process is performed, this space is not readily available, or at least some filtering steps need to be performed. In Section 2.2.2 these different preprocessing steps in the context of document clustering are explained.

The space C, also called the 'meaning space', is a finite and discrete set of labels. These labels are assigned to the patterns by the clustering algorithms in order to assign them to one or more clusters. The variety of cluster algorithms suggests that several approaches are possible to identify the hidden structure of a collection of patterns.

2.2.2 Conditioning a document clustering problem

To obtain a suitable environment to perform a clustering process, several conditions need to be fulfilled. Not every document is described in the same manner: they are stored as different content types such as lists of words or as full text. Some files only exist in a hard copy format, other are stored in one of the vast number of possible file types. The diversity in input types needs to be translated into one general input format, enabling a quantitative comparison between every pair of documents. Once this transformation is performed, the different steps of the clustering process can be executed. Figure 2.2 provides a graphical overview of the different stages identified in the chosen clustering procedure [Jain et al., 1999].

The first component considers the translation of a document towards a practical representation. The input for this component is a collection of plain text files, so all conversion steps are already performed. In the context of the document clustering, a frequently used representation format is the Vector Space Model (VSM), wherein a document is represented as a vector. The terms become the features of this vector. This is further discussed in the next paragraph. Feature extraction or selection can be performed in combination with the creation of this representation format (see Section 2.2.3). In order to reduce the number of features of a document,

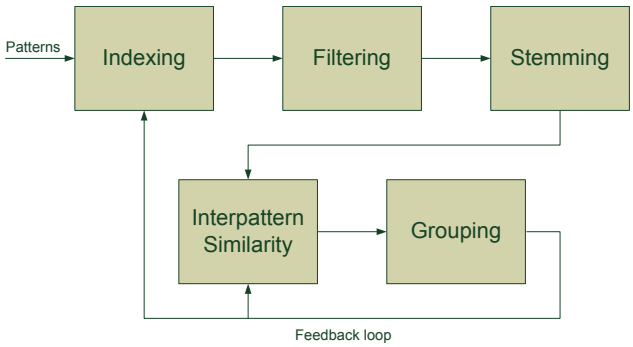


Figure 2.2: Graphical overview of the different stages in the chosen clustering procedure.

stemming is often performed to reach this goal. Obtaining a quantification of the proximity plays a central role in the clustering process. A feedback loop is provided if the grouping process output should affect subsequent preprocessing steps and/or the similarity computations. The next paragraphs present a detailed overview of this process in the context of document clustering.

2.2.2.1 Vector space model

The usage of the VSM as information representation model is a popular approach in the context of IR and text mining [Baeza-Yates and Ribeiro-Neto, 1999]. It takes a collection of plain text documents as input, which is consequently transformed into an "index". This transformation, also called tokenization, omits the structure of the document and breaks the document into a list of terms, or so-called unigrams, which are present in the content of the document. Several languages also contain meaningful combinations of unigrams, so-called multigram words. Examples of such multigram words are 'United Nations' and 'Knowledge Management Systems'. To identify these multigrams, the input texts need to be matched against entries in a lexicon [Turney and Pantel, 2010]. This matching procedure however does not determine a unique tokenization [Sproat and Emerson, 2003]. As stated in the introduction, no semantical information will be used in the developed techniques, and therefore multigrams are considered out of scope. This conversion is also known as the 'bag-of-words'-approach. This transformation results in a vector d_j, represented as:

$$d_j = [f_{1,j}, f_{2,j}, \ldots, f_{n,j}] \tag{2.1}$$

with $f_{i,j}$ the frequency of term i in the document d_j.

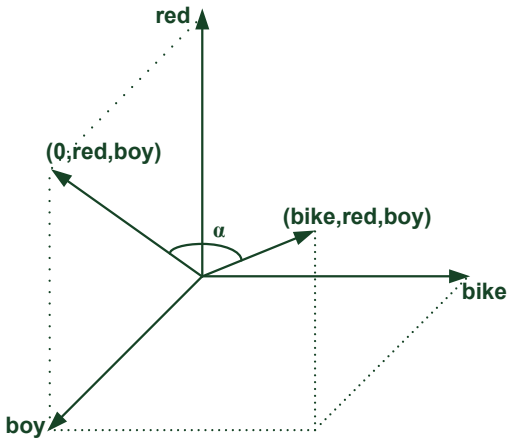


Figure 2.3: Example of a VSM wherein two documents are represented in a three-dimensional term space

This transformation is shown in Figure 2.3. In this figure, a document with the word sequence "The boy on the red bike" is translated into the VSM. The transformation results in a three-dimensional vector $[1, 1, 1]$, reflecting the frequencies of the terms "bike", "red" and "boy", respectively. If another document with another sentence "The red boy" is added to the VSM, the two-dimensional vector $[0, 1, 1]$ appears. Not all words are translated into the VSM, such as 'the' in the previous example. This filtering is further discussed in Section 2.2.3.2.

When this transformation is performed for the complete document collection, this index is reformed to a (very large) Term-by-Document Matrix (TDM) wherein every column represents a document. The rows are defined by the vector space dimensions, where each dimension represents a unique term or feature. The complete list of terms present in the document collection is called a vocabulary. As such, the documents effectively become vectors in the defined vector space. The resulting index or matrix representation for the example described in this section is depicted in Figure 2.4.

If more documents are added to the collection, additional rows and columns are included in the matrix. This is shown in Figure 2.5.

As the length of the documents frequently differ, this transformation often results in vectors of different length. To account for this difference, often a transformation called normalization is performed. This normalization divides the vector entries by the vector length. For the length of a vector, the 2-norm is often selected [Jain et al., 1999].

The strength of the conventional VSM lays in its simplicity. One of the

	Doc 1	Doc 2
bike	1	0
boy	1	1
red	1	1

Figure 2.4: The resulting term-by-document matrix with 3 terms describing 2 documents

	Doc 1	Doc 2	Doc 3	...	Doc m
bike	1	0	0	...	0
boy	1	1	1	...	0
red	1	1	0	...	1
car	0	0	2	...	0
...	...	...	...	...	...
nth term	2	0	1	...	2

Figure 2.5: A term-by-document matrix with n terms describing m documents

major drawbacks is that it removes the correlations between the terms in the documents due to the imposed orthogonality of the term space [Silva et al., 2004]. The documents are represented by the extracted terms, and the relationship among them is not considered. Due to this oversimplification, relationships such as polysemy and synonymity are omitted in the vocabulary. As will be shown in Section 2.5.2, correlation between terms becomes an important factor in high dimensional environments. To account for this loss of correlation, weighting schemes, which are described further, are often applied to counteract this disadvantage.

Another major drawback is the poor representation of long documents in function of similarity analysis (large dimensionality and a small scalar product for the dissimilarity measurement). The presented multi-vector representation technique in Chapter 6 counteracts this drawback. Because of the statistical nature, no semantics or semantic word relations are considered. Similarity based on synonyms is thus not possible.

In literature other representation formats can be found, based upon this classic vector space model:

- *Generalized Vector Space Model* [Wong et al., 1985]
  The main idea of this type of vector space model is to incorporate Boolean algebra into a vector space. The terms are represented as a linear combination of pairwise orthogonal vectors associated with the atomic expressions. Such an atomic expression Z is a conjunction of the terms t_i in which each term appears once in a complemented or uncomplemented form (i.e. $Z = c_1$ AND c_2 ... AND c_k with $c_i = t_i$ or $\neg t_i$). This separate, corrective procedure is used to take the correlations between terms into account.

- *Topic-based Vector Space Model*
  In contrast to the classical VSM, this vector space is defined by fundamental topics, i.e. every dimension represents a so-called fundamental topic. These topic vectors are assumed to be orthogonal and independent of each other. An example of this space is shown in Figure 2.6, where three topics are defined. The synonyms are grouped according to their covering topic, as indicated in this figure. Every term is represented as a vector with a specific weight for each of the topic vectors. A term weight, with a value between 0 and 1, is the algebraic length of the term vector. The direction of the vector indicates the relevance according to the fundamental topics. A document is then represented as a vector which is the sum of the vectors of its terms, as shown in Figure 2.7. The procedure however to obtain the term vector lengths and the angles defining the space is not formally defined, thus limiting the applicability of the model [Polyvyanyy and Kuropka, 2007]. An approach to (semi-)automate this procedure can be found in [Becker and Kuropka, 2003].

- *Enhanced Topic-based Vector Space Model* [Polyvyanyy and Kuropka, 2007]
  This improved vector space model tries to solve the already mentioned problems related to the creation of the term space by employing ontologies. These ontologies are obtained from Wordnet [Miller et al., 1990].

2.2.2.2 Weighting schemes

As previously indicated, not all terms in a document contain the same amount of information. By using a weighting scheme, this difference in informational value is expressed. The objective of the application of a weighting scheme is to assign the highest values to the terms that represent the highest informational value in the content of a document. Low weights are assigned to terms that are less important. Examples of low-weighted terms are stopwords and numbers.

One of the easiest weighting schemes is the *Binary weighting scheme*. This weighting scheme assigns a 1-value to a term if this term occurs in a document, and a 0-value otherwise.

The more advanced weighting schemes are based on a heuristic approach. There exists a multitude of weighting schemes [Everitt et al., 2009], but all of

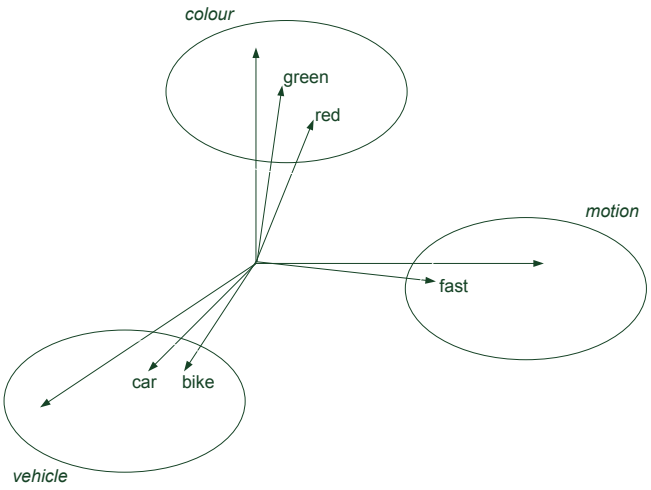


Figure 2.6: Visualization of a topic-based vector space

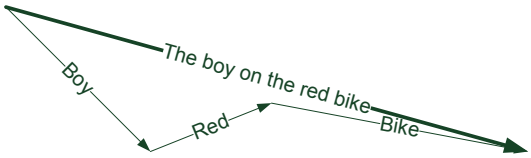


Figure 2.7: Graphical representation of a document in a Topic-based Vector Space Model

them have in common that they allocate a weight equal to 0 to dimensions that do not occur in a specific document. Most weighting schemes are composed of a local and global component. The local component describes the informational share of the word in the content of a document. The global component describes this share among all words in the global document collection. Popular weighting schemes are:

- Term Frequency Inverse Document Frequency (TFIDF):
  The weight $w_{i,j}$ of a term i in document j therefore is

$$w_{i,j} = f_{i,j} \cdot \log(\frac{n}{n_i}) \tag{2.2}$$

  wherein the Term Frequency (TF)-component $f_{i,j}$ is the number of occurrences of a term i in document j. n is the number of documents,

and n_i is the number of documents that contain term i. The second factor of this equation is also called the Inverse Document Frequency (IDF).

- Okapi BM25 scheme [Robertson et al., 1992]:
  The weight w_i of a term i in document j is defined as

$$w_{i,j} = \frac{(k_1+1)\cdot f_{i,j}}{(k_1[(1-b)+b\cdot(\sum_{i=1}^{m_j} f_{i,j}/\sum_{j=1}^{n}\sum_{i=1}^{m_j} f_{i,j})]+f_{i,j})} . log(\frac{n-n_i+0.5}{n_i+0.5}) \tag{2.3}$$

  $\sum_{i=1}^{m_j} f_{i,j}$ and $\sum_{j=1}^{n}\sum_{i=1}^{m_j} f_{i,j}$ are the number of terms in document j and the average size of a document in the collection, respectively. b is the document normalization factor of the weighting scheme and has a default value of 0.6 to 0.75. k_1 is defined as a scaling constant. The second factor in this weighting scheme is also known as the relevance feedback factor, as this weighting scheme was originally used for relevance ranking of documents in search engines [Manning et al., 2008].

2.2.2.3 Measurement of proximity

The representation of a document as a vector enables the application of techniques of the domain of Linear Algebra in the space spanned by the domain vocabulary [Baeza-Yates and Ribeiro-Neto, 1999]. This vector representation also enables the introduction of the notion of proximity. Calculating proximity or distance between pairs of documents is essential to reach the objective of a clustering process (Section 2.4). These distance measures are called dissimilarity measures if the condition of triangular inequality is not met [Trefethen and Bau, 1997]. Unless it is explicitly mentioned differently, it is assumed that this condition is fulfilled in the course of this dissertation.

In literature, several measures are described. For instance, McGill [McGill, 1979] alone lists more than 60 different similarity measures. However, not all proximity measures are applicable in each environment. A general distinction is made between categorical and continuous features. In the context of document clustering, the features adhere to the second type.

The most popular distance measure in the context of document clustering is the cosine distance measure [Baeza-Yates and Ribeiro-Neto, 1999]. This measure between two document vectors d_1 and d_2 is defined as

$$d(d_1, d_2) = 1 - cos(\alpha) \tag{2.4}$$

where α is the angle between the two vectors. The range of this angle is constrained to the range of $0-90$, so the range of the distance measure is 0 to 1. The cosine of the angle is calculated as

$$cos(\alpha) = \frac{d_1 \cdot d_2}{\|d_1\| \cdot \|d_2\|} \tag{2.5}$$

with $d_1 \cdot d_2$ as the dot product between the two vectors.

In Section 3.2.3 a variant on the cosine distance measure is explained that, applied in a clustering context, results in a better cluster performance. In a text based environment Pearson correlation [Everitt et al., 2009], Jaccard coefficient [Jain et al., 1999] or Euclidean similarity [Korenius et al., 2007] have also proven their merits in this context.

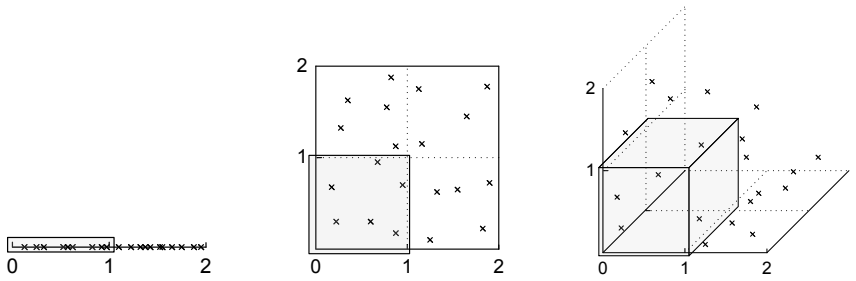
2.2.3 Dimensionality reduction

Because only few terms appear in all documents, the TDM is very sparse. This property of the term-by-document matrix can be exploited to generate a representation of the matrix that uses little memory. This is often necessary because the vocabulary that is typically used in a set of documents can easily contain several thousands of terms. The resulting high dimensionality of the vector space has some important consequences. First of all, it leads to what is known as the beforementioned 'curse of high dimensionality', which is described in the next section. Secondly, high dimensionality makes calculations in the vector space computationally expensive. To counter these dimensionality-related issues, a preferable step in the phase of several information retrieval processes, is the reduction of the number of dimensions or features. This reduction can be obtained by selecting features when the term by document matrix or index is constructed, the so-called feature filtering. The other group of techniques, known as feature extraction techniques, performs one or more matrix operations on the term-by-document matrix. Common objective of these techniques is to improve classification performance and/or computational efficiency [Jain et al., 1999]. In very high dimensional environments, the performance is however questioned: clusters are often situated in different subspaces, and these global reduction techniques do not retain this underlying structure. [Kriegel et al., 2009]. In literature several approaches can be identified to filter the list of terms extracted from a document to obtain a reduced number of dimensions. In the following paragraphs the curse of high dimensionality is summarized, followed by a review of techniques, applicable on the features of documents, that try to counter this phenomenon.

2.2.3.1 Curse of high dimensionality

The general phenomenon 'curse of high dimensionality' refers to the exponential increase of volume as a function of its dimensionality [Parsons et al., 2004]. Figure 2.8 illustrates the influence on the density of a space containing a small collection of items when the number of dimensions increases. In the original one-dimensional space (Figure 2.8(a)), the items are randomly placed in the interval $[0, 2]$. As a result, a dense grouping is obtained, and nine items are located in a unit line segment. This selected unit line segment is indicated with a greyed box. Adding

one dimension to this space, decreases the overall density of the space (Figure 2.8(b)). The items are spread across this new dimension, resulting in six items located in a unit square. If a three dimensional space is constructed, only five items are situated in the unit cube (Figure 2.8(c)).



(a) 9 items are located in the unit line segment

(b) 6 items are located in the unit square

(c) 5 items are located in the unit cube

Figure 2.8: Schematic overview of the influence of the curse of high dimensionality. Adding one dimension to an existing space spreads the points across the new dimension, increasing the distance.

This curse, identified by Bellman [Bellman, 1961], has a serious consequence in the context of clustering. If a structure is embedded in a collection, the number of items necessary to reveal this structure is exponentially related to the increasing number of dimensions. Often this number of items is not available. The consequence is, when using a distance measure, that all items are approximately equally far away from each other, making a distinction based on this proximity impossible and making the notion of distance, as a central role in clustering, increasingly meaningless. The resulting effect on the clustering process is a decline in the performance of these algorithms to distinguish items that are closer to each other than to the other items. This effect often occurs in the context of document clustering, because adding a document to a collection often considerably increases the number of terms.

As is indicated in the following sections, a prior reduction of dimensionality becomes highly favorable [Otoo et al., 2001]. This reduction can serve as an answer to the curse of high dimensionality and the previously mentioned computability issue of such collections. A third important reason for this reduction is the visualization of the items and the clustering result. This reduction, in the most extreme situation to two or three dimensions, offers the possibility to graphically summarize these document repositories and their structure. However, every reduction has its price: a severe reduction of dimensionality can destroy important properties or structures in the data and finally lead to loss of information [Kriegel et al., 2009].

2.2.3.2 Feature filtering

It can become computationally prohibitive to consider the complete vocabulary in the further process, as indicated in the previous section, but also not all terms are as important as others. This last aspect is covered in Section 2.2.2.2. Some of the terms are almost completely superfluous, due to several reasons. Examples are ambiguous words due to the context (e.g. terms completely composed of numbers), irrelevant words (stopwords) and long terms due to conversion errors (e.g. omitted white spaces). To account for this, several filtering steps can be performed to reduce the computational burden. This feature filtering phase is in fact the first step of the feature selection, as further explained. In the following paragraphs several frequently used feature filtering steps are discussed.

Language recognition

A first approach to reduce the number of dimensions, is the identification of the language of the document or part thereof. This technique has an indirect influence on the dimensionality because it does not actively reduce the number of dimensions. By identifying the language of every document (part) separate matrices can be constructed for every language instead of one large multilingual matrix.

One of the more popular methods to identify the language of a document is based on the stopwords used in the document [Jain et al., 1999]. For every language a set of stopwords is created that uniquely identifies the language. Cross lingual words such as "in" (Dutch - English - German) or "of" (Dutch - English) are thus omitted. In Appendix D.1 a selection of stopwords is displayed for the languages Dutch, French, German and English. Another popular language recognition technique uses discriminating character n-grams (i.e. a sequence of n characters) to identify a language [Dunning, 1994]. The technique assumes that languages have certain strings which only, or frequently, occur in that language, such as "eux" in French or "ery" in English.

Language recognition also plays an important role in the filtering steps described in the following sections.

Stopword and other filtering steps

The second approach in the filtering process is to omit the stopwords from the document. The generally accepted motivation is that these words convey few information and have no positive influence on the final results of several IR techniques [Mladenic and Grobelnik, 1999, Craven et al., 2000]. Every term is screened to verify whether it appears in the stopword list of the corresponding language. These lists easily contain more than 2000 words. In literature, however, other research recently indicates the presence of valuable information in some stop words [Cordeiro and Brazdil, 2004]. These stop words are used in commonly used expressions (e.g. stopword "for" in a price indication: two items for three euros).

This identification, however, requires semantic information which, as stated in the introduction of this dissertation, is not taken into account in the developed techniques.

Other filtering steps are also included in the construction process of the matrix:

- *Value and other numerical words*
  Values and other words containing any numerical reference contribute ambiguous information to the content of the document because the structure of the document is omitted in the conversion to the bag-of-words format

- *Word length*
  Words that have an exceptional length, such as only one or two characters or more than thirty characters are suspected that they do not convey much information. Either they are rare words or misspellings.

- *Single-occurrence words*
  The filtering of single-occurrence words is important in the context of document clustering (see Section 2.4). These words have no influence on the structure identification process. For describing the content of a document however, these words can contain substantial information.

Stemming

The last approach to reduce the dimensionality by feature filtering is the projection of every word to its stem. A stem is defined as the part of a word that is common to all its variants obtained by inflexion [Kroeger, 2005]. Terms with a common stem will usually have similar meanings. An example of such a group of words is 'connect', 'connection' and 'connecting', which are mapped to the stem 'connect'. The process of stemming is performed by a suffix stripping algorithm. The most popular stemming solution is the Porter Stemming Algorithm [Porter, 1980]. This algorithm was originally designed for the English language, but currently several language variants exist. The reported reduction on the size of the vocabulary, obtained by the application of this stemming algorithm, is on average 33% [Orengo and Huyck, 2001].

2.2.3.3 Feature selection

The previously described feature filtering steps are part of a larger group of techniques, called feature selection. To reduce the size of this vocabulary, only these terms or features are withheld by the feature selection techniques that are considered to be relevant or informative for the process. The most intuitive approach in this context is the already mentioned use of a stopword list. This list can be extended to these features or terms that are domain specific and therefore do not carry much relevant information. An example is the presence of a company template in the collection. Because several documents contain one or

more occurrences of the company name, this name becomes irrelevant for further processing. Therefore the list can be extended with this term. This approach however requires a considerable a-priori knowledge of the collection to create such an application-specific list. This manual restriction is therefore not recommendable as a general approach to reduce the dimensionality.

Other techniques exist that try to estimate the information value of the features of the documents. A first example is the Zipf-curve approach. This Zipf-curve, which is discussed in more detail in Section 3.3, is the result of sorting the domain vocabulary in order of descending frequency. The obtained Zipf-curve is composed of three zones: the first zone, containing the most frequently used terms, is considered to contain non-relevant terms. If no further filtering is applied to the collection, this part would contain almost all stopwords and domain specific words. The second zone is the collection of terms that moderately occur throughout the collection. This zone is considered to contain the most interesting terms that allow to differentiate documents based on their occurrence. The third and last zone is, as the first zone, containing non-relevant terms. These low-frequency terms are often misspelled words. These terms can be omitted from further processing. The Zipf-curve and the three zones are displayed in Figure 2.9.

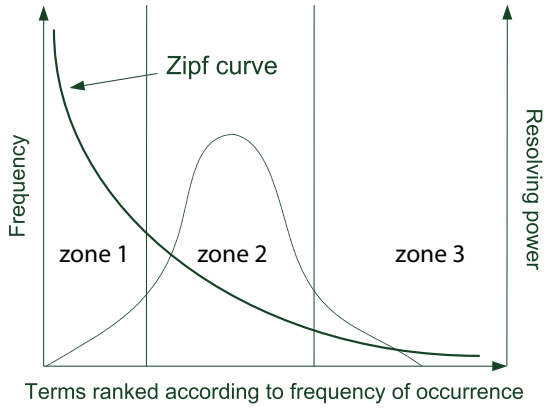


Figure 2.9: Graphical representation of the significance interval

Weighting schemes are also used to obtain a further restriction on the number of features. This selection is then based either on local weighting, or local and global weighting depending on the composition of the weighting scheme. TFIDF will provide a feature selection based on the term frequencies in the document, as well as on the IDF component considering the complete document collection.

2.2.3.4 Feature extraction

Feature extraction is based on a linear or non-linear function of the set of original features. Several techniques of the collection of linear feature extraction techniques are rooted in the concept of eigenvectors and eigenvalues of the Document-by-Term Matrix (DTM) of the original feature space. Examples of this category are the Principal Component Analysis (PCA), or the related Singular Value Decomposition (SVD) and Latent Semantic Indexing (LSI). In the next paragraphs, these feature extraction techniques are briefly explained.

Principal Component Analysis Often called as one of the most valuable results of the applied algebra, PCA is a frequently used dimensional reduction technique. This technique is an orthogonal linear transformation that constructs a new coordinate system wherein each coordinate, or principal component, describes the variance of the projected data on these new coordinates. The principal components are ordered in such manner that the first principal component accounts for the largest variance of the projected data, while each of the succeeding components accounts for as much of the remaining variance as possible.

Mathematically, PCA is defined as

$$Y = \Psi X \tag{2.6}$$

wherein X represents the (mean-centered) data in the original coordinate system. The rows of the matrix Ψ contain the eigenvectors of the covariance matrix of the data in the original coordinate system. The columns of matrix Y contain the principal components.

Singular Value Decomposition Another important dimensional reduction technique, closely related to the PCA [Wall et al., 2003], is the SVD [Skillicorn, 2007], which decomposes a DTM D with rank r into the product of three matrices:

$$D = U \cdot \Sigma \cdot V^{'} \tag{2.7}$$

Σ is the diagonal matrix containing the singular values σ_i of the matrix D with rank r, with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r$ and $\sigma_i = 0$ with $i > r$. The singular values are the square roots of the n eigenvalues of the matrix $D \cdot D^{'}$. These singular values σ_i represent the amount of variation along the i axis. The matrix T and D are orthonormal and respectively contain the left and right singular vectors of the matrix $D \cdot D^{'}$. These matrices U and V represent the terms and documents in the transformed space.

From this decomposition a lower rank approximation D_l can be obtained by truncating or reducing the number of singular values to l. The lower rank approximation can be obtained by using the following formula:

$$D_l = U_l \cdot \Sigma_l \cdot V_l^{'} \tag{2.8}$$

The matrix Σ_l is created out of Σ by replacing the $r - l$ smallest singular values with zeros. U_l and V_l are respectively the left and right singular vectors of the l largest singular values. Following the Eckart and Young theorem [Eckart and Young, 1936] this lower rank approximation is the closest $l-$rank approximation in least square sense. This approximation is the best square approximation of D by a matrix of rank r based on the 2-matrix norm [Trefethen and Bau, 1997].

Latent Semantic Indexing The mathematical technique of LSI [Deerwester et al., 1989] is an application of the SVD technique. The lower previously obtained rank approximation of matrix D serves as the base of the LSI technique. In Equation 2.8, U_l is called the term-concept matrix, Σ_l the singular values matrix, and $V_l^{'}$ the concept-document vector matrix. The LSI technique assumes that an underlying or latent structure in word usage is present in a document, and that this structure is partially hidden due to the variability in word usage. Performing the SVD calculation, combined with the dimensionality reduction, has the effect of estimating this latent structure. It preserves most relevant semantic information in the documents while suppressing the noise and other unwanted artifacts of the original term space of the matrix D. The LSI method aims to identify associative patterns in the relationships between so-called concepts and terms appearing in documents. Therefore an important feature of LSI is its ability to extract the conceptual content of a body of text by establishing associations between those terms that occur in similar contexts. A query vector q entered by a user can then be compared to the document vectors using the following equation:

$$q_{red} = q^{'} \cdot U_l \cdot \Sigma_l^{-1} \tag{2.9}$$

The matrix Σ_l is a diagonal matrix containing the singular values. q_{red} is the query vector in the reduced term space. Using a distance measure and a predefined threshold, the closest documents to the q_{red} vector can be returned as a result to the user. The advantages of the LSI technique, as discussed in [Rosario, 2000], are

- Synonymy
  Due to the latent structure assumed to be present in the text, a concept is often described by different terms. This enhances the identification of terms describing a similar topic.

- Polysemy
  In contrast to synonymy, polysemy is the group of words that have multiple meanings. The presence of these words hampers the information retrieval activities. LSI eases this by identyfing the rare and less important usages as noise.

- Term dependence
  As indicated in Section 2.2.2.1, a VSM assumes no correlation between the terms. The strong associations between words in a language is better approximated in a LSI space.

2.3 Ideal Clustering Algorithm

In literature a vast collection of clustering algorithms can be identified, and this collection is fast growing due to the apparently infinite storage capacity of computer networks nowadays and the resulting new opportunities emerging from this evolution [Jain et al., 1999, Kriegel et al., 2009]. Frequently these algorithms are focused on a few application areas, and they are not capable to adequately perform in other domains. Several algorithms, for example, require a-priori knowledge of the collection to estimate the expected number of clusters or the dissimilarity measure that needs to be used.

The following derived list of requirements are the desired properties for an ideal clustering algorithm for text mining purposes [Zaane et al., 2002, Han, 2005]:

- *Scalability*
  The algorithm should be able to operate on different sizes of input collections, and the performance and quality of the clustering results should at most decrease linearly with an increasing data size.

- *Dealing with different types of attributes*
  The attributes or dimensions of an item can be of a diverse set of data types. These data types can be either boolean, categorical or numerical. One item can also contain different attribute types.

- *Discovering clusters with arbitrary shape*
  The ability to discover clusters with an arbitrary shape is heavily dependent on the used dissimilarity measure. Using the Euclidean distance measure, for example, favors the identification of spherical cluster shapes.

- *Minimal required domain knowledge to determine input parameters*
  A-priori required knowledge of the content of the collection has a baleful influence on the unsupervised and automatic process that clustering should be. The most frequently required key parameter is the number of clusters. Others include an initial seed of cluster centers and one or more thresholds for cluster density or to halt the clustering process.

- *Ability to deal with noise and outliers*
  Noise is present in all practical problems. An ideal clustering algorithm should be able to perform adequately even in the presence of a substantial noise level.

- *Insensitivity to order of input records*
  The algorithm should provide consistent results regardless the order of the input of the items.

- *Scalability to high dimensionality*
  The ability to handle high dimensional datasets requires dealing with the curse of high dimensionality. This property is currently at the basis of significant research efforts (see Section 2.5).

- *Interpretability and usability*
  The algorithm should produce clear and interpretable results, so that the end-user can draw appropriate conclusions from the presented clustering results.

In literature, to the best of our knowledge, no single algorithm has been identified that can fully satisfy all these requirements. For a practical application, the selection of a clustering algorithm should be based on its properties and capabilities. Often, multi-mode clustering approaches (combining several techniques) are used to approximate the above requirements.

2.4 Traditional Clustering Techniques

Several approaches can be used to create a general classification of the vast number of cluster algorithms that exist nowadays [Jain et al., 1999, Anderberg, 1973].

Clustering algorithms have different properties based on the following clustering types:

- *Hierarchical or flat clustering*
  A hierarchical clustering algorithm constructs a tree structure representing nested clusters at different levels of detail. This structure allows the identification of several clustering results. A flat or partitional clustering iteratively reassigns items to clusters in order to optimize an objective function. These algorithms determine a structure containing all clusters at once.

- *Iterative clustering*
  An iterative cluster algorithm, such as K-means, iteratively reallocates items to obtain a better clustering result. Other algorithms, such as hierarchical clustering techniques, do not exhibit this characteristic.

- *Hard or soft clustering*
  The clustering algorithm can label items according to the cluster to which they are classified. If cluster membership is expressed in a boolean variable, the clustering is considered to be hard or crisp. An item can thus belong at

most to a single cluster. If the variable can take values in the range $[0, 1]$, the process is considered to be soft clustering. Soft clustering thus assigns a degree of cluster membership to each item.

- *Incremental or non-incremental clustering*
  Incremental clustering algorithms update the clustering result each time a new item is added to the set. In contrast non-incremental clustering algorithms assume that the full collection is available at the start of the clustering process. Adding a new item requires full reprocessing of all items.

- *Deterministic or stochastic*
  The notion of stochastic refers to the fact that the assignment of items to clusters is based on stochastic processes. In contrast to deterministic clustering, repeated clustering processes on a document collection thus frequently result in a different clustering outcome.

In the following sections an overview will be provided on different types of clustering algorithms. Many algorithms represent clusters by means of centroids or medoids. A centroid of a cluster is defined as the fictive 'average' item of a cluster. A medoid is an item for which the average dissimilarity to all items in the same cluster is minimal.

2.4.1 Partitional or flat clustering

A partitional or flat clustering algorithm identifies a single partitioning of the collection. These iterative algorithms generally adopt the following approach: after an initial assignment of the items to clusters, the algorithm iteratively reassigns items to other clusters with the objective to optimize a criterion function. Based on computational reasons a heuristic approach to optimize this function is preferred. Exploring all possible k partitions C for a collection with n documents requires the processing of a huge number of possible partitions [Mercer, 2003]:

$$C(n,k) = \frac{1}{k!}\sum_{i=1}^{k}(-1)^{(k-i)}\binom{k}{i}i^n$$

This number is clearly computational prohibitive. Besides the heuristic approach, often resulting in the identification of a local optimum, the iterative approach of using different initialization configurations is often used in an attempt to reach the global optimum [Jain et al., 1999].

2.4.1.1 K-means

One of the most popular applied clustering algorithms is the K-means method [MacQueen, 1967, Hartigan, 1975]. The objective function of the K-means method

is to minimize the within-cluster distances, defined as the distance between an item and the centroid of its related cluster. The algorithm is composed of the following steps:

1. Determine or choose the number of clusters k
2. Randomly create k clusters and calculate theirs cluster centers, or select directly k random items as initial centroids.

while *not converged* **do**
  Assign each item to its nearest centroid ;
  Recalculate the centroids;
end

The optimization criterion function of the K-means algorithm is the squared error criterion [MacQueen, 1967]. Often applied convergence criteria monitor the number of reassignments; or a decrease in squared error below a predefined threshold. The algorithm performs well for collections with isolated and compact structures.

2.4.1.2 Variations on K-means

A large number of variations of the original K-means algorithms has been reported in literature. The variations are situated in

- the manner of selecting the initial seeds [Forgy, 1965, Astrahan, 1970, McRae, 1971] or initial partitions [Forgy, 1965, MacQueen, 1967, Wolfe, 1970]
- the representation of every cluster, e.g. centroid or medoid [Anderberg, 1973]
- hard assignment of an item to a cluster, opposed to fuzzy assignment [Anderberg, 1973]
- the time point at which the update of the centroid or medoid occurs [Forgy, 1965, MacQueen, 1967, Jancey, 1966]
- the number of iterations in the reallocation process [MacQueen, 1967]
- variable or fixed number of clusters [Ball and Hall, 1965, MacQueen, 1967, Wishart, 1969]

As already indicated, a large collection of examples for these variants exists. The K-medoids approach is one of the more popular variants, because the notion of centroid is often not applicable or interpretable. Instead of using a centroid as the representation of a cluster, K-medoids uses medoids to characterize the different clusters. ISODATA is another well known variant of the

K-means algorithm [Ball and Hall, 1965]. This self-organizing algorithm allows the automatic adjustment of the number of clusters during an iteration by merging similar clusters; or splitting clusters with large standard deviations. The number of input parameters required for this algorithm is larger compared to the original K-means procedure.

An example of fuzzy assignment is the Fuzzy K-means or K-harmonic means. The objective function used by this algorithm is based on the harmonic mean of the distance of each item to its cluster center. This function rewards items located near any centroid [Hamerly and Elkan, 2002]. [Anderberg, 1973] and [Jain et al., 1999] are useful references for more information on these topics.

In recent times, an ongoing interest can be observed for the K-means approach, due to the usability of the algorithm. This is especially valid in the domain of Bioinformatics, where the renewed interest for K-means resulted in several recent variants. The disadvantage that K-means might converge to a local optimum, depending on the initialization process, led to a number of genetic algorithms, such as the Fast Genetic K-means Algorithm and the Incremental Genetic K-means Algorithm [Lu et al., 2004]. Other evolutionary approaches are suggested to bypass the problem of the local optimum [Jain et al., 1999].

2.4.1.3 Graph-based clustering

Instead of performing a conversion of items to a set of vectors or other models, the items and their relationships can also be stored in a graph structure. A graph structure is formed by a set of nodes and edges, where nodes represent the items and (weighted) edges describe the relationship between the items. In the context of graph structures, no single definition of a cluster is generally adopted. The loosest possible definition of a graph cluster is that of a connected component in which any pair of nodes is reachable by a path of edges. The most stringent is that each cluster should be a maximal clique in which every pair of nodes is connected by a single edge [Schaeffer, 2007]. Further on in this dissertation, a cluster in graph context is considered to be a connected component.

To identify these components, graph partitioning techniques are used. One of the best known techniques is the construction of the minimal spanning tree. Starting from this tree, removing the $k - 1$ links with the lowest similarity results in a partitioning into k clusters. A large group of partitioning techniques can be categorized as cut-based partitioning techniques. The objective of these techniques is to divide the graph into k disjoint subgraphs such that the sum of the edges that are cut is minimized. Spectral graph partitioning is frequently used to reach this objective, which is discussed in more detail in Section 6.2. A comprehensive overview of this type of partitioning is given in [von Luxburg, 2007, Schaeffer, 2007]. A profound survey of graph clustering can be found in [Schaeffer, 2007].

2.4.2 Hierarchical clustering

Hierarchical clustering algorithms differ from the partitional clustering approach in the creation of a hierarchy summarizing the similarity structure between the items. The construction of this tree, also called dendrogram, is based on an algorithm called linkage. This algorithm calculates distances between clusters in a hierarchical clustering process. In literature five popular linkage algorithms can be identified [Anderberg, 1973]:

- *Average linkage*
  The dissimilarity between two clusters is calculated based on the notion of average distance. This measure is defined on the average distance between each point of the first cluster and all points in the other cluster. The two clusters having the lowest average distance are candidates for the merging process.

- *Centroid linkage or Unweighted Pair-Group Method using Centroids*
  This linkage is based on the distance between the two centroids of the pair of clusters.

- *Complete linkage*
  This linkage criterium defines the linkage between two clusters as the greatest distance between two items, one of both clusters. This criterium tends to produce very tight clusters of similar cases.

- *Single linkage*
  Opposed to complete linkage, single linkage between two clusters is defined as the minimum distance between two items, one of both clusters. This linkage criterium tends to create long chains resulting in sparse and ragged clusters.

- *Ward's linkage*
  This linkage is defined as the total sum of squared deviations from the centroid of a cluster. The two clusters that merge, cause the smallest increase in this sum.

An example of a dendrogram is shown in Figure 2.10. In this dendrogram 7 items are linked together based on a linkage criterium. The X-axis indicates the merging time, so by reading from left to right the total clustering process can be examined. In the example singletons 6 and 7 are combined first to form a non-singleton cluster. In the next step item 5 is appended to this cluster. This process is repeated until all items are grouped into one large cluster. This type of hierarchical clustering is called agglomerative clustering. In the opposite approach, called divisive clustering, the starting configuration is one large cluster containing all items. In the following steps, the cluster having the lowest cluster linkage is

split into two clusters. This process repeats until only singleton clusters remain in the collection.

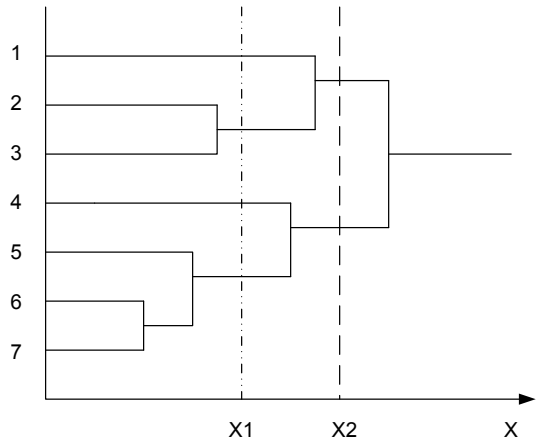


Figure 2.10: Example of a dendrogram tree

Cutting the dendrogram at a certain point along the X-axis, corresponds to an intermediate clustering result. In Figure 2.10 two cuts are shown. The first stripe-dotted cut $X1$ results in four clusters, of which two are singleton clusters. The second (striped) cut $X2$ results in only two clusters, i.e. $\{(1, 2, 3\}$ and $\{4, 5, 6, 7\}$.

2.4.3 Mixture resolving or probabilistic clustering

This collection of clustering algorithms assumes that the items are generated by a statistical model, so these items are considered to be a random sample of several probability distributions. Such clustering algorithm thus tries to recover these original models, and governs the identification of the clusters and the assignment of the items to the clusters. The region around the mean of each (supposedly unimodal) distribution constitutes a natural cluster. The initial selection of a model therefore requires substantial knowledge about the structure of the data, and thus does not comply with the requirements for an ideal clustering algorithm as described in Section 2.3. To elevate this disadvantage, in several algorithms it is assumed that the individual clusters originate from a Gaussian distribution [Jain et al., 1999].

To identify these clusters, the algorithms start from the assumption that the set of parameters of every model is unknown. Through iteration, the values of these parameters are updated in order to obtain the best possible approximation.

This clustering approach has the following important properties [Berkhin, 2002]:

- the interpretation is facilitated by the use of the predefined models.
- the ability to identify complex structures.
- the cluster representation is independent of the representation of the items. Because the density of the points is converted to a probability function, the representation of a cluster is independent of the representation of an item.
- the fast convergence of the parameter set towards the optimal setting.

The Expectation-Maximization (EM) algorithm [Dempster et al., 1977] is a commonly used probabilistic clustering algorithm. The selection of the parameters for the model is based on the criterion of maximum likelihood. The EM algorithm is often applied to analyze collections containing missing or incomplete data or features.

In this context, the EM algorithm is composed of two phases: the algorithm starts by estimating the missing data (E-phase) followed by estimating the different parameters of the model by means of maximum likelihood (M-phase). Mathematically this likelihood can be defined as

$$f(\phi) = \prod_{i=1}^{n} \zeta(x_i, \phi) \tag{2.10}$$

In this equation ϕ is the set of parameters configuring the statistical model. $\zeta(x_i, \phi)$ is the probability density function of the distribution. Every item is generated from exactly one of these distributions $\zeta(x_i, \phi)$, so the joint density of x_i and $p_{i,j}$ can be described as

$$\prod_{j=1}^{k} (\gamma_j \zeta_j(x_i))^{p_{i,j}} \tag{2.11}$$

where γ_j represents the probability that item i originates from the model described by the density function ζ_j. k and n are the number of clusters and the number of items, respectively. $p_{i,j}$ are binary variables indicating whether item i was generated by ζ_j or not. In the soft clustering variant the variables $p_{i,j}$ obtain values from the range $[0, 1]$.

2.4.4 Density-based clustering

Density-based clustering is related to mixture-resolving clustering because both approaches are based on the assumption that items originate from different distributions. Opposed to mixture-resolving clustering, density-based clustering

does not start from any mixture distributions. The identification of the structure is obtained through the identification of regions of high density, the so-called modes.

An interesting aspect of this type of clustering algorithms, is the ability to identify clusters with arbitrary shapes. As opposed to several partitional or hierarchical clustering techniques, the similarity measurement gives no preference to certain shapes [Berkhin, 2002]. A main drawback of density-based clustering is often the lack of interpretability of clustering results.

Two main approaches can be identified in density-based clustering [Berkhin, 2002] :

1. Connectivity approaches
2. Density function approaches

The first group of techniques uses the notion of connectivity in a dense region. The connectivity depends on the definition of a ϵ-neighbourhood around an item. This neighborhood is defined as the space within a distance (based on a specified measure) ϵ around an item. Generally speaking, if the ϵ-neighborhoods of two items overlap, these two items are then directly connected. Two items p_1 and p_n are called density-reachable if there is a sequence of items p_i between p_1 and p_n with $1 < i < n$ where each p_i and p_{i+1} are directly density-reachable. In this manner, clusters can be formed of connected items. Items that are not connected to any other item, are considered to be outliers. Examples of this type of density-based clustering are DBSCAN (Section 2.5.1.4), GDBSCAN, OPTICS and DBCLASD [Mercer, 2003].

The second group of techniques connects an influence function with every item. This influence function is predefined, and superposing the influence functions of all the items creates an overall density function in the attribute space. An example of such an influence function for two dimensions x and y is shown in Equation 2.12.

$$f(x, y) = \exp^{-\frac{|x-y|^2}{2\sigma^2}} \tag{2.12}$$

The parameter σ defines the smoothness of the influence function.

Peak values of the superposed density function indicate the presence of clusters. The identification of these peak values is obtained in various ways. Two common methods are the application of thresholds or the use of hill-climbing techniques.

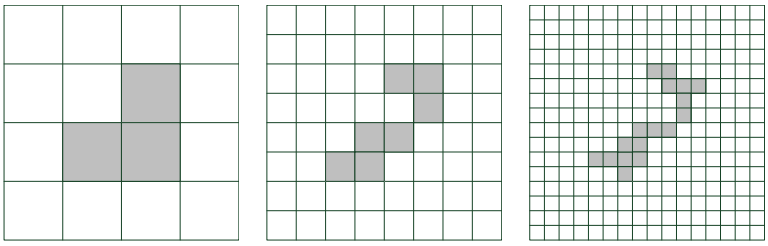
One well-known example of the density-function approach is the DENCLUE clustering algorithm, which is discussed in Section 2.5.1.5. Yager [Yager and Filev, 1992] proposed the Mountain Method, a simple and effective density method to estimate the number and locations of the clusters. A grid in the item space is constructed, and every grid point is replaced by a potential function, depending on the number of points in the vicinity. The grid point with the highest aggregated potential value is chosen as the first cluster center. Once the first cluster center is assigned, the potential of the remaining grid points

is reduced proportionally according to the distance to the cluster center. This process is repeated until the highest remaining potential value is below a predefined threshold value. This simple method is however less effective when dimensionality of the item space increases. Subtractive clustering is a variant of the Mountain Method. Instead of considering a grid point as a potential cluster center, a data point is taken as cluster center. In this manner, not all grid points need to be evaluated, so the problem is independent of the dimensionality of the item space.

2.4.5 Grid-based clustering

Density-based clustering algorithms face a performance issue when the dimensionality of the collection is high. To alleviate this issue, grid-based clustering imposes a grid data structure on the spatial data. This grid divides the space into a finite number of cells, which are used during the course of the clustering process.

The usage of a grid in this type of clustering is different from its application in some of the density-based clustering techniques such as DENCLUE: density-based clustering can use a grid to efficiently compute the aggregation of the influence functions in each data point. Grid-based clustering reverses this approach: the grid is used as a summary of the items included in a cell of the grid. An example of such a grid is shown in Figure 2.11. This grid is a simplified result of the well-known STING (Statistical Information Grid-based) algorithm [Wang et al., 1997], which divides the item space in several rectangular cells, and repeats this on different levels of detail. The grey cells indicate a high density of items. The STING algorithm is further discussed in Section 2.5.1.6.



(a) First layer of grid (b) Second layer of grid (c) Third layer of grid

Figure 2.11: Representation of a multi-resolution grid hierarchy for STING grid-based clustering, composed of three layers

Apart from the STING algorithm, other popular examples of this clustering type include CLIQUE (Clustering In QUEst) [Agrawal et al., 1998] and Wavecluster [Sheikholeslami et al., 2000]. Wavecluster first creates a multi-resolution grid on which a wavelet transformation is applied to identify dense

regions in the new space.

The main advantage of grid-based clustering is the insensitivity to the number of data points. The number of cells forming the grid determines the complexity of the algorithm. Therefore a grid-based technique is performing better for large datasets compared to density-based approaches. The inherent summarization of the information, however, causes a drop in effectiveness when the dimensionality becomes high [Han et al., 2001].

2.5 Clustering in Large Datasets

The advent of computer networks made it clear that the developed clustering approaches were not designed or suited to be applied on large and high-dimensional datasets, and therefore their performance degraded significantly when the size and dimensionality of the collection increased. In the following sections the first approaches to tackle this issue of enlarging datasets are summarized.

2.5.1 Traditional algorithms

2.5.1.1 CLARA

One of the first suggested approaches to overcome the rising problems of large datasets, is the CLARA (Clustering LARge Applications) algorithm [Kaufman and Rousseeuw, 1990]. This algorithm extends the previously mentioned K-medoids method by clustering only a sample or subset of the large collection. This two-phase clustering approach is in fact a combination of a sampling criterium and one of the most common implementations of the K-medoids clustering approach, the Partitioning Around Medoids (PAM) algorithm [Kaufman and Rousseeuw, 1990]. After the initial clustering of the sample, the remaining items are assigned to these obtained clusters. The objective function is a function of the distance from every item to its nearest medoid. These phases are repeated in order to find a good approximation of the global minimum of this objective function. As suggested in [Mercer, 2003] this clustering approach is only useful when the sample is a relevant representative of the complete collection. If n indicates the total size of the collection and n_s is the size of the sample, the validity of this approach is obviously in doubt when $n \gg n_s$. The construction of this subset requires either considerable knowledge about the content of the collection, or this approach must be performed iteratively to obtain stable clusterings. This sampling approach is applicable to all clustering algorithms, hence it is a more general clustering approach.

In [Ng and Han, 1994] the poor performance of this CLARA approach in the context of large datasets is discussed. One iteration of the PAM algorithm has computational complexity of $O(k(n-k)^2)$ with n the number of items and k

the number of clusters. This complexity makes PAM not efficient for very large datasets. PAM is applied on each sample CLARA draws from the collection. The complexity of each iteration becomes $O(kn_s^2 + k(n-k))$ with n_s the size of the sample [Zhang and Couloigner, 2005].

2.5.1.2 CLARANS

An improvement to the CLARA algorithm is called CLARANS (Clustering Large Applications based on Randomized Search) [Ng and Han, 1994]. This algorithm tries to combine the best features of the PAM and CLARA approach. PAM is able to find the global minimum of the objective function related to the collection. The first phase of the CLARA algorithm has a computational complexity of $O(n)$ because the algorithm examines randomly obtained samples. The difference in approach compared to CLARA is the concept of neighborhood. CLARANS uses this concept of neighbourhood to search for a random sample in the neighbourhood of a temporary solution. If an improved solution can be found in the neighbourhood, the obtained solution will be withheld as a candidate solution. Two parameters are needed in the application of the algorithm, i.e. $n_{neighbours}$ the maximum number of neighbours to visit of a specific solution, and n_{temp}, the maximum number of temporary solutions. The experiments performed to compare CLARA and CLARANS indicated that CLARANS is able to find cluster results of better quality than CLARA [Ng and Han, 1994]. CLARA however can obtain clustering results in less computational time than CLARANS [Ng and Han, 1994], which still has a computational complexity of $O(n^2)$. In this equation, n represents the number of items.

2.5.1.3 BIRCH

Balanced Iterative Reducing and Clustering using Hierarchies or BIRCH [Zhang et al., 1996, Zhang et al., 1997] is a hierarchical approach that uses a so-called Clustering Feature tree (CF-tree) to summarize the items of the data collection into nodes, each representing small subclusters.

A node i in the CF-tree contains the following parameters

$$CF_i = (o_i, ls_i, ss_i) \tag{2.13}$$

with o_i the number of items in subcluster i. ls_i and ss_i are respectively the linear sum and the square sum of the o_i items.

The hierarchical structure of the CF-nodes is a height-balanced tree that serves as the input for a hierarchical clustering process. An example of such a CF-tree is shown in Figure 2.12. The CF-tree contains two types of nodes: leaf and non-leaf nodes. Each non-leaf nodes contains at most $n_{children}$ child nodes, while a leaf node contains at most n_{items} items. Each leaf node has two pointers, which are used to chain all leaf-nodes together, to optimize scans in the tree. All entries in

a leaf node must satisfy a threshold parameter s, which describes the maximum distance an item may have with the centroid of the leaf node.

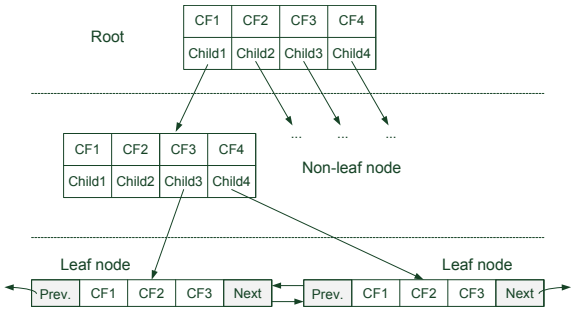


Figure 2.12: Example of a CF-tree with a maximum number of child nodes in leaf and non-leaf of 3 and 4, respectively. [Zhang et al., 1996]

The algorithm employs four different stages during each clustering process:

1. A linear scan on all items is performed, and the items are one by one inserted in a CF-tree. This insertion process is discussed in the next paragraph.

2. The CF-tree is summarized, by removing outliers and the optimization of clusters, to a desirable (memory) size depending on the properties of the clustering algorithm employed in phase 3.

3. A clustering algorithm (i.e. hierarchical clustering) is employed using the leaves of the CF-tree as input, resulting in effective distance calculations (Computational complexity of $O(n^2)$). The cluster-features are clustered into k groups.

4. An optional refinement phase is performed on the output of step 3 by reallocating items to the nearest clusters.

The insertion procedure can be summarized as follows: starting at the root node, the item recursively descends the CF-tree and selects the closest child node based on a distance metric. When an item enters a leaf node and the predefined tree parameters are violated, a split operation is performed on this leaf node. When the insertion operation is concluded, the information in the CF-tree is updated along the path the item has followed. Once the complete CF-tree is constructed, an existing clustering algorithm can be applied in phase 3 to cluster the leaf nodes based on their CF values, e.g. hierarchical agglomerative clustering.

To construct the CF-tree, only a single scan of the item set is required. Additional scans can however improve the cluster quality. Because a tree structure is used, it is not required that the complete CF-tree is in the computer memory

during the clustering process. The algorithm is linearly scalable in terms of number of items while maintaining a good cluster quality [Han et al., 2001]. Due to the parameters of the CF-tree, the size of the cluster does not always correspond to the user's expectation of natural clusters. Finally, due to the existence of a maximal diameter, BIRCH is known only to perform well when the underlying structure exhibits a spherical structure [Han et al., 2001, Milenova and Campos, 2002].

The overall computation complexity is minimum $O(n^2)$, depending on the clustering algorithm employed in the third phase. Due to the summarization of the collection, the algorithm in practice only requires half the computational time, compared to the agglomerative hierarchical clustering algorithm [Han et al., 2001].

2.5.1.4 DBSCAN

This density-based clustering algorithm, Density-Based Spatial Clustering of Applications with Noise or DBSCAN [Ester et al., 1996] is based on density-reachability (see Section 2.4.4). The DBSCAN algorithm requires two input parameters: ϵ and n_{min}. The ϵ value defines the influence space or ϵ-neighborhood around an item. An item i is called a core object when at least n_{min} items are in the ϵ-neighborhood of item i.

The algorithm randomly selects items and identifies their ϵ-neighborhood. When one core object is identified, a new cluster is started based on this object. All the items in this neighborhood become members of this cluster.

Different situations are possible during the process of the algorithm, such as multiple core objects in one cluster or the merging of two clusters. If the requirements of a core object for an item is not reached and the item is not within the neighborhood of a core object, then this item is considered as an outlier. The clustering algorithm halts when no new assignments can be performed.

DBSCAN has several advantages such as the ability to identify noise, no input parameter for the number of clusters and the ability to identify arbitrarily shaped clusters. If the input data is available in an index structure (e.g. R^*-tree [Beckmann et al., 1990]), the computational complexity of this algorithm is $O(n \cdot \log(n))$. If not, the computational complexity becomes of quadratic nature due to the distance calculations. DBSCAN is frequently used with the Euclidean distance measure, which is a considerable disadvantage in high-dimensional spaces. Tests also indicated that the algorithm does not perform adequately on layered datasets, i.e. datasets exhibiting different levels of detail [Ghosh and Strehl, 2006].

2.5.1.5 DENCLUE

DENCLUE, or DENsity-based CLUstEring, is another density-based clustering algorithm applying a density distribution function on each item, as described in Section 2.4.4. In this algorithm, clusters are called density attractors, which are the local maxima of the overall aggregated density function. These attractors

are identified by means of a gradient function and a hill climbing algorithm. Items attracted to a density attractor and for which the sum of the influence function values is higher than a predefined value, are identified as a cluster. The computational complexity is reported as $O(n \cdot log(n))$ with n the number of items in the dataset [Berkhin, 2002].

2.5.1.6 STING

Introduced in Section 2.4.5, the Statistical INformation GRID [Wang et al., 1997], abbreviated to STING, is a grid-based clustering approach defining rectangular cells in the data space. A hierarchical structure is created in this clustering approach, because each cell at a higher level is subdivided in a number of cells formed at a lower level. The number of layers of the grid is parameter or density based.

This structure is shown in Figure 2.11. In this figure, a high level cell is composed of four lower level cells. Each cell at every level stores attribute-dependent and -independent information. The following five parameters are associated with each cell, and with each attribute:

- Mean of all values of this attribute
- Standard deviation of all values
- Minimum value of this attribute
- Maximum value of this attribute
- Type of distribution this cell follows (e.g. normal, exponential or no distribution)

The distribution of a cell is defined by the user or obtained by hypothesis tests [Wang et al., 1997, Han et al., 2001]. The attribute-independent parameter is the number of items in the cell. The clustering process is performed based on a predefined density level threshold. This parameter guides the topdown process to find density regions satisfying this threshold. At a certain layer, each cell is estimated to be relevant or irrelevant for the clustering result. If a cell is found to be irrelevant, it is omitted from the further process. Otherwise a cell is refined by entering the next lower level. This iterative process stops when the lowest level has been reached and no further refinement of a cell is possible.

The STING approach exhibits several interesting properties. The hierarchical structure enables incremental updating of the tree structure, and the parallel processing of (sub)trees. The algorithm only requires a single pass through the dataset ($O(n)$) with n the number of items in the dataset) to generate the hierarchical structure. The identification of the density regions is at worst ($O(c)$, where c is the number of cells at the lowest layer. Because the number of cells

is often substantially lower than the number of items, the processing time of the algorithm is considerably lower compared to other techniques [Han et al., 2001].

STING does not store relationships between the child cells and neighboring cells to construct a parent cell. This, combined with the rectangular shaped cells (only horizontal and vertical boundaries in the data space are possible) limits the identifiable shapes of the clusters to isothetic polygons (as indicated as greyed regions in Figure 2.11).

2.5.1.7 CURE

Clustering Using REpresentatives or CURE [Guha et al., 1998] is an agglomerative hierarchical clustering algorithm using multiple items for the cluster representation in order to enable the algorithm to identify non-uniform sized or shaped clusters. A fixed number of well scattered items are selected to represent each cluster, after a shrinking process towards their cluster centers by a specified fraction, the so-called shrinking factor ς ranging between 0 and 1. At every phase in the merging process of the hierarchical algorithm, the clusters with the closest pair of representatives are selected. Several tests indicated that this variant enables a better identification of irregular shaped clusters and is less sensitive to outliers [Guha et al., 1998, Han et al., 2001]. Two passes are used to enable the algorithm to handle large datasets. The first pass consists of selecting a random sample of the dataset, and to partition this sample. A first partial clustering step is performed on each partition. In the next pass, a second clustering step is performed on all partial clusters to obtain the desired clustering result. The partitioning and partial clustering is performed to sieve the possibly selected outliers in the random sample. The reported computational complexity of this algorithm is $O(n^2 \cdot log(n))$.

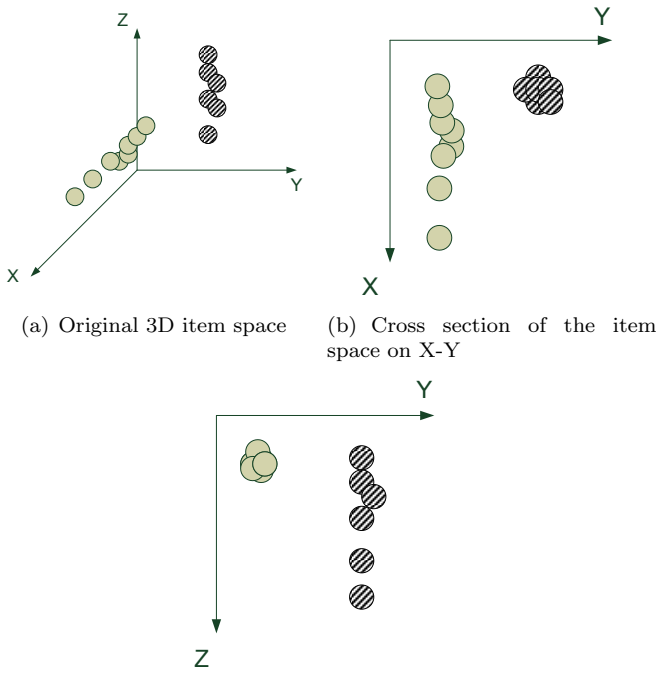
2.5.2 Advanced algorithms

As stated previously, the applicability of the traditional algorithms is often limited in a high dimensional feature space. These algorithms consider the full dimensionality of the input space, however many of these input dimensions are often irrelevant and can be considered as noise. As recently indicated by Kriegel [Kriegel et al., 2009], the main challenge for clustering in a high dimensional space is that different subsets of features are relevant for different clusters. This means that items group together in subspaces of the original space, but the subspaces of the clusters can vary [Parsons et al., 2004]. Besides the irrelevant features, correlations between sets of features influence the identification process of the underlying structure. These issues of relevance and correlation between features are called *local feature relevance* and *local feature correlation*.

These phenomena have a profound influence on feature extraction techniques described in Section 2.2.3.4. These techniques operate on the global data space of the collection, they transform the space to a lower dimensional approximation and

thus create a single subspace. This is in contrast with the mentioned phenomena, indicating the possible presence of a cluster in multiple subspaces. Therefore these global techniques can not be applied in (very) high dimensional environments [Kriegel et al., 2009].

These issues are graphically explained in Figure 2.13. Figure 2.13(a) displays a 3-dimensional item space containing two items, whereas Figures 2.13(b) and 2.13(c) are the cross sections or projections on the X-Y and Z-Y subspaces. These figures point out that each item is projected as a dense subset in one of these figures. Traditional feature selection or extraction does not perform well in this example because every dimension is only relevant for one group of items. Clustering in the original item space will not reveal this structure because each of the items is spread out along (only) one dimension [Aggarwal et al., 1999].



(a) Original 3D item space

(b) Cross section of the item space on X-Y

(c) Cross section of the item space on Y-Z

Figure 2.13: Representation of clusters in different subspaces

Instead of applying a global approach to the feature space, novel localized approaches are required taking local feature relevance and correlation into account in order to identify the relevant subspaces. Identifying these subspaces is however a challenging task, depending on the orientation of the subspaces. Almost all

algorithms described in literature are only able to identify *axis-parallel subspaces*. A few algorithms are capable of identifying the *arbitrarily or generalized oriented subspaces* [Patrikainen and Mannila, 2004, Kriegel et al., 2009]. The difference between these types of subspaces is shown in Figure 2.14. Cluster C1 exists in an axis-parallel subspace, while clusters C2 and C3 exist in two arbitrarily oriented subspaces.

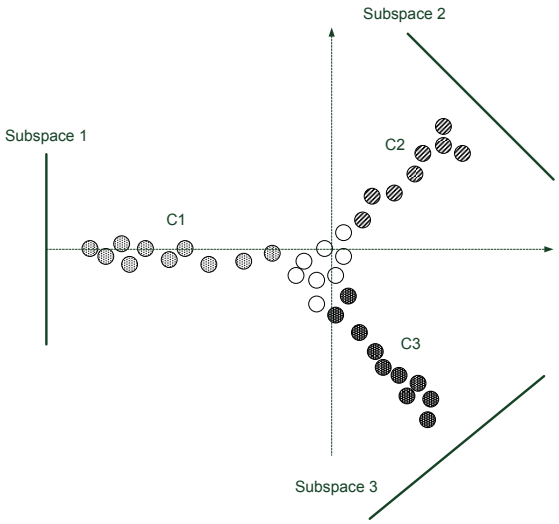


Figure 2.14: Illustration of the two types of orientation of subspaces in a two-dimensional space

Novel localized approaches are suggested to deal with this problem. Often these approaches include or are adapted versions of traditional techniques discussed in Section 2.4.

2.5.2.1 Axis-parallel subspace clustering

In literature, the following approaches for axis-parallel subspaces have been identified [Woo et al., 2004, Patrikainen and Mannila, 2004, Kriegel et al., 2009]:

- Projected clustering
- Subspace clustering
- Hybrid clustering

In the following paragraphs, these types of clustering approaches and algorithms are briefly described.

Projected clustering Aggarwal [Aggarwal et al., 1999] introduced the notion of a projected cluster for the algorithm PROCLUS (PROjected CLUStering). He defined a projected cluster i as a subset n_{SC_i} of items together with a subset m_{SC_i} of features in such manner that the items in SC_i are closely clustered in this subspace. Relating this definition to Figures 2.13(a) and 2.13(b) leads to the clustering of the striped items in a subspace spanned by features X and Y. These algorithms try to identify a unique assignment for each item to subspace clusters.

The PROCLUS algorithm consists of three phases. The first or initialization phase uses CLARANS (Section 2.5.1.2) to identify a set of possible medoids based on a sample set, proportional to the number of required clusters k. The refinement phase iteratively improves the quality of the clustering by replacing the medoids. The quality is measured by minimizing the standard deviation of the distance between items in the so-called *locality* or neighborhood to each medoid. Once the best set of medoids is identified, one more pass is performed over the items to improve the quality of the clustering. In this last phase, noise is identified if the distance to the nearest medoid exceeds a predefined threshold. Because of the k-medoid identification step in the first phase, the clustering is not stable: multiple runs of the algorithm result in different solutions. Furthermore the algorithm exhibits a tendency to identify equal sized spherical clusters [Kriegel et al., 2009].

A variant of the PROCLUS algorithm is FINDIT, Fast and INtelligent subspace clustering algorithm using DImension voting [Woo et al., 2004]. FINDIT uses a different similarity measure, *Dimension-oriented distance*, which expresses the similarity between two items as the number of dimensions in which the difference in value for both items is smaller than a given ϵ. The motivation is that in high dimensions it is more meaningful to be close to each other in many dimensions than to be very close to each other in only a few dimensions [Woo et al., 2004]. Another adaptation is the inclusion of a heuristic to measure correlation based on neighborhood information, because the obtained similarity between two items is dependent on the selected dimensions. Therefore it is important to identify the correlated dimensions. The FINDIT algorithm is reported to improve cluster quality and a better identification of outliers.

Subspace clustering Subspace clustering algorithms aim at finding all clusters in all subspaces of the original feature space [Kriegel et al., 2009]. The first clustering algorithm using the notion of subspaces, CLIQUE or CLustering In QUEst [Agrawal et al., 1998], is an integrated approach combining density-based and grid-based clustering. The complete item space is partitioned into non-overlapping rectangular cells. A pass over the grid is performed to identify the dense cells, i.e. cells containing more than τ items. Starting from the one-dimensional cells, $(l + 1)$-dimensional dense cells are identified from a set of l-dimensional dense cells. This process is based on the assumption that if a k-dimensional region is dense, the $(l - 1)$-dimensional region containing it should

also be dense. A cluster is defined as a maximal set of adjacent dense cells. To identify which subspaces (and the corresponding dense cells) are interesting, a heuristic is applied to avoid the processing of all cells, which increases exponentially in function of the dimensionality. This heuristic increases the efficiency of the algorithm, but a possible loss of true clusters is reported [Kriegel et al., 2009].

In literature several variants of the CLIQUE algorithm exist. The ENCLUS algorithm [Cheng et al., 1999], abbreviation for Entropy-based subspace CLUStering, also divides the space in fixed, non-overlapping, rectangular cells, but it uses the density and entropy as criteria to evaluate each subspace. If the entropy is below a predefined threshold value, the cell is withheld. nCluster [Liu et al., 2007a] releases the regular grid and incorporates an overlap window with size δ for one-dimensional cells. The MAFIA (MAximal Frequent Itemset) algorithm [Gan and Wu, 2004] extends CLIQUE by using an adaptive grid based on the distribution of the items to improve the cluster quality and the (time) efficiency of the algorithm. The CLIQUE and variant approaches are effective at finding arbitrarily shaped clusters, but the performance in high dimensional environments is however less promising due to scalability issues of a grid in function of the dimensionality.

An interesting extended algorithm based on DBSCAN is SUBCLU [Kailing et al., 2004]. Similar to the forming of the clusters in the CLIQUE method, the density identification in a subspace is based on the DBSCAN cluster model. SUBCLU thus identifies all clusters that DBSCAN would identify if it was applied to each subspace. Kriegel [Kriegel et al., 2009] reports that SUBCLU obtains a higher cluster performance than the grid-based approaches, but requires a higher runtime.

Hybrid clustering The category of hybrid clustering encompasses those algorithms that do not fit in projected nor subspace clustering. No hard assignment is made for each item resulting in overlapping clusters [Kriegel et al., 2009], but also not all subspaces are identified. In general, hybrid clustering algorithms identify overlapping clusters, but do not have the complete identification of all clusters as an objective. When the relevant subspaces are obtained, other full-dimensional clustering techniques are often applied to obtain an improved cluster performance. A well-known example of this type of algorithms is the grid-based algorithm DOC [Procopiuc et al., 2002], which uses a Monte Carlo procedure to identify an optimal clustering with high probability. Other examples include MINECLUS [Yiu and Mamoulis, 2003], FIRES [Kriegel et al., 2005] and P3C [Moise et al., 2008].

2.5.2.2 Generalized subspace clustering

Identifying clusters in this configuration is even more challenging compared to the axis-parallel oriented subspaces, because in case of arbitrarily oriented subspaces

the number of subspaces is infinite. These subspaces occur as groups of items located near a common hyperplane of an arbitrary dimensionality in the original data space. These hyperplanes describe linear dependencies or correlation among several features, and therefore this clustering is often referred to as correlation clustering [Kriegel et al., 2009]. Correlation clustering thus groups items into subsets, called correlation clusters, such that the items in the same correlation clusters are all related to a common hyperplane. The objective of these clustering algorithms is to identify the relevant hyperplanes.

The first category of this type of clustering algorithms is biclustering or pattern based clustering [Hartigan, 1972, Wang et al., 2002, Yang et al., 2002]. The objective is to identify groups of items that exhibit a similar behavior in their features. The similarity measurement is thus performed by comparing feature patterns. To reach this object, biclustering or two-mode clustering, operates simultaneously on the rows and terms of the document-by-term matrix. In [Kriegel et al., 2009] different types of biclusters are discussed based on the relationship between the features [Achtert et al., 2007]:

- Constant biclusters
- Biclusters with constant values on rows or columns
- Biclusters with coherent values
- Biclusters with coherent evolutions

In literature many biclustering algorithms can be found, most of which are able to identify only one of the bicluster types. Examples of such algorithms are FLOC [Wang et al., 2002], CoCLUS [Cho et al., 2004], OPSM [Ben-Dor et al., 2002] and MaPle [Pei et al., 2003]. An in-depth overview is given in [Madeira and Oliveira, 2004, Kriegel et al., 2009].

Pattern-based clustering is a special type of correlation clustering, because it is limited to positive correlations between features. This is extensively discussed in [Bohm et al., 2004, Kriegel et al., 2009]. Comparing the relatively simple patterns, that are identifiable by bicluster modeling, with the complex negative and positive correlations appearing in large datasets, the suggestion is made by Kriegel [Kriegel et al., 2009] that these algorithms are not suitable to tackle the general problem statement of arbitrarily oriented subclusters.

As explained in Section 2.2.3.4, PCA is a well-known technique to identify arbitrarily oriented directions describing high variance in a dataset. A local application (i.e. a selected set of dimensions) is however required in order to identify the clusters in different subspaces. The difference among the several PCA-based approaches is based on the manner of applying PCA and the selection of items and subspaces.

The previously described ORCLUS algorithm maintains a K-means approach. After selecting the k seeds, all items are assigned to the seeds with a

distance measure based on the weak eigenvectors of each cluster (in order to obtain the highest density in the subspace). The algorithm however is encumbered by several restrictions to the possible dimensionalities of the subspaces [Patrikainen and Mannila, 2004]. Two parameters are required as input for the algorithm: the number k of clusters to be found and the approximate dimensionality l of the subspaces containing these clusters. 4C [Bohm et al., 2004] is a density-based technique using a ϵ-neighborhood approach to identify the density of an item. Using this neighborhood, an adapted eigenvalue matrix is used in the PCA to derive the distance between two items. Other examples that include a PCA approach are COPAC, ERiC and CURLER which are capable of identifying nonlinear correlation clusters. The HiCO approach of Achtert [Achtert et al., 2007] is a hierarchical density-based approach to identify a hierarchy of correlation clusters. None of the previously mentioned techniques are hierarchical, however several correlation clusters of low dimensionality can together form a larger correlation cluster of higher dimensionality creating a large collection of possible clusters.

Besides the PCA-based approach, a few other approaches have been suggested to identify arbitrarily oriented subspaces. These approaches are based on fractals, random sampling or Hough transformation. An extensive overview is given in [Kriegel et al., 2009].

2.6 Discussion

The research for clustering techniques is coined by the supportive role an algorithm should provide in the KMS aimed for in the research projects. A number of desired functionalities were based on the use of a clustering algorithm:

- the identification of missing knowledge in a company: the identification of available tacit knowledge is based on the assumption that documents related to different knowledge workers provide a good summary of this implicit knowledge [Vertommen et al., 2008]. This results in a need to process a large collection of documents, on which a clustering algorithm needs to be employed for the tacit knowledge structuring process.

- compatibility of companies or divisions: the compatibility is also expressed in common tacit knowledge, and is thus also the result of applying a clustering algorithm on the document collection.

- identification of metadata for documents: for the identification of the metadata of documents, a clustering process needs to be applied on the collection to group similar documents

- enhancement of search engine performance: by pre-clustering the documents,

the matching of a query to a document cluster can decrease the required process time for the retrieval of documents.

In the vast amount of available clustering algorithms, several requirements were matched in order to identify one or more cluster algorithms applicable in the context of knowledge management. The set of requirements were defined by the application domain:

- Computational complexity: a computational complexity $(n \cdot log(n))$ with n the number of items is the first important property. As the number of documents in a knowledge base is an important issue in the performance, the time complexity of the clustering algorithm deserves special attention.
- Incremental property: as the updating of a knowledge base is an important process for the clustering algorithm, the clustering algorithm should exhibit this property.
- Interpretability: as the clustering provides a supportive task for knowledge management, the clustering algorithm should deliver interpretable clustering results.
- Subspace clusters: the domain of document clustering often results in cluster structures hidden in subspaces. Identifying the clusters in these subspaces therefore is an important property.

During the literature study, no single algorithm could be identified exhibiting all the desired properties. Therefore the research was initiated to develop clustering techniques that are able to reach these objectives. In Chapter 3 this research track is explained.

2.7 Cluster Validity Measurement

Applying a cluster algorithm to a collection of items will return a clustering scheme for this collection, however the proposed scheme is obviously not always the optimal solution. Improper parameter selection or an algorithm not suited for the collection are two of the main reasons for suboptimal solutions. To assess a clustering result, several cluster validity measures exist. They all measure two important properties: compactness and separation of the result [Berry and Linoff, 2004]. For compactness, the members of every cluster must be as close as possible to each other. Separation indicates the distance between the clusters, that needs to be maximized. As shown in the previous sections a wide variety of different clustering algorithms exists with different (often implicit) measures of quality.

As stated in [Qian et al., 2004], a possible method to compare different distance or dissimilarity measures is to study their retrieval performance in terms of precision and recall in several test environments. Other approaches are described in literature to validate cluster performance, such as Entropy, Purity, Variation of Information and the Mirkin Metric [Aliguliyev, 2009]. Cluster validation is a severe challenge because many validation measures often provide inconsistent information about the clustering performance [Wu et al., 2009a].

In literature three main validity approaches can be identified:

- External criteria: the results of a clustering algorithm are compared to an a-priori known structure of the collection. This is especially valid in standardized datasets, where experts have agreed on the structure.
- Internal criteria: opposed to external criteria, the internal criteria rely on figures or properties of the clustering result itself.
- Relative criteria: these evaluation measures are based on a comparison of a clustering result to other clustering results, obtained through the repeated application of the same algorithm with different sets of parameters values, also called clustering schemes. The idea is to choose the most optimal clustering scheme from a set of possible schemes.

2.7.1 External measures

For some datasets used in the presented research, an a-priori known structure or so-called 'golden standard' was available to assess the developed techniques. In this section, a measure related to precision and recall is explained. The subsequent paragraphs describe measures related to entropy.

2.7.1.1 F-measure

A popular measure for evaluating a clustering result, for which a reference clustering is available, is the F-measure [Chen et al., 2004]. This measure for a cluster i is a weighted harmonic mean of precision and recall of cluster i, and is mathematically defined as

$$F_i = \frac{2 \cdot recall_i \cdot precision_i}{recall_i + precision_i} \tag{2.14}$$

To calculate the $precision_i$ and $recall_i$, each cluster must be assigned to a known topic label. This is obtained in the following manner:

- Find the largest group of documents on the same topic within a cluster
- Label this cluster as the cluster covering that topic

- Mark this cluster as labeled, and the topic label as assigned
- Repeat with the remaining clusters and topics that are left
- Check that every cluster is covering a different topic. If not, the cluster containing the smaller group of documents on the same topic is set to be relabeled to a different topic.
- Repeat until all clusters are covering a different topic.

So for every cluster, precision and recall can be calculated. Precision is defined as the fraction of the documents in the cluster that originate from the assigned topic. Mathematically the precision of cluster i is defined as

$$precision_i = \frac{|C_{i,j}|}{|C_i|} \tag{2.15}$$

where $|C_{i,j}|$ is the number of documents in cluster i related to a topic j. $|C_i|$ is the number of documents in cluster i. Recall is defined as the fraction of documents of the total collection, originating from the assigned topic, that are assigned to the cluster. Mathematically the recall of cluster i is defined as:

$$recall_i = \frac{|C_{i,j}|}{nt_j} \tag{2.16}$$

where $C_{i,j}$ is the number of documents in cluster i related to a topic j. nt_j is the number of documents in the total collection that are related to topic j.

An overall clustering result is evaluated with its average F-measure, defined as

$$\overline{F} = \frac{\sum_{i=1}^{k} |C_i| \cdot F_i}{n} \tag{2.17}$$

$|C_i|$ is the number of documents cluster i contains, while $F - measure_i$ represents the F-measure for cluster i. n is the total number of documents in the collection. k is the number of clusters.

Based on the capabilities of the F-measure, as a weighted combination of precision and recall, to capture the homogeneity and completeness of a cluster in one resulting figure, this common validation measure was chosen in the presented research tracks.

2.7.1.2 Cluster Entropy

The cluster entropy [Zhao and Karypis, 2002] measures the homogeneousness of a cluster i, and is defined as

$$\text{entropy}_i = -\sum_{j}^{K} \frac{|C_{i,j}|}{nt_j} \log \frac{|C_{i,j}|}{nt_j} \tag{2.18}$$

$|C_{i,j}|$ is the number of documents in cluster i related to a topic j. T is the number of topics present in the dataset. nt_j is the number of documents in the total collection that are related to topic j.

2.7.1.3 Purity

To calculate the purity of a cluster i [Zhao and Karypis, 2002], the cluster is assigned to the topic T which is most frequently represented in this cluster. The purity of a cluster i is defined as

$$\text{purity}_i = \frac{1}{|C_i|} max_t(|C_{i,topic}|) \tag{2.19}$$

$|C_i|$ is the number of documents cluster i contains. $|C_{i,topic}|$ is the number of documents in cluster i related to a specific topic, which is the most represented topic in this cluster. The overall purity of a clustering result is calculated as the weighted sum of the purities of all the clusters.

2.7.1.4 Class Entropy

Cluster purity and entropy only refer to the homogeneity of a cluster, they do not take the recall of each topic into account [Zhao and Karypis, 2002] . Therefore, the measure of class entropy for topic j was introduced and is defined as

$$\text{class entropy}_j = -\sum_{i}^{k} \frac{|C_{i,j}|}{|C_i|} \log \frac{|C_{i,j}|}{|C_i|} \tag{2.20}$$

$|C_{i,j}|$ is the number of documents in cluster i related to topic j. $|C_i|$ is the number of documents cluster i contains. k is the number of clusters.

The total class entropy of a clustering result is defined as

$$\text{total class entropy} = \sum_{j}^{K} \frac{nt_j}{n} \text{ entropy}_j \tag{2.21}$$

in which nt_j is the number of documents of topic j. T is the number of topics present in the dataset. n is the total number of documents in the collection.

2.7.1.5 Other popular measures

The previously described quality measures only rely on the distribution of single texts in the document collection. Other techniques exist that are based on the distribution of pairs of texts. Examples of other techniques are

- Rand Statistic [Rand, 1971]

- Jaccard Coefficient [Jaccard, 1901]
- Hubert's Γ statistic [Bezdek and Pal, 1998]
- Normalized Γ statistic [DeShon and Alexander, 1996]

For further information concerning these measures, the referenced literature can be consulted.

2.7.2 Internal measures

As previously introduced, internal measures verify the cluster scheme of a clustering algorithm by the properties of the result. They can be subdivided in the following categories [Nguyen and Rayward Smith, 2008]

- Intra-cluster: these measures assess the distances between the items of the same cluster (e.g. sum of all squares, maximum diameter or centroid radius)
- Inter-cluster: this type of measurement assesses the distance between the clusters (e.g. sum of distances between centroids and sum of averaged distances between clusters)
- Mixed: Intra- or inter-cluster measurements are often not enough to sufficiently describe the quality of a cluster scheme. A mixed approach therefore is a combination of inter- and intra-cluster measures which provides a more complete measure for the quality of a clustering.
  A popular mixed measure is the Silhouette measure, defined by Rousseeuw [Rousseeuw, 1987]. If for every item i, $d(i)$ represents the average dissimilarity of item i to the other members of its cluster (i.e. cosine dissimilarity), and $q(i)$ represents the lowest average dissimilarity of another cluster to i, then Silhouette is defined as

$$Silhouette = \frac{q(i) - d(i)}{max\{q(i), d(i)\}} \tag{2.22}$$

  This quality measure results in a value in the range between $[-1, 1]$. -1 indicates i is clustered to the wrong cluster, whereas 1 indicates the opposite. The average S value for all items indicates the compactness of the complete dataset.

An extensive overview of popular internal measures can be found in [Milligan, 1981].

2.7.3 Relative measures

The underlying motive for relative measures is to select the best clustering result of a set of possible solutions, in relationship to a pre-defined criterion. In general, as indicated by Haldiki [Halkidi et al., 2001], two cases exist when using these measures, distinguished based on the knowledge or absence of knowledge of the required number of clusters in the set of parameters. For each of these cases, a different scenario needs to be followed. The bottom line of this measurement type is that the algorithm is run multiple times, every time using a different parameter setting. If the number of clusters is not included in the parameter set, the most stable region is sought for varying parameter values. In the other case, the results of these runs are plotted ordered by increasing number of clusters. The optimal setting is identified as a 'significant knee' in the plot [Vertommen et al., 2008]. If no underlying structure is present, no knee will appear in this plot. An in-depth overview of these techniques is given in [Halkidi et al., 2001].

2.8 Statistical significance in clustering

In the previously mentioned algorithms and quality measures, no indication is provided discussing statistical significance testing in the domain of clustering. As stated by [Hill and Lewicki, 2006], cluster analysis is not a typical statistical test. For many collections where clusterings algorithms are applied on, no *a priori* hypotheses are available. A clustering algorithm is often used in exploratory phases of an analysis process, and therefore statistical significance testing is not appropriate in this domain [Green et al., 1989].

Chapter 3

Improving Clustering Techniques

The research presented in this chapter provides an overview of three related research tracks. The motivation of this research, as is explained in the previous chapter, can be briefly summarized as to discover methods or techniques to enable clustering on larger datasets.

The first section of this chapter presents the first research track, namely a tree-based incremental clustering algorithm. Besides the desired incremental property, the algorithm also fits an interesting computational and space complexity requirement. The novelty of this algorithm is the representation format of a cluster, which only contains terms that are sufficiently present in the documents assigned to that cluster. This feature filtering is performed on the vector representation of each cluster in order to accommodate for the possible different subspaces these clusters could be located in.

Section 3.2 introduces a Pairwise Adaptive Dissimilarity measure for document clustering, which dynamically selects the features for the distance calculation. For each distance calculation, a different selection of features is performed. Besides the dynamic selection of features, a dynamic assessment of the number of selected features is also introduced. This dynamic selection creates different subspaces wherein each distance calculation is performed.

The last research track discussed in this chapter is the construction of a weighting scheme based on the law of Zipf. This law explains a relationship between the frequency of a term and the informative importance of this term in the total term space. This relationship is grafted in the weighting scheme in order to assign a higher value to more informative words compared to less informative.

3.1 Incremental Tree-based Clustering Algorithm

3.1.1 Introduction

In a typical KMS, updates of the knowledge base, such as insertions or deletions of documents, are not immediately propagated when these actions occur. Often these updates are collected and processed in batch mode at a predefined time instance or period, such as each night or at the end of the week. This updating process must be efficient enough in order to provide the end-user with the most recent information. Due to the very large size of knowledge bases these days, the algorithms governing these processes ideally exhibit incremental properties in order to enable updates of the current cluster structure with the recently inserted or deleted items, instead of performing a complete reclustering activity of the updated document collection.

As is explained in literature, a tree-based clustering algorithm is a promising research track based on its computational and memory complexity [Jain et al., 1999]. Moreover due to the structure of tree-based algorithms the memory usage can be reduced. For the popular b-order B^+-tree with h levels [Comer, 1979], the following properties concerning complexity hold:

- Maximum storage number is $n = b^h$
- The space required to store the tree is $O(n)$
- Inserting a document requires $(O(\log_b(n))$ operations in the worst case scenario
- Retrieving a document requires $O(\log_b(n))$ operations in the worst case scenario
- Removing a document requires $O(\log_b(n))$ operations in the worst case scenario

Several research attempts on incremental clustering algorithms can be identified in literature, resulting in incremental variants of algorithms discussed in Chapter 2. Examples are incremental DBSCAN [Ester et al., 1998] or incremental spectral clustering [Ning et al., 2010]. A brief introduction on this type of algorithms is provided in [Berkhin, 2002].

3.1.2 Algorithm

The novel clustering algorithm introduced in this chapter is based on the algorithm of Wong [chiu Wong and chee Fu, 2000]. He proposed a clustering algorithm suited for the incremental handling of large collections of web pages. A document such as a webpage however often does not contain a large content part (less then 100 words) which results in a sparse DTM compared with average repositories.

The results of this algorithm however were promising, and thus an adaptation of this algorithm was considered to enable the handling of larger documents.

The original incremental Document Cluster (DC)-tree is formed as a hierarchy of instances of two types of nodes: DC-leaf and DC-non-leaf nodes. In the continuation of this dissertation, the names of these types are abbreviated to leaf and non-leaf nodes. The difference between these types of nodes lies in the number of documents a node contains: a leaf nodes only contains documents, while a non-leaf node contains a combination of document and other (leaf or non-leaf) nodes. In this manner, documents can thus reside in the entire tree.

Documents themselves can also be grouped in a structure, a so-called document basket: if documents are similar enough according to a similarity measure, they are grouped together in a basket. A basket is treated as one entity in the further construction and processing of the tree. Such a basket can finally contain an infinite number of documents.

The original tree presented by Wong is constructed based on the following parameters:

1. Branching factor $n_{branching}$ defines the number of child entries a leaf or non-leaf node at most can contain. A child entry is either a node or a document.

2. Representative feature vector of DC_i, which contains the average frequencies of the terms of the documents in the subtree DC_i related to node i.

3. Two similarity thresholds (s_1, s_2), guiding the insertion process of a document in a leaf or non-leaf node.

4. Minimum number of children $n_{children}$ in a node.

To accommodate larger documents in a higher dimensional environment, the following adaptations were included in the previous parameter set:

1. A third similarity threshold is included. The extension of the set of similarity measures enables the ability to optimize the behavior of a document during the insertion process in a non-leaf node. This is explained in more detail in the next section.

2. The representative feature vector of a subtree DC_i is adapted to only include the weights of the relevant terms in the vector. Terms are considered relevant if they appear in at least half of documents in the subtree DC_i, as will be explained in the next paragraphs.

In the DC-tree, each non-leaf node has the following structure: $(DC_i, Child_j)$ with $i, j \in 1, \ldots, B$. $Child_j$ is a pointer to a document basket or subtree. For a leaf node, the structure is changed into (DC_i, Doc_j). Doc_j represents a pointer to a document or a document basket j.

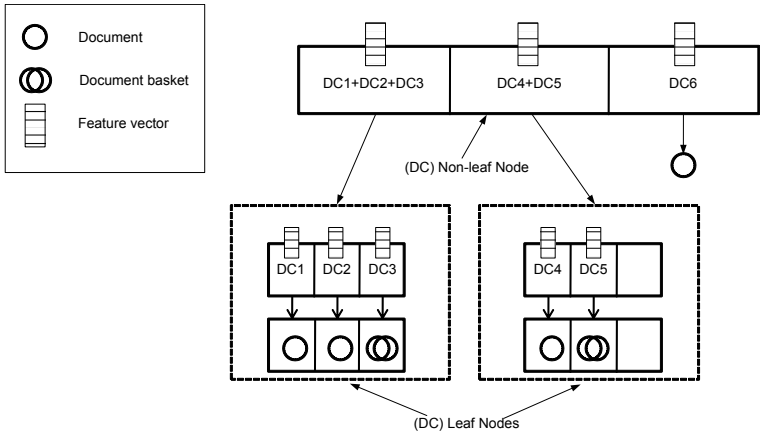


Figure 3.1: Example of an incremental clustering tree, containing 8 documents in one non-leaf node and two leaf nodes

An example of a DC-tree is given in Figure 3.1. In this example, B and M are set to 3 and 2, respectively. At depth 0 of the DC-tree, a non-leaf node is shown which contains two subtrees and a document. These subtrees, at depth 1, are leaf-nodes containing single documents or document baskets.

The feature vector of each node is obtained in the following manner: the vector contains all the average term weights of the documents residing in the subtree of this node, restricted in the following manner: if the term appears in more than half the number of documents in the subtree, the actual term weight is included in the vector. Otherwise, the term weight is set to zero. For example, if term t only appears in two of the five documents residing in a certain subtree, the related value for this term in the resulting feature vector will be set to zero.

The construction of a DC-tree is initiated based on a random sample of the initial document collection. The construction of this tree is governed by two processes, insertion and splitting. For adaptations in a latter stage, two additional processes are required: merging and deletion. In the following sections, an overview is provided of these processes.

3.1.2.1 Insertion and deletion procedure

The insertion process handles the addition of a document d to the DC-tree. The process starts at the root of the tree, wherein the document d recursively descends the tree by identifying the most similar child node according to a similarity measure. This type of tree does not force the item into a lower level when no child nodes are similar enough to this item.

The behavior of the document in the tree is determined by the three previously

introduced similarity thresholds. The difference in usage of these thresholds depends on the type of node the document tries to enter:

- To insert a document in a leaf node, the similarity threshold should at least result in a value higher than s_2.
- To insert a document in a non-leaf node, the compared threshold is s_3.
- The threshold s_1 is used when a document is compared with the feature vector of a document basket.

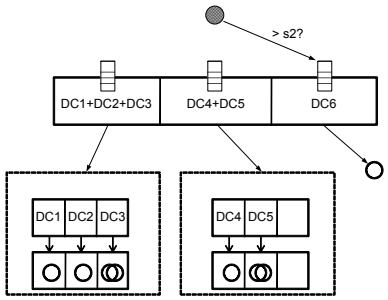
If the respective threshold is not met (depending on leaf or non-leaf node, i.e. s_2 or s_3), the document will be set as a child of the last successfully entered node.

The process is graphically explained in Figure 3.2, and algorithmic descriptions are provided in Appendix F.1. In Figure 3.2(a), the process starts with a document that will be inserted in the root node of a DC-tree. This DC-tree has a value of $_2$ for the $n_{children}$ parameter (i.e. minimum number of child nodes). This node is composed of two non-leaf nodes, *DC1+DC2+DC3* and *DC4+DC5*, and one document *DC6*. Each of these entries is represented by its feature vector.

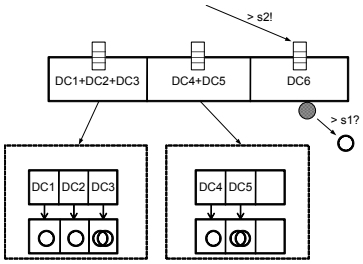
The Figures 3.2(b) and 3.2(c) indicate the usage of the two thresholds s_1 and s_2. In Figure 3.2(a) a document is matched to the document *DC6*. If this match is successful (as shown in Figure 3.2(b)), the document will be compared with the document to form a basket. This comparison is based on threshold $s1$. If the threshold is not reached (shown in Figure 3.2(c)) the new document is not combined with the other document. The new document will form a new leaf node as a child of node *DC6+DC7*. In this manner, the node *DC6* is converted to a leaf node *DC6+DC7*.

If the insertion of the document in a node containing one or more documents fails, the insertion process for non-leaf entries is initiated (shown in Figure 3.2(d)). The similarities with the child nodes is compared to threshold s_3. If no calculated similarity exceeds the threshold, the document will reside in the root node as new child node. Figure 3.2(e) indicates a successful insertion in the first child node. This child node is a leaf node, so thresholds s_2 and s_1 are used. If both s_2 and s_1 are met for the first child node, the document is inserted together with the existing document in document basket *DC1*, as indicated in Figure 3.2(f).

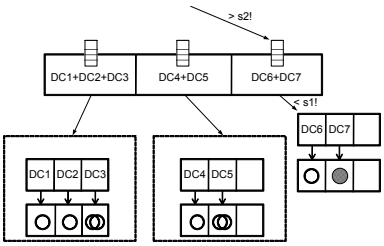
To reduce the sensitivity of the input order, a retreat and reinsertion step is added to the insertion procedure. If, during the insertion process, the new document in a non-leaf node is not able to reach a minimal similarity value (above a predefined percentage of s_3), the insertion process is halted and the document is put at the end of the insertion queue. A maximum number of possible reinsertions is set at the beginning of the clustering process before this document is considered as an outlier.



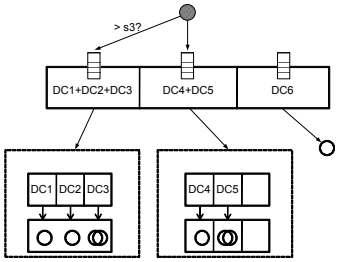
(a) Insertion at the root node of a DC-tree.



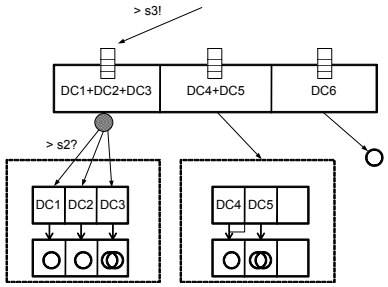
(b) After the successful insertion in the root node, a pairwise comparison with the leaf node is performed based on threshold $s1$.



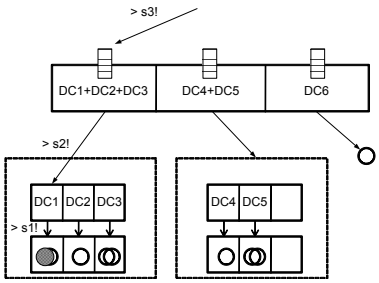
(c) The previous pairwise comparison failed, so the inserted document forms a new child node.



(d) Pairwise comparisons with the non-leaf nodes.



(e) After successful insertion in a non-leaf node, pairwise comparisons are performed on the child nodes.



(f) An example of the insertion of a document that results in a document basket.

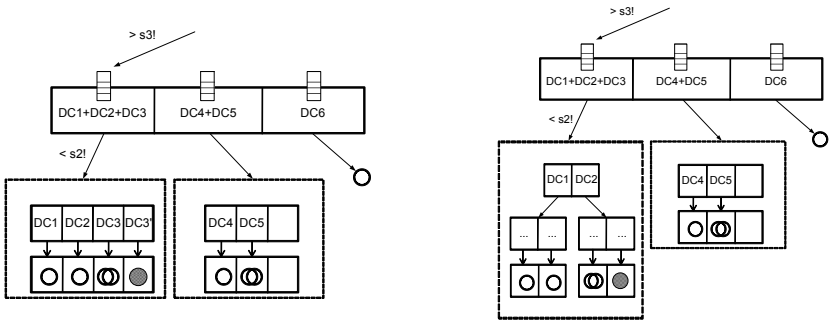
Figure 3.2: Insertion process of a new document in the DC-tree

In Appendix F.1, algorithmic descriptions are provided for these processes.

The deletion procedure of a child node in a node containing $n_{children} + 1$ child nodes, results in the removal of the child node and an updating process of the affected nodes towards the root of the tree. If however the resulting number of child nodes drops below $n_{children}$, a merging procedure is initiated combining the related parent node and its siblings. The procedure of this complete process is similar to the deletion procedure of the B^+-tree [Knuth, 1973, Comer, 1979].

3.1.2.2 Splitting and merging procedure

The insertion of a document in a node possibly results in a full node containing $n_{branching} + 1$ child nodes. In such event, the node will be split in two nodes which become child nodes of a newly created node. The splitting process is shown in Figure 3.3.



(a) The insertion process results in an oversized node

(b) The result of the splitting procedure applied on this node

Figure 3.3: The creation of new nodes as a result of the splitting procedure applied on the DC-tree

The splitting of a node into two nodes dividing the collection of child nodes is governed by the following objective function: maximizing the similarity between the child nodes belonging to the same parent node. Several approaches in literature exist to obtain this subdivision [Knuth, 1973, Comer, 1979]. In Appendix F.1, an algorithmic description of this process is provided.

The original setup was to calculate the similarity matrix containing all child nodes. The pair resulting in the lowest similarity value is selected as seed points for each of the groups of nodes forming the new child nodes. The other child nodes are assigned according to the similarities in the matrix and the selected seeds. To guarantee the minimum number of child nodes in the parent node, an additional reassignment phase can be performed.

The following two adaptations were performed on the splitting procedure to reduce the required computational time:

- the previously introduced feature vector representation allows a binary (optimal) comparison of the vectors (i.e. conversion to a binary representation based on the presence or absence of a feature in the original vector). The larger the overlap between the features of a pair of nodes, the larger the resulting similarity value between this pair of nodes.

- a bias towards large nodes as seeds is implemented in the procedure. This means that if the similarity between the largest nodes is below the threshold $s3$, the related pair is taken as seeds for the splitting procedure. If not, the next pair of smaller nodes is selected for this comparison process.

The inverse procedure, called merging, occurs when a document is deleted in a leaf node. This deletion operation can cause a cascade of merging operations towards the root of the tree. The merging procedure is similar to the procedure for the B^+-tree [Knuth, 1973, Comer, 1979].

3.1.2.3 Cluster Identification

For the cluster identification process, the following approach is adopted. A breadth-first search algorithm is applied starting from the root of the tree in order to find the relevant clusters. A relevant cluster is defined as a subtree containing at least a predefined number of items (n_{min}) in its subtrees. This threshold can be extended towards a lower and upper threshold value as suggested by Wong [chiu Wong and chee Fu, 2000]. When during the search process a relevant cluster in a node is found, the subtrees are not further investigated.

Each relevant cluster is represented by the feature vector of the related node. In our research, two formats for the representative feature vector are used: the first format is the full average vector containing all the features obtained after the preprocessing phase. The second format is the condensed vector where only the features are retained appearing in more than half of the documents of the cluster.

The usage of the number threshold enables the identification of clusters at various levels of details. The larger the n_{min} value, the broader and less condense the identified clusters become.

In a KMS this threshold offers several functionalities. The n_{min} closest documents can be suggested as interesting literature to a knowledge worker, based on his profile [Vertommen et al., 2008]. If n_{min} is increased, other documents can be identified less related to the profile of the user. Another application is the construction of a summary of the categories of a document collection, in which every category is described by the n_{min} most relevant documents of each category.

3.1.3 Validation

In this section, the datasets and the procedure is explained in order to validate the developed technique. In the experiments the proposed clustering technique is compared to the agglomerative hierarchical clustering algorithm. The used test cases have varying sizes and dimensionalities in order to observe the behavior of the proposed algorithm in these changing conditions.

3.1.3.1 Datasets

The test sets constructed for these experiments were derived from three standardized datasets:

- Ohsumed [Hersh et al., 1994]
- Reuters RCV1 corpus [Lewis et al., 2004]
- Banksearch [Sinka and Corne, 2005]

From these datasets, several test environments were constructed to fit the needs of this research, as described in Appendix A. Summarized, the following test environments were constructed for the validation process:

- Ohsumed 1: contains 29 test sets, each composed of 15 documents originating from 24 topics.
- Ohsumed 2: contains 20 test sets, each composed of 50 documents originating from 4 topics.
- Reuters 1: contains 19 test sets, each composed of 15 documents originating from 20 topics.
- Reuters 2: contains 20 test sets, each composed of 50 documents originating from 4 topics.
- Banksearch: contains 21 test sets composed of a varying number of documents originating from 11 topics.

3.1.3.2 Experimental setup

All test sets were treated identically before the clustering phase. This preprocessing phase contained the following steps:

- stopword removal using a list of 571 English stopwords
- stemming using the Porter stemming algorithm to reduce the number of terms [Porter, 1980]

- application of the TFIDF weighting scheme [Jain et al., 1999]

The reference clustering algorithm used to compare the performance of the incremental clustering algorithm, was the unsupervised hierarchical agglomerative clustering method using average linkage [Jain et al., 1999]. This algorithm was chosen based on its proven performance when applied on document collections [Zhao and Karypis, 2002, Wu et al., 2009b]. The number of clusters in the clustering result was set to the number of topics present in the test set.

For the incremental clustering algorithm, the following configuration parameters were set:

- The number of documents for the cluster identification process was set to half the known number of documents present in the reference structure.
- The applied similarity measure is the cosine distance measure.
- For the small test sets, Ohsumed 1 and Reuters 1, the (s_1, s_2, s_3)-values were set to $(0.8, 0.5, 0.3)$.
- For the other (larger) test sets, these values were set to $(0.6, 0.4, 0.25)$.
- The minimum and maximum number of children is set to 2 and 10, respectively.

A clarification for these parameter values is given in the following sections.

3.1.3.3 Validation measure

For each test set all documents were selected so as not to belong to more than one topic. Therefore it is assumed that each topic is represented in the clustering result by one cluster of documents. Each cluster of documents is labeled with a specific topic as follows:

- Find the largest group of documents on the same topic within a cluster.
- Label this cluster as the cluster covering that topic.
- Mark this cluster as labeled, and the topic label as assigned.
- Repeat with the remaining clusters and topics that are left.
- Check that every cluster is covering a different topic. If not, the cluster containing the smaller group of documents on the same topic is unlabeled.
- Repeat until all clusters are covering a different topic.

Each cluster is given a score by calculating its F-measure (see Section 2.7.1.1).

3.1.3.4 Results

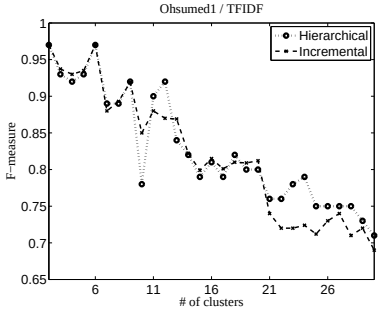
In this section, the results of comparing the proposed algorithm with the hierarchical clustering algorithm, applied on various test environments, are presented. Figure 3.4(a) presents the results for the first Ohsumed test set. For these experiments, the clustering results of 17 out of 29 datasets result in a higher scoring for the proposed clustering algorithm. The overall average improvement in F-measure is 1% (a F-measure value of 0.8283 compared to 0.8199). Remarkable is that the quality measure validates the scoring of the proposed algorithm significantly lower for the larger and higher dimensional datasets, indicated with a peak difference value of 8% (0.79 compared to 0.724 for test set 23).

The scoring difference for the second Ohsumed test set, presented in Figure 3.4(b), is similar to the first Ohsumed test set: the hierarchical algorithm obtains on average a better validation measure of almost 1% (an average F-measure of 0.4125 compared to 0.4055) over the complete test set. The overall scoring range for this Ohsumed is low, which constraints elaborate conclusions based on this test set.

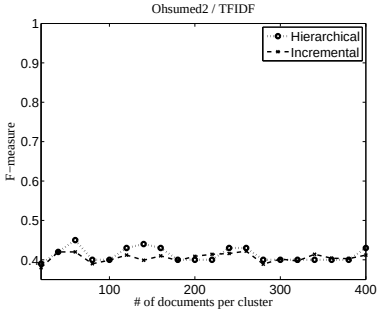
The conclusions for the two Ohsumed test sets remain valid for the two Reuters test sets. The results for the first Reuters test set (Figure 3.4(c)) are similar for the two clustering algorithms, although the scoring for the hierarchical clustering in overall is slightly higher (F-measure of 0.4550) compared to the proposed clustering algorithm (F-measure of 0.4472). The lower scoring for the proposed clustering algorithm at a higher number of documents, which was noticeable for the Ohsumed 1 test set, is also visible for the Reuters 2 test set (Figure 3.4(d)). Up to 2000 documents the proposed algorithm obtains better validation results, but for test sets with a higher dimensionality the scoring declines faster compared to hierarchical clustering. This can be seen in Figure 3.4(d) at the point where more than 500 documents are assigned to the clusters. For the last test set, Banksearch, the clustering of 6 datasets results in higher scores for the proposed algorithm, but also in this dataset the overall difference between the algorithms is rather small (F-measure of 0.5270 compared to 0.5063).

3.1.4 Parameter values

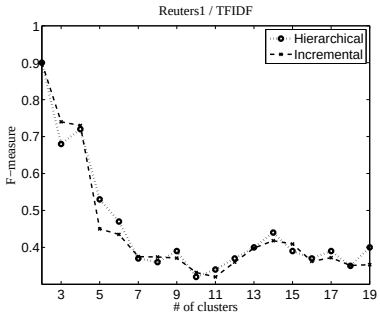
Research efforts were performed to identify an optimal set or range for the threshold values s_1, s_2 and s_3. Various parameter values were used in the clustering algorithm which was applied on the previously described test environments. These tests resulted in the conclusion that no optimal set of parameter values could be identified that was independent of the size and dimensionality of the dataset. This effect is shown in Figure 3.5, wherein the incremental clustering algorithm is applied, using two different sets of parameter values, on Reuters 1 and Ohsumed 2 test environments. Each parameter set is optimal for one of the test environments. As can be noticed in Figure 3.5, no single set of parameters results in overall



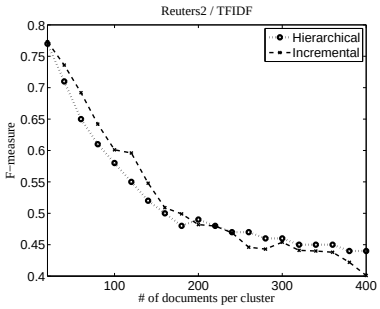
(a) F-measure results for the Ohsumed 1 test environment



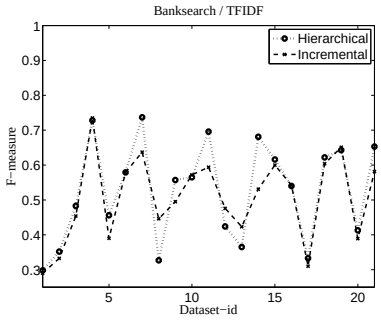
(b) F-measure results for the Ohsumed 2 test environment



(c) F-measure results for the Reuters 1 test environment



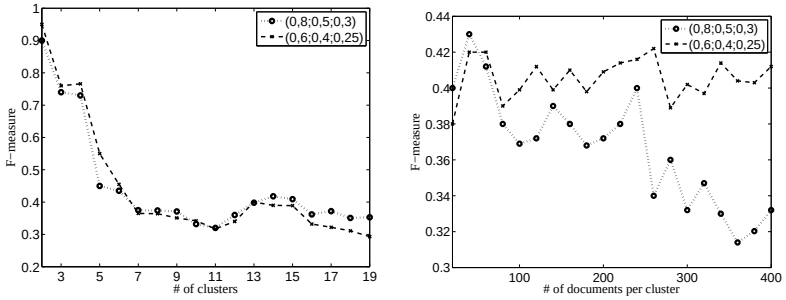
(d) F-measure results for the Reuters 2 test environment



(e) F-measure results for the Banksearch environment

Figure 3.4: Results of the hierarchical agglomerative clustering algorithm and the proposed incremental clustering algorithm on the five test environments

optimal clustering results. The first parameter set $(0.8, 0.5, 0.3)$ results in a better scoring for the Reuters 1 set (shown in Figure 3.5(a)), while the second parameter set results in a better scoring for the Ohsumed 2 set as shown in Figure 3.5(b)).



(a) F-measure results for the Reuters 1 test environment

(b) F-measure results for the Ohsumed 2 test environment

Figure 3.5: Results of a different set of parameter values on the Reuters1 and Ohsumed2 test environments

The sensitivity of these parameters was another point of attention. Such a sensitivity result is shown in Table 3.1, which presents an overview of the cluster quality using several parameter values on the Ohsumed2 test environment. In each column, a parameter varies while the other two thresholds remain constant.

The influence of threshold s_1, the threshold for a document basket, is indicated in the first column. A higher value for this parameter results in purer clusters which are less polluted by wrongly assigned documents. However, for larger and fuzzy datasets such as Ohsumed, a too strict value for this parameter results in a strong decline in cluster quality.

The second threshold, the threshold guiding the insertion in a leaf node, follows a similar reasoning but the peak value is reached in a lower value range. The course is also flatter than for the first threshold.

For the third threshold except for the extreme values the cluster quality remains equal over the complete range. This threshold, which guides the insertion in non-leaf nodes, thus obtains the best cluster quality for a lower value. A high value for this parameter also results in a high number of reinsertions, resulting in higher computing times. For the other test environments, similar conclusions concerning the sensitivity are drawn.

As no stringent conclusions can be drawn for the values of these parameters, further experiments on a larger scale are required to obtain more clear indications.

The optimal values for the minimum and maximum number of child nodes were set to 2 and 10 respectively, as these values were concluded to be optimal for this range of dataset size.

Table 3.1: Average F-measures when varying the threshold values for the incremental clustering algorithm on the Ohsumed2 test environment. The optimal parameter set for this test environment is (0.6;0.4;0.25). For each column, the indicated parameter differs while the other parameters have their optimal parameter value.

Value	s_1	s_2	s_3
0.2	0.2391	0.2190	0.3715
0.25	0.2447	0.3023	**0.4055**
0.3	0.2647	0.3197	0.4051
0.35	0.3131	0.3714	0.3989
0.4	0.3345	**0.4055**	0.4014
0.5	0.3996	0.4012	0.3743
0.6	**0.4055**	0.3981	0.3367
0.7	0.3836	0.4039	0.3235

3.1.5 Conclusions

From the results shown in the previous section, no overall improvement can be noticed when the incremental clustering algorithm is compared with the hierarchical agglomerative clustering algorithm for several test environments derived from standardized datasets. Intermediate test results indicated that the algorithm is not able to fully recover from an incorrect assignment in an early phase of the clustering process, which is more likely to occur in fuzzy datasets often present in a KMS. The contaminated feature vector subsequently influences the rest of the DC-tree and thus hampers the clustering process. To avoid such an incorrect assignment, a reinsertion of a document was made possible as explained previously. Several other variants were introduced to bypass this problem:

- Feature vectors: weighted; unweighted and binary vectors to represent the subtrees. The applied feature vector representation during the tests, which only considers the features appearing in more than half of the documents, obtained the highest cluster quality results.
- Thresholds: the original set of two thresholds was extended to three thresholds in order to distinguish between leaf and non-leaf nodes, and in the leaf nodes itself a differentiation can be made between similar and less similar documents. If documents are grouped in a document basket, they are treated in the rest of the clustering process as one document. Research was performed on the ideal number of reinsertions. The conclusion is that the improvement in cluster quality for a high number of reinsertions (e.g. 50 for 6000 documents) is negligible, besides the logical substantial influence on

the computational time. In the presented tests, this parameter was set at a value of 0.05 to 0.1% of the total number of documents in order to make this threshold less sensitive to the number of documents.

- Distance measures: besides cosine similarity, other measures were applied such as variance disturbance and Ward distances. Preliminary tests indicated that the application of the Pairwise Adaptive Dissimilarity measure, discussed in the next section, provided promising results. A further investigation is left as future work.

The results however also indicate that adaptations on the proposed algorithm are required in order to be suited for applications in larger datasets. The first indication is the sensitivity of the parameter values for the different test environments. No optimal value range for the different parameters could be identified that performed well for these test environments. These values thus depend on the type and dimensions of the datasets. As is observed from the results of the different test environments, the obtained cluster quality is also dependent on the size, and thus also dependent on the dimensionality. The main reason for this observed decline in cluster quality is attributed to the format of the feature vectors: as the dimensionality increases, these vectors insufficiently discriminate their documents from the complete document collection, as can be explained by the curse of high dimensionality.

In order to enable the processing of these datasets, other research efforts were initiated that manipulate the input during the preprocessing phase of a clustering process. These two research activities are the development of a new dissimilarity measure and a weighting scheme. These two research tracks will be discussed in the next sections.

3.2 Pairwise Adaptive Dissimilarity Measure

3.2.1 Introduction

One of the preferred approaches to bypass the curse of high dimensionality, as discussed in Section 2.2.3.1, is to construct a new dissimilarity measure. Due to the increasing dimensionality of the document collections, the application of a conventional proximity measure typically results in quasi-equidistant values. In the following sections a new dissimilarity measure to tackle this problem is introduced.

The following section discusses some basic principles, as well as related work on proximity measures. Section 3.2.3 describes in detail the proposed Pairwise Adaptive Dissimilarity measure. Sections 3.2.4 and 3.2.5 provide an experimental validation of the developed algorithm. Section 3.2.6 presents the results of this validation process. Finally, Section 3.2.7 offers conclusions based on the analysis of the performed validation tests.

3.2.2 Cosine dissimilarity measure variants

Due to its general acceptance as one of the popular dissimilarity measures, the cosine distance measure, as described in Section 2.2.2.3, is the basis of several variants. This short overview does not intend to cover all existing variants, but rather indicates the importance of the cosine distance measure to the field of clustering.

The *cosine distance measure with tolerance window* is an attempt to use the original distance measure with the application of one or more boundaries [Ramakrishnan, 2004]. This distance measure corrects the cosine distance to match peaks within a tolerance window, e.g. to account for errors between vectors originating from experimental and theoretical spectra in bioinformatics. If the tolerance window w is reduced to zero, the dissimilarity measure reduces to the original cosine measure.

The *adjusted cosine similarity*, originating from the domain of Collaborative Filtering, handles the difference in importance (or weighting) for each of the compared vectors [Sarwar et al., 2001]. When comparing user-based vectors (e.g. recommendation system for music), the original cosine dissimilarity measure has a serious drawback: the difference in rating scale between different users. The adjusted cosine similarity offsets this drawback by integrating a corresponding vector average in the similarity measure.

The *Binary cosine similarity measure* is defined as

$$s(d_1, d_2) = \frac{|d_1 \cap d_2|}{\|d_1\| \cdot \|d_2\|} \tag{3.1}$$

This similarity is computed in exactly the same way as the regular cosine similarity except for the fact that a term receives a score of 1 in the corresponding entry of the index matrix when it appears in a document and 0 otherwise.

The *Binary Dice* [Snijders et al., 1990], related to the *(extended) Jaccard coefficient* [Jain and Dubes, 1988], is defined as

$$s(d_1, d_2) = \frac{2 \cdot |d_1 \cap d_2|}{|d_1| + |d_2|} \tag{3.2}$$

with $|d_1 \cap d_2|$ the number of entries where the vectors d_1 and d_2 both had non-zeros, $|d_1|$ and $|d_2|$ the number of non-zero entries for vectors d_1 and d_2. This similarity measure equals the binary cosine value when the two compared document vectors contain the same number of non-zero entries. The binary cosine similarity penalizes less compared to the Dice coefficient when the number of non-zero vector entries is very different for the two documents. The interpretation of the resulting values is similar to the cosine similarity measure: a value of 0 means that the two documents are entirely dissimilar, while a value of 1 means the opposite. Compared to the original cosine dissimilarity measure, these variants are less suited for a document clustering environment [Ghosh and Strehl, 2006].

3.2.3 Proposed dissimilarity measure

In order to decrease the degree of noise in the feature set, a novel dissimilarity measure is proposed based on a combination of the original cosine dissimilarity measure and an observation specific feature selection. The measure is calculated similarly to the original dissimilarity measure, but for each pair of observations the inner product is obtained within a customized subspace. This dissimilarity measure is formally written as

$$d(d_{1,m_{SC}}, d_{2,m_{SC}}) = 1 - \frac{(d_{1,m_{SC}} \cdot d_{2,m_{SC}})_{1,m_{SC} \cup 2,m_{SC}}}{\|d_{1,m_{SC}}\|_2 \cdot \|d_{2,m_{SC}}\|_2} \tag{3.3}$$

$d_{1,m}$ is the vector containing the m most important features of the original vector d_1. The importance of a term in this context is reflected by its weight assigned by a scheme, as indicated in Section 2.2.2.2. Because a global weighting scheme reflects the importance of a term in the context of the document collection, this feature selection step at the dissimilarity calculation step is a manner to reduce the dimensionality to the document's most important terms.

The number of terms m_{SC} to use in the application of the dissimilarity measure can be determined in two manners. The first approach is the most straightforward: a fixed number of terms is used in the calculation procedure. This number is selected beforehand, and is based on the average term density per document vector of the term-by-document matrix. The subspace, constructed per pair of compared vectors, thus has limited dimensionality, in the range between m_{SC} and $2 \cdot m_{SC}$ defined by the union of the two reduced document vectors. This method has some drawbacks, such as a bias to similarity due to a possible strong difference in actual term density of the documents and the possible selection of terms in the unfavorable parts of the Zipf-Curve [Manning and Schtze, 1999]. The following small example explains these drawbacks. Suppose the following matrix D describing a collection of two documents and nine terms :

$$D = \begin{bmatrix} 4 & 4 & 4 & 1 & 1 & 1 & 4 & 1 & 4 \\ 4 & 3 & 0 & 0 & 1 & 1 & 0 & 0 & 4 \end{bmatrix}$$

The cosine dissimilarity measure applied to this matrix results in a value of 0.2346. Suppose the matrix $D1$ is constructed in the manner described above with m_{SC} equal to three. Because more than three terms have the same highest value for the first row, these three terms are randomly selected among the collection of highest values. In this example; the first, second and ninth are selected. In the second row, only three entries are non-zero. The following matrix $D1$ is thus created:

$$D1 = \begin{bmatrix} 4 & 4 & 4 \\ 4 & 3 & 4 \end{bmatrix}$$

The cosine dissimilarity measure applied to this matrix results in a value of 0.0082 which falsely indicates a high similarity between the two artificial

documents. If another value was chosen among the highest values of the first row, suppose the third entry, the following matrix $D2$ was created:

$$D2 = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 4 & 3 & 0 & 4 \end{bmatrix}$$

The cosine dissimilarity measure in this example still results in a value of 0.1410.

The second approach tries to overcome the drawbacks shown in the previous example. This approach is based on a dynamic measurement of the number of terms of the compared document vectors. This is

$$m_{SC} = p * min(m_1, m_2) \tag{3.4}$$

In this formula m_1 and m_2 are the numbers of terms that document vectors d_1 respectively d_2 contains. The parameter p indicates the percentage of terms taken into account for the dissimilarity measure. In this manner, for a given dataset, the same percentage of terms will be taken into account for every dissimilarity measurement. The m_{SC}-value is thus recalculated for every dissimilarity calculation between a pair of documents. By retaining only a percentage of non-zero terms for every vector, the two major drawbacks previously described can be overcome.

Consider the previous matrix D and, for reasons of clarity, a value of 1 is chosen for parameter p. Equation (3.4) results in a value for m_{SC} of 5. The following matrix $D3$ can be created.

$$D3 = \begin{bmatrix} 4 & 4 & 4 & 1 & 1 & 4 & 4 \\ 4 & 3 & 0 & 1 & 1 & 0 & 4 \end{bmatrix}$$

The cosine dissimilarity measure applied to this matrix results in a value of 0.2253, which is closely related to the measure obtained from the matrix D.

Two boundary conditions are applied to this dissimilarity calculation. If the density of one of the document vectors is below a predefined threshold, the resulting number of terms is assigned a predefined value. This threshold can be defined as e.g. a percentage of the average term density of the document collection. This threshold is introduced due to a similar problem as indicated in the previous example: if one document contains less terms than the average density of terms, then Formula 3.4 is biased towards similarity. An upper limit for the m_{SC}-value also can be identified. This upper limit guarantees the computational advantage of the new dissimilarity measure over the original cosine dissimilarity measure. This upper limit is further explained in Section 3.2.6.

This new variant on the cosine dissimilarity measure is not a distance measure according to the definition of a metric [Everitt et al., 2009]. This definition states that a dissimilarity measure $d(i, j)$ is termed as a distance measure if it complies with the triangle inequality $d(i, j) + d(j, m) \geq d(i, m)$ for the pairs of data points

(i, j), (i, m) and (j, m). The following term-by-document matrix D and the related dissimilarity matrix DM clearly shows that the metric inequality with $m_{SC} = 3$ is not necessarily valid when applying the new dissimilarity measure.

$$D = \begin{bmatrix} 1 & 2 & 4 \\ 2 & 2 & 3 \\ 0 & 0 & 2 \\ 0 & 1 & 1 \\ 3 & 3 & 0 \end{bmatrix} \tag{3.5}$$

$$DM = \begin{bmatrix} 0 & 0.0277 & 0.5037 \\ 0.0277 & 0 & 0.3695 \\ 0.5037 & 0.3695 & 0 \end{bmatrix} \tag{3.6}$$

Metric dissimilarity measures enable computational advantages such as lowering computational burden in neighborhood search due to the triangle inequality [Puzicha et al., 1999]. The newly proposed dissimilarity measure, like the original cosine measure, is therefore not suited to be applied in combination with metric based techniques.

3.2.4 Validation datasets

All datasets employed in these experiments were preprocessed in the same manner, were subjected to the same clustering method and two weighting schemes, and were evaluated using the same performance criteria. The only variations between the experiments were the dissimilarity measure used in calculating the clustering. The test sets constructed for these experiments were derived from three standardized datasets:

- Ohsumed [Hersh et al., 1994]
- Reuters RCV1 corpus [Lewis et al., 2004]
- Banksearch [Sinka and Corne, 2005]

From these datasets, the subsets described in Section 3.1.3.1 and Appendix A were used in the validation process.

3.2.5 Validation experimental setup

All previously described test sets were treated identically before the clustering step. This treatment contained following steps:

- stopword removal using a list of 571 English stopwords
- stemming using the Porter stemming algorithm to reduce the number of terms [Porter, 1980]

- application of the TFIDF weighting scheme or the BM25 weighting scheme [Jain et al., 1999]

The goal of these experiments was to evaluate the variation in clustering results for two different dissimilarity measures, i.e. the cosine and the Pairwise Adaptive Dissimilarity measure, and two different weighting schemes, i.e. TFIDF and BM25 schemes. The clustering algorithm used to compare the performance of the two distance measures in these experiments, was the unsupervised hierarchical agglomerative clustering method using average linkage [Jain et al., 1999]. The number of clusters in the clustering result was set to the number of topics present in the test set.

For each test set all documents were selected so as not to belong to more than one topic. Therefore it is assumed that each topic is represented in the clustering result by one cluster of documents. Based on this assumption, the F-measure validation measure is used to quantify the cluster quality in the clustering result (see Section 2.7.1.1). The method to determine the F-measure is identical to the method described in Section 3.1.3.3.

3.2.6 Results

The results of the clustering algorithm for each of the test sets are shown by plotting the average F-measure scores assigned to each clustering result. This clustering step is calculated for the cosine dissimilarity measure and the Pairwise Adaptive Dissimilarity measure. The Pairwise Adaptive Dissimilarity measure was calculated using the dynamic m_{SC}-definition. The required parameter p was set to the range of $0.35 - 0.4$ since these settings experimentally proved to provide the best performance for all used test environments.

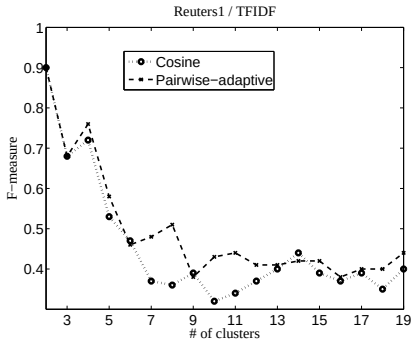
The related computational times required to construct the dissimilarity matrices are also given to indicate the performance improvement of the Pairwise Adaptive Dissimilarity measure compared to the cosine dissimilarity measure. The different test were executed on a dual core 2.4 GHz platform containing 4 GB of RAM.

In this section the cluster validation results of all test sets with the two weighting schemes are presented. In all figures, results corresponding to the cosine dissimilarity measure are indicated with a dotted line. The Pairwise Adaptive Dissimilarity measure results are indicated with a dashed line.

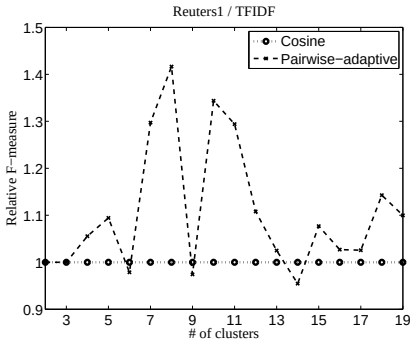
Clustering validation with TFIDF weighting scheme

The results of applying the previously described clustering method in the two test environments derived from the standardized Reuters RCV1 and Ohsumed document collection are shown in Figures 3.6 and 3.7. For 15 out of 18 datasets of the Reuters1 test environment, the proposed dissimilarity measure results in a

higher F-measure value. The obtained improvement ranges from -4.5% to 41% for the first Reuters test environment. Figure 3.6(a) presents these obtained F-measures. The average improvement for this test environment is 10.6% and is presented in Figure 3.6(b).

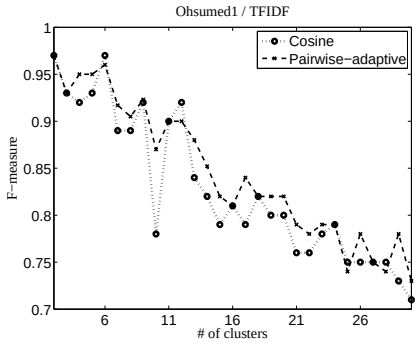


(a) F-measure results of the Reuters1 test set, using the TFIDF weighting scheme.

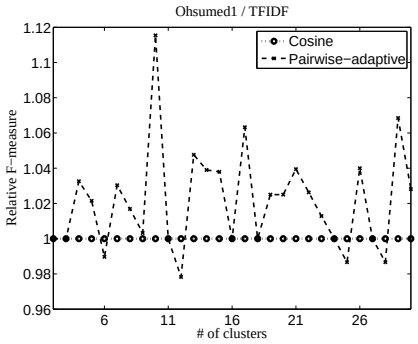


(b) Relative F-measure of the Reuters1 test set, using the TFIDF weighting scheme.

Figure 3.6: Results of the application of the Pairwise Adaptive Dissimilarity and cosine distance measure on the Reuters1 test environment, using the TFIDF weighting scheme



(a) F-measure results of the Ohsumed1 test set, using the TFIDF weighting scheme.



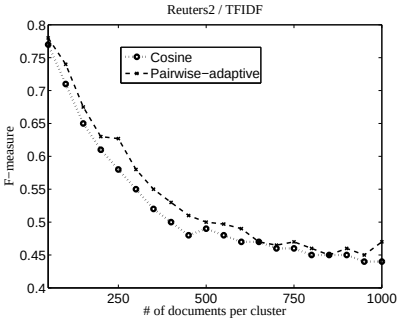
(b) Relative F-measure of the Ohsumed1 test set, using the TFIDF weighting scheme.

Figure 3.7: Results of the application of the Pairwise Adaptive Dissimilarity and cosine distance measure on the Ohsumed1 test environment, using the TFIDF weighting scheme

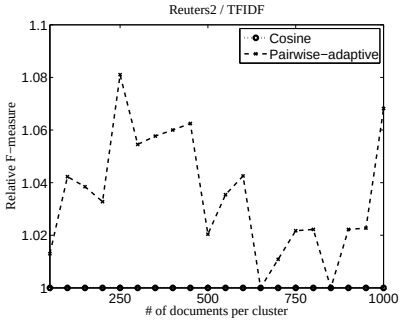
The results of the Ohsumed1 test environment, presented in Figures 3.7(a) and

3.7(b), show an average improvement of 2.6%. For this test environment 25 test sets out of 29 result in an improved F-measure. The improvement ranges from -2% to 11% compared to the original cosine dissimilarity measure.

The larger environments Reuters2 and Ohsumed2, presented in Figures 3.8 and 3.9, also result in an overall improvement. All sets of the Reuters2 test environment result in an equal or higher F-measure score for the new dissimilarity measure, as shown in Figure 3.8(a). Figure 3.8(b) indicates improvements up to 8%, the average improvement for this environment is 3.5%. For the Ohsumed2 test environment only two test sets out of 20 result in a slightly lower F-measure. The average improvement is 5.5%, ranging from -2.3 to 47.5%. These results are shown in Figures 3.9(a) and 3.9(b).



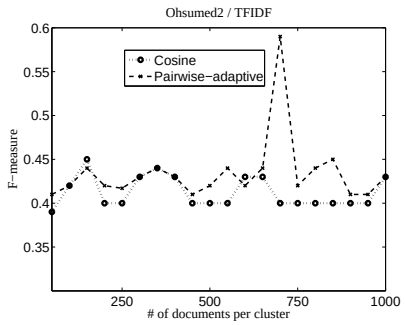
(a) F-measure results of the Reuters2 test set, using the TFIDF weighting scheme.



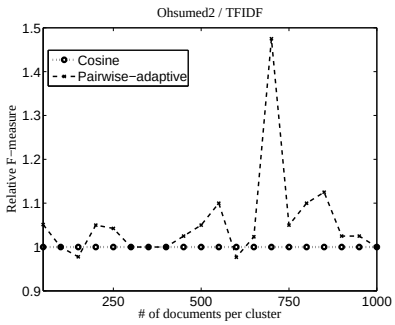
(b) Relative F-measure of the Reuters2 test set, using the TFIDF weighting scheme.

Figure 3.8: Results of the application of the Pairwise Adaptive Dissimilarity and cosine distance measure on the Reuters2 test environment, using the TFIDF weighting scheme

Finally the results of the Banksearch test environment are shown in Figure 3.10. The improvement for this test environment ranges up to 77 %. The overall average improvement for this test environment is 19.5%. These results are shown in Figures 3.10(a) and 3.10(b).

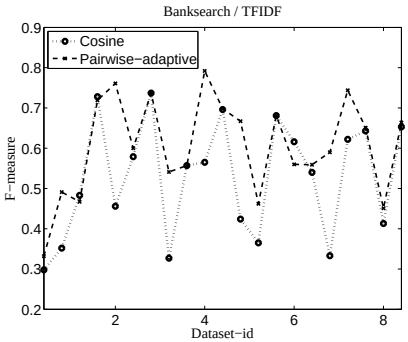


(a) F-measure results of the Ohsumed2 test set, using the TFIDF weighting scheme.

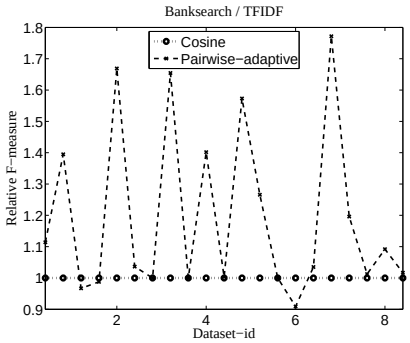


(b) Relative F-measure of the Ohsumed2 test set, using the TFIDF weighting scheme.

Figure 3.9: Results of the application of the Pairwise Adaptive Dissimilarity and cosine distance measure on the Ohsumed2 test environment, using the TFIDF weighting scheme



(a) F-measure results of the banksearch test set, using the TFIDF weighting scheme.



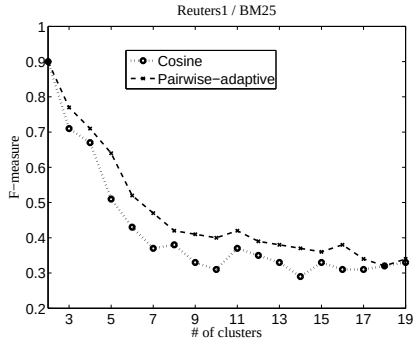
(b) Relative F-measure of the banksearch test set, using the TFIDF weighting scheme.

Figure 3.10: Results of the application of the Pairwise Adaptive Dissimilarity and cosine distance measure on the Banksearch test environment, using the TFIDF weighting scheme

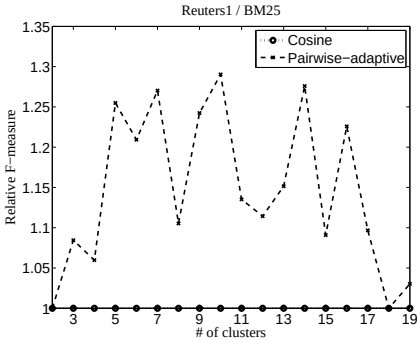
Clustering validation with BM25 weighting scheme

All the test sets described in Section 3.2.4 were also subjected to a second series of tests using the BM25 weighting scheme. In Figure 3.11, the absolute and average F-measure are shown per test set for the Reuters 1 test environment. All test sets of this environment resulted in a higher F-measure compared with the scoring of

the original cosine dissimilarity measure, as shown in Figure 3.11(a). The BM25 weighting scheme results in an average improvement in F-measure of 14.65% for this test environment (see Figure 3.11(b)).



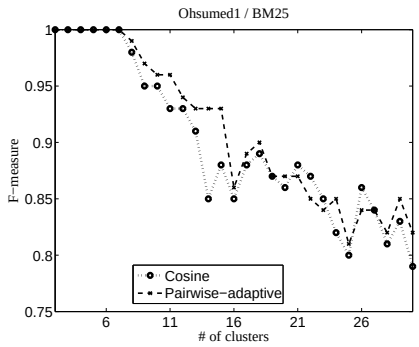
(a) F-measure results of the Reuters1 test set, using the BM25 weighting scheme.



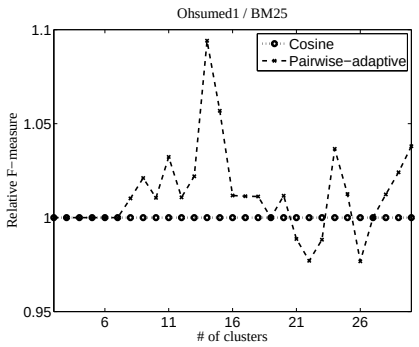
(b) Relative F-measure of the Reuters1 test set, using the BM25 weighting scheme.

Figure 3.11: Results of the application of the Pairwise Adaptive Dissimilarity and cosine distance measure on the Reuters1 test environment, using the BM25 weighting scheme

The average increase in performance for the first Ohsumed test environment, shown in Figure 3.12(b), is similar to the increase noted when the TFIDF weighting is used, namely 2.67%. The increase for this test environment ranges to 9.3% as shown in Figure 3.12(a).



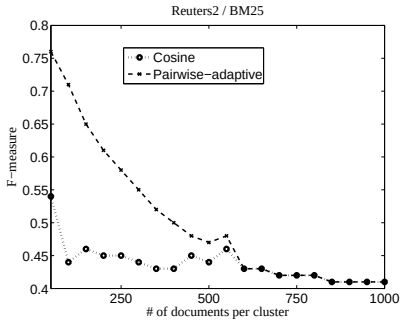
(a) F-measure results of the Ohsumed1 test set, using the BM25 weighting scheme.



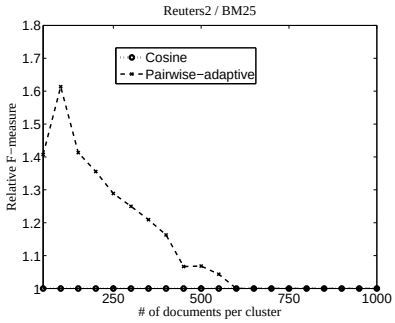
(b) Relative F-measure of the Ohsumed1 test set, using the BM25 weighting scheme.

Figure 3.12: Results of the application of the Pairwise Adaptive Dissimilarity and cosine distance measure on the Ohsumed1 test environment, using the BM25 weighting scheme

The results for the two other environments of Reuters and Ohsumed are shown in the next figures. Figure 3.13 displays the F-measures for the Reuters 2 test environments. The average increase for the F-measures becomes 5.5%. The relative values for this test environments are shown in Figure 3.13(b).



(a) F-measure results of the Reuters2 test set, using the BM25 weighting scheme.

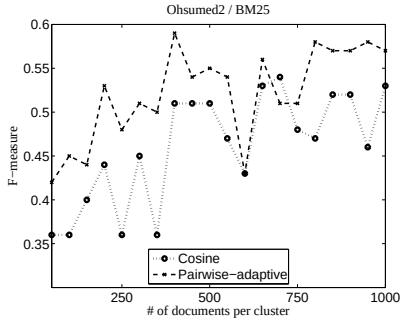


(b) Relative F-measure of the Reuters2 test set, using the BM25 weighting scheme.

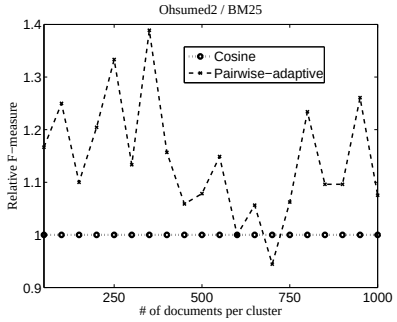
Figure 3.13: Results of the application of the Pairwise Adaptive Dissimilarity and cosine distance measure on the Reuters2 test environments, using the BM25 weighting scheme

The results for the Ohsumed 2 test environment is shown in Figure 3.14.

These F-measures score on average 14.39% higher when the proposed dissimilarity measure is employed. The relative values for this test environments are shown in Figure 3.14(b).

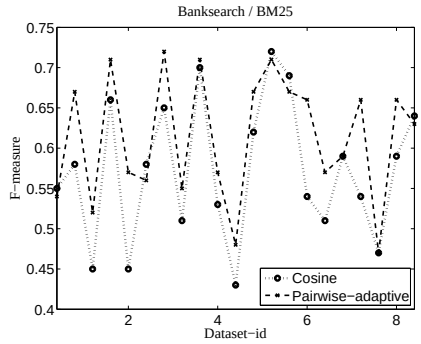


(a) F-measure results of the Ohsumed2 test set, using the BM25 weighting scheme.

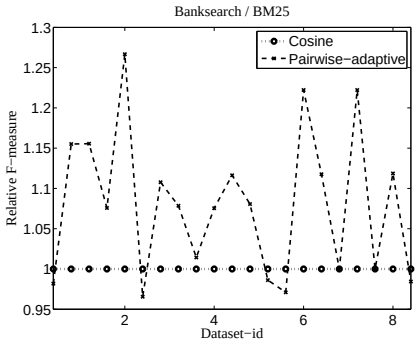


(b) Relative F-measure of the Ohsumed2 test set, using the BM25 weighting scheme.

Figure 3.14: Results of the application of the Pairwise Adaptive Dissimilarity and cosine distance measure on the Ohsumed2 test environment, using the BM25 weighting scheme



(a) F-measure results of the Banksearch test set, using the BM25 weighting scheme.



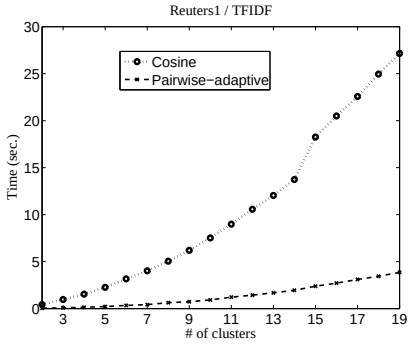
(b) Relative F-measure of the Banksearch test set, using the BM25 weighting scheme.

Figure 3.15: Results of the application of the Pairwise Adaptive Dissimilarity and cosine distance measure on the Banksearch test environment, using the BM25 weighting scheme

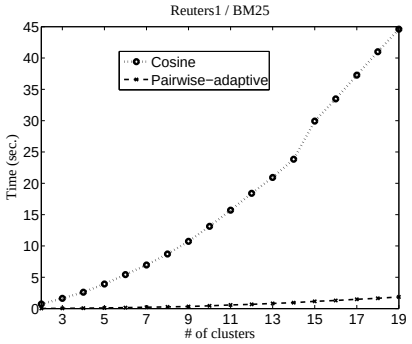
The results of the last environment, the Banksearch tests, are shown in Figure 3.15. For 17 test sets out of 20, the proposed dissimilarity measure results in a higher F-measure value with an average improvement of 7%. These results are shown in Figures 3.15(a) and 3.15(b).

Computational time

In this paragraph an overview is given of the computational time required for the construction of the dissimilarity matrices. The computational results are presented for each of the two weighting schemes. In all figures for the two weighting schemes, a clear difference can be seen between the cosine dissimilarity measure and the Pairwise Adaptive Dissimilarity measure. For the smaller test sets, Reuters1 and Ohsumed1, related Figures 3.16 and 3.17 show that the Pairwise Adaptive Dissimilarity measure decreases the necessary computational time approximately with a factor six to ten.



(a) Computational time results for constructing the dissimilarity matrix of the Reuters1 test set, weighted with the TFIDF weighting scheme

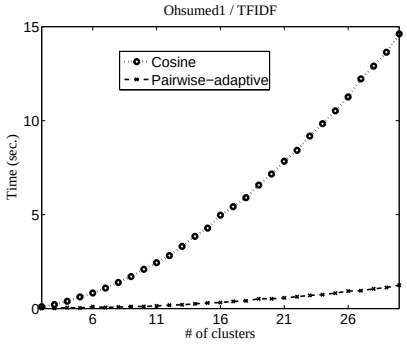


(b) Computational time results for constructing the dissimilarity matrix of the Reuters1 test set, weighted with the BM25 weighting scheme

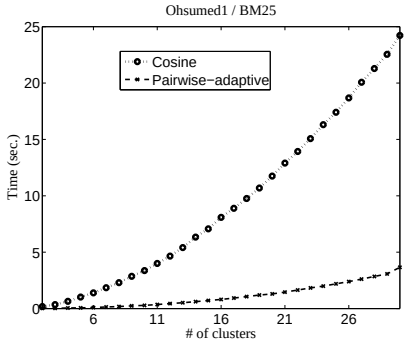
Figure 3.16: Computational time results for constructing the dissimilarity matrix of the Reuters1 test environment

For the larger test sets Reuters2 and Ohsumed with related Figures 3.18(a) to 3.19(b), this difference is still a factor of three to eight. For some test sets of the Banksearch collection, Figures 3.20(a) and 3.20(b), the difference in computational time is even larger.

Figures 3.16, 3.17, 3.18, 3.19 and 3.20 also indicate the quadratic nature of the construction of the dissimilarity matrix. To identify the origin of this lower computation time needed for the new dissimilarity measure, the time complexity is further investigated. This research results in a crude upper limit for the number of terms to be chosen. The computational complexity, expressed as the number of

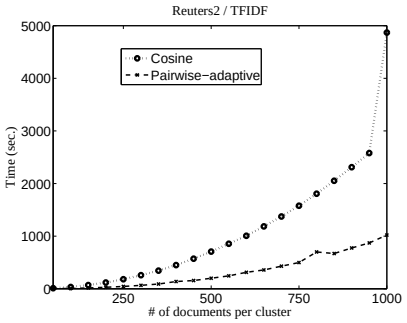


(a) Computational time results for constructing the dissimilarity matrix of the Ohsumed1 test set, weighted with the TFIDF weighting scheme

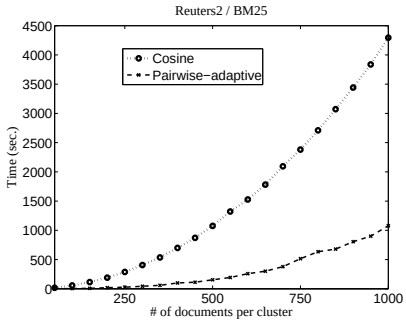


(b) Computational time results for constructing the dissimilarity matrix of the Ohsumed1 test set, weighted with the BM25 weighting scheme

Figure 3.17: Computational time results for constructing the dissimilarity matrix of the Ohsumed1 test environment



(a) Computational time results for constructing the dissimilarity matrix of the Reuters2 test set, weighted with the TFIDF weighting scheme



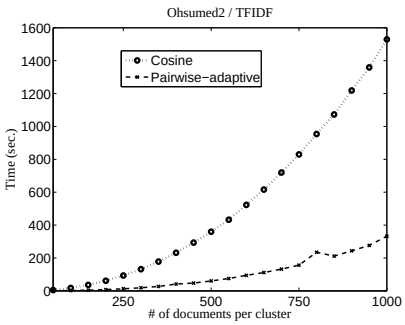
(b) Computational time results for constructing the dissimilarity matrix of the Reuters2 test set, weighted with the BM25 weighting scheme

Figure 3.18: Computational time results for constructing the dissimilarity matrix of the Reuters2 test environment

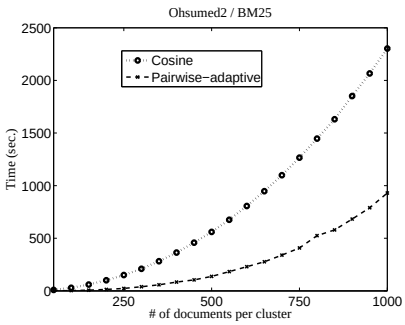
required operations, of the cosine dissimilarity measure between two documents is of a linear nature, formulated as

$$6 \cdot m + 1 \tag{3.7}$$

where m is the number of dimensions in the vector space model. The computational complexity of the construction of the dissimilarity matrix is

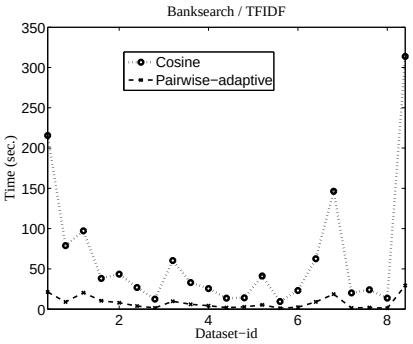


(a) Computational time results for constructing the dissimilarity matrix of the Ohsumed2 test set, weighted with the TFIDF weighting scheme

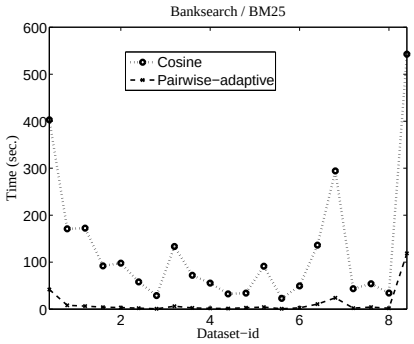


(b) Computational time results for constructing the dissimilarity matrix of the Ohsumed2 test set, weighted with the BM25 weighting scheme

Figure 3.19: Computational time results for constructing the dissimilarity matrix of the Ohsumed2 test environment



(a) Computational time results for constructing the dissimilarity matrix of the Banksearch test set, weighted with the TFIDF weighting scheme



(b) Computational time results for constructing the dissimilarity matrix of the Banksearch test set, weighted with the BM25 weighting scheme

Figure 3.20: Computational time results for constructing the dissimilarity matrix of the Banksearch test environment

therefore formulated as

$$(6 * m + 1) \cdot \frac{n \cdot (n-1)}{2} \tag{3.8}$$

or summarized as

$$O(n^2 \cdot m) \tag{3.9}$$

with n the number of documents in the repository.

The computation complexity of the Pairwise Adaptive Dissimilarity measure is derived as

$$(2 \cdot m \cdot log(m) + 14 \cdot m_{SC}) \tag{3.10}$$

where d is the number of dimensions in the vector space model. m_{SC} is the (average) number of terms taken into account in the dissimilarity calculation. The computational complexity of the construction of the distance matrix therefore is formulated as

$$(2 \cdot m \cdot log(m) + 14 \cdot m_{SC}) \cdot \frac{n \cdot (n-1)}{2} \tag{3.11}$$

The sorting step is executed once before the actual dissimilarity calculation, therefore the computational complexity can be reduced to

$$n \cdot m \cdot log(m) + \frac{n \cdot (n-1)}{2} \cdot 14 \cdot m_{SC} \tag{3.12}$$

or summarized as

$$O(n^2 \cdot m_{SC} + n \cdot m \cdot log(m)) \tag{3.13}$$

Comparing equations 3.8 to 3.12 indicates that the value for the number of terms m_{SC} is roughly limited to the intersection of these two complexity functions. The m_{SC}-value at this intersection is

$$m_{SC} = [6 \cdot m + 1 - \frac{2 \cdot m \cdot \log(m)}{n-1}] \cdot \frac{1}{14} \tag{3.14}$$

where m is the dimensionality of the document collection. n is the number of documents in the collection.

Singular Value Decomposition

The application of SVD on the original TFIDF-weighted matrix results in similar F-measure scores for all test sets examined in this research. The application of this feature extraction method however transforms the original sparse matrix into a reduced full matrix [Trefethen and Bau, 1997]. The resulting computational load of applying the Pairwise Adaptive Dissimilarity measure to these matrices is higher compared to the load of the cosine dissimilarity measure. In Equation 3.13 the value of K equals the dimensionality of the term space. The resulting complexity thus becomes $O(n^3)$. The computational load for the test set is shown in Table 3.2.

This new dissimilarity measure based on a prior feature selection step thus is not suitable for application in combination with any feature extraction method resulting in a full matrix.

# of terms	P. A. diss. (sec.)	Cosine (sec.)
150	1.6307	0.0080
300	10.9493	0.0212
450	33.7489	0.0454
600	76.6627	0.0823
750	146.4718	0.1267
900	247.1154	0.1764
1050	366.5702	0.2350
1200	541.7428	0.3019
1350	762.1712	0.3820
1500	1050.2000	0.4705

Table 3.2: Computational time of the proposed measure (abbreviated to P. A. diss.) and the cosine dissimilarity measure, when SVD is applied on the terms space

3.2.7 Conclusions

In this section the approach of constructing a new dissimilarity measure in order to bypass the curse of high dimensionality is presented. This dissimilarity measure is composed of a feature selection step prior to every dissimilarity measurement. For each pairwise comparison between two documents, a relevant subset of terms is selected that will contribute to the measured similarity between both related vectors. Two approaches for selecting the number of terms based on the document-by-term density are explained. The first approach applies a feature selection step with a fixed number of terms for every pairwise comparison. The other approach dynamically selects a number of features for every document based on a predefined ratio. An initial guidance for this parameter is presented based on the conducted tests. This parameter setting can be further improved based on other cluster quality validation measures. Boundary conditions on the selection of the number of terms are also derived, taking the computational complexity into account.

The new dissimilarity technique results in an improved overall clustering performance compared with the original cosine dissimilarity measure, illustrated by means of the F-measure function in five different test environments. These test sets vary in the number of topics or the number of documents. The results were also obtained with a lower required computational time, thus creating a considerable speed gain.

Algorithmic descriptions of this dissimilarity measure are provided in Appendix C.2.

3.3 A Zipf-based Weighting Scheme

As indicated in Section 2.2.2.1, weighting schemes are used in VSM to compensate the absence of term correlations. A vast range of weighting schemes exists, as described in Section 2.2.2.2, and several of them have obtained a certain popularity among researchers and in industry [Jain et al., 1999]. In this section, a weighting scheme is introduced which aims to incorporate the informational value of a word, based on an empirical linguistic law. In the first section, a short overview is provided of several linguistic laws resulting from the concept of Large Number of Rare Events (LNRE). The successive sections provide an overview of one of the best-known linguistic law, the Zipf law, followed by the motivation for developing a new weighting scheme based on this law. In Subsections 3.3.4 and 3.3.5 the two Zipf-based approaches are introduced. Results of applying this weighting scheme on several datasets and related conclusions close this section.

3.3.1 Introduction

In literature, several research efforts can be identified that try to explain various universal regularities that characterize texts originating from different domains and languages [Baayen, 1991]. The best-known research results were obtained by George Zipf [Zipf, 1949], who derived a law describing the distribution of word frequencies in a document or document collection, according to which the frequency of the terms decreases inversely to the rank of the terms. Zipf's law is discussed in the next sections. In a later stage, Mandelbrot mathematically and conceptually improved Zipf's model [Mandelbrot, 1961]. Beyond Zipf's law describing term frequencies, other notable identified regularities are Heap's law and the bursty nature of words. Heap's law describes the growth of a vocabulary size as a function of the number of words in a document collection. The burstiness of words concerns the increased likelihood of a word to appear in a document, if it has already appeared in a document considered its overall frequency across the entire collection [Baayen, 2002]. In the last decade, an increased interest is noticed in the more general concept of LNRE distributions, which founds the previously mentioned laws. This concept was introduced by Khmaladze [Khmaladze, 1988], and led to so-called LNRE models. These models form a specialized collection of statistical models and are used to estimate the distribution of occurrence probabilities of words based on a finite set of samples, such as documents.

3.3.2 Zipf's law

Zipf's law is the empirical law that is related to the Zipfian distribution which is one of the families of related discrete power law probability distributions. It states that, regarding the vocabulary of a document collection, the frequency of any word in the vocabulary is inversely proportional to its rank in the sorted vocabulary.

When this phenomenon is graphically displayed, a so-called Zipf curve is obtained (as indicated in Figure 3.21).

Two separate Zipf laws can be identified, each describing the conduct of the Zipf curve in the high and low frequency range of the graph [Booth, 1967].

3.3.2.1 First Zipf law

The first Zipf law, often considered as the only Zipf law [Manning and Schtze, 1999], describes the high frequency range of the graph.

If C represents a collection of texts, then Zipf's first law [Booth, 1967] is defined as:

Definition 3 *Let*

- f_i *be the frequency of a term* i *in a full document collection*
- r_i *the rank of the term* i *in an ordered list of all terms, according to the frequency*

then there exists a constant cte for which

$$f_i \cdot r_i = cte \tag{3.15}$$

This definition describes a proportional relation between f_i and $\frac{1}{r_i}$. This indicates that, in statistical terms, the most frequent term appears twice as often as the second most frequent term.

3.3.2.2 Second Zipf law

As the first Zipf law is only valid for the high frequent terms, the second law is valid for the low frequent range [Booth, 1967, Moyotl-Hernández and Héctor, 2004]. This law is used to estimate the number of terms appearing q times, based on the number of terms appearing only once. This law is based on the reasoning described in the following paragraphs.

The number of terms appearing only once in a text j containing m_j terms is delimited by the following inequality

$$2 > \frac{m_j \cdot cte}{r_i} \geq 1 \tag{3.16}$$

with r_i the rank and cte the constant of the Zipf law. From this equation, a minimum and maximum rank can be derived for terms appearing only once in the vocabulary, which results in $r_{min} = \frac{1}{2} \cdot cte \cdot m_j$ and $r_{max} = cte \cdot m_j$.

If I_1 describes the number of terms that have a single appearance in the text, then this number can be expressed in function of this minimum and maximum

rank:

$$I_1 = r_{max} - r_{min} \quad \text{which is equal to:} \tag{3.17}$$

$$I_1 = \frac{1}{2} \cdot cte \cdot m_j \tag{3.18}$$

For a frequency of f, a similar reasoning leads to

$$I_f = \frac{1}{f(f+1)} \cdot cte \cdot m_j \tag{3.19}$$

This results in the formulation of the second Zipf law, which describes the number of terms with frequency f in function of the number of terms with single appearance as follows:

$$\frac{I_f}{I_1} = \frac{2}{f(f+1)} \tag{3.20}$$

3.3.2.3 Interpretation of the Zipf laws

Related to this statistical pattern, an empirically proven relationship between the frequency of terms and its informational value is observed [Booth, 1967]. Terms appearing very frequently or rare are considered to convey little information. In contrast, averagely frequently appearing terms are considered to contain the most informational value. The most relevant terms thus tend to be found towards the middle of the frequency range of a domain vocabulary, sorted by decreasing frequency. If such an ordered domain vocabulary of a document or document collection is obtained, an informational curve can be fitted on the related Zipf or frequency curve. The frequency pattern and the related informational value or resolving power are graphically illustrated in Figure 3.21. The previously mentioned frequency ranges are indicated in this figure. No clear guidelines are given how to delimit these ranges.

3.3.3 Motivation

In Section 2.2.2.2 an overview was given of the TFIDF weighting scheme, and a graphical representation of this scheme is shown in Figure 3.22(a). This figure represents the application of TFIDF to the terms of a document, given that the document consists of 100 indexed terms, and a total of 90 documents is contained in the indexed document collection. As indicated in Section 2.2.2.2, the motivation behind this weighting scheme is to assign high weights to terms that frequently occur in a small number of documents.

Judging from Figure 3.22(c), compared to the Zipf law TFIDF favors a different selection of terms. The contour plot shows a high density in the lower left quadrant, corresponding to the rapidly increasing scoring of terms in that quadrant. A small

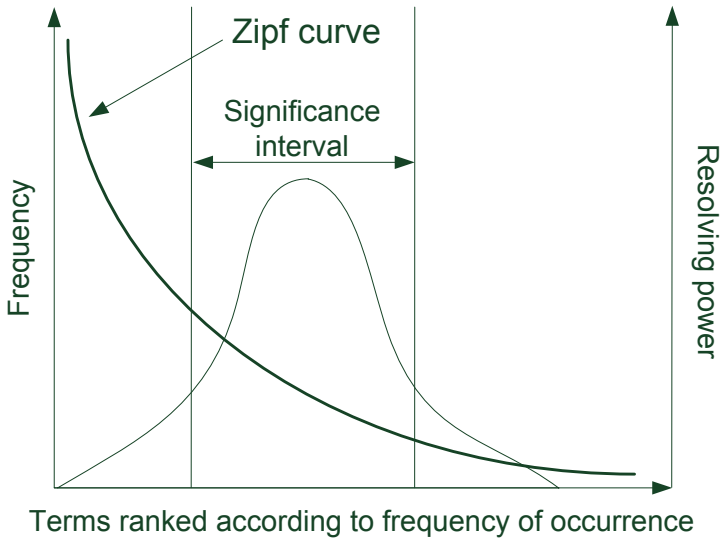


Figure 3.21: Graphical representation of a Zipf Curve and its related significance interval

change in the number of documents containing a specific term has a considerable effect on the weight of that term.

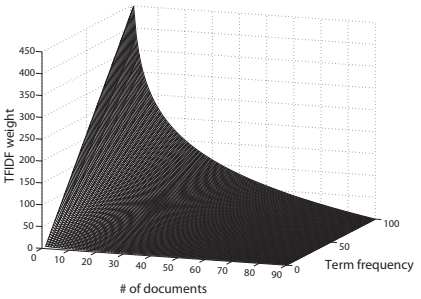
To have the optimal selection of terms according to Zipf, the terms must be roughly represented in following order of descending favor [Church and Gale, 1995] as indicated with numbers in Figure 3.22(c): from the lower left quadrant, over the lower right or the upper left quadrant, to the upper right quadrant. This reasoning is based on the average term frequency in these quadrants. In the next sections, two approaches are presented that follow this order.

3.3.4 First approach to a mid-frequency favoring weighting scheme

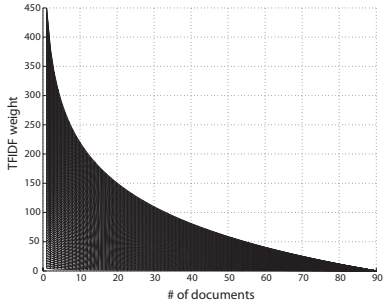
Based on the significance of mid-range terms explained in the previous section, a new scheme was developed to obtain an improved weighting of the terms. As explained in the previous section, dimensions represented in the lower left quadrant of Figure 3.22(c) are considered the most resolving among the domain vocabulary.

Considering the importance of those terms, terms in the lower left quadrant should score significantly higher than the other quadrants of Figure 3.22(c). The left upper and the right lower should have an equal weighting, the last quadrant should receive the lowest scores.

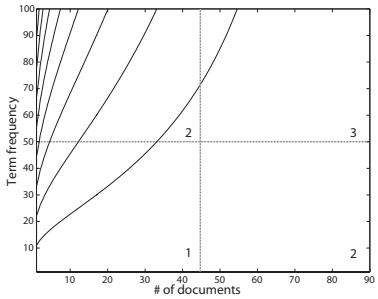
The proposed new weighting scheme consists of three factors:



(a) 3D representation of a collection of 90 documents and 100 indexed terms per document



(b) 2D projection of the TFIDF weighting scheme



(c) Contourplot of the TFIDF weighting scheme.

Figure 3.22: 2D and 3D representations of the TFIDF weighting scheme

$$w_{i,j} = \overbrace{\frac{1}{\sqrt{f_{i,j}^2 + n_i^2}}}^{1} \cdot \underbrace{n_i \cdot itf_{i,j}}_{2} \cdot \overbrace{f_{i,j} \cdot idf_i}^{3} \tag{3.21}$$

In this formula n_i is the number of documents that contain term i, and $f_{i,j}$ denotes the raw frequency of term i in document j. The factors idf_i and $itf_{i,j}$ stand for Inverse Document Frequency of term i and Inverse Term Frequency (ITF) of term i in document j, respectively. The ITF component itf_i of term i is a function of the number of terms m_j a certain document j contains and the term frequency of term i in document j:

$$itf_{i,j} = \log \frac{m_j}{f_{i,j}} \tag{3.22}$$

Factor 3 of the new weighting scheme is identified as the TFIDF factor as discussed

in Section 2.2.2.2. Factor 2 is composed of the Document Frequency (DF) - ITF, therefore rare terms occurring in a document are favored. The overall behavior of this component is favoring rare terms occurring in a considerable number of documents. As mentioned in Section 2.2.2.1, to account for varying document length, the TF- and ITF-component are normalized by the number of dimensions occurring in the document.

The result of combining Factor 2 and 3 is a weighting scheme that favors all mid-range dimensions occurring in an average number of documents. Disadvantage of this weighting scheme is the overall scoring: the scheme favors too many terms, and there is no significant scoring difference between interesting and non-interesting terms. Factor 1 therefore is added to shift the maximum towards the origin and to increase the scoring difference. The overall function of the scheme is:

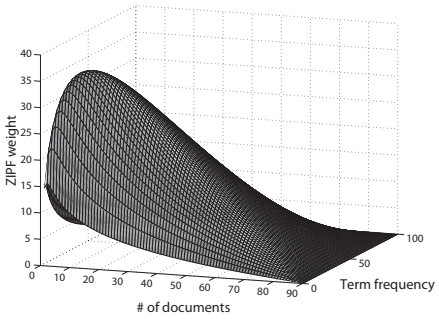
$$w_{i,j} = \frac{n_i \cdot f_{i,j}}{\sqrt{f_{i,j}^2 + n_i^2}} \cdot \log \frac{m_j}{f_{i,j}} \cdot \log \frac{n}{n_i} \tag{3.23}$$

In this function n represents the number of documents in the collection, m_j is the number of terms (or dimensions) in document j.

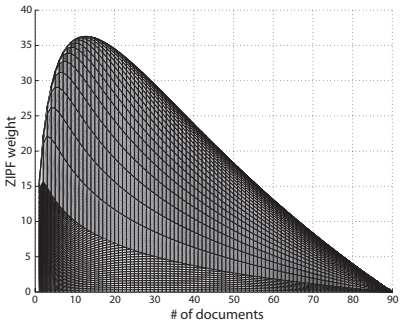
A graphical representation is shown in Figure 3.23. This figure represents how the proposed weighting scheme is applied to the terms of a document, given that the document consists of 100 indexed terms, and a total of 90 documents is contained in the indexed corpus. The contourplot is shown in Figure 3.23(c). The contourplot shows a high density gradient in the lower left quadrant, indicating the fast increasing scoring of terms in that quadrant. From this figure, it is clearly shown that this weighting scheme assigns the highest weighting score to the mid-range terms.

3.3.5 Zipf-based weighting scheme

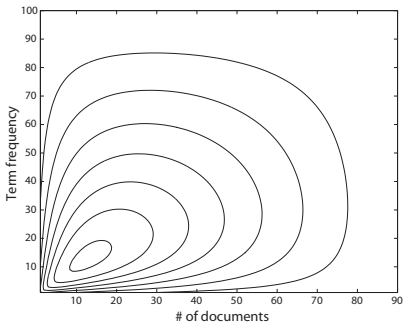
The approach presented in the previous section however still does not select all terms that are considered to be interesting according to the Zipf law. Considering Figure 3.23(c), not only the region around the origin but also the terms covering the regions closely to the X- and Y-axis need to obtain a high weighting. Therefore another approach was developed, based on the mathematical descriptions of Section 3.3.2. As described in that section, two entirely different rules describe two extremes of the term distribution. The first law is valid in the region of the high frequent terms, as the second law is valid in the opposite low frequent region. The frequency which delimits the two regions is called the Transition Point (TP) [Moyotl-Hernández and Héctor, 2004]. This frequency value splits the domain vocabulary into two collections of terms, based on studies of the Zipf law [Zipf, 1949, Booth, 1967]. Based on the hypothesis of Booth, the features whose



(a) 3D representation of the ZIPF weighting scheme of a collection of 90 documents and 100 indexed terms per document



(b) Projection of the 3D representation of the ZIPF weighting scheme



(c) Contourplot of the ZIPF based weighting scheme.

Figure 3.23: 2D and 3D representations of the ZIPF based weighting scheme.

frequency are closer to the TP of the collection need to obtain a higher weighting compared to other terms.

To obtain the TP, the following reasoning is used [Moyotl-Hernández and Héctor, 2004] : Equation 3.20 is used to provide an estimate of the number of terms of frequency f in the lower frequency region. Applying a limit for I_f towards 1 (or f reaching the highest possible frequency in the text)), the frequency can be obtained by splitting the domain vocabulary in

higher and lower frequency terms:

$$\lim_{I_f \to 1} \frac{I_1}{I_f} = \frac{f(f+1)}{2} \tag{3.24}$$

which results in: (3.25)

$$f_{TP} = \frac{(-1+\sqrt{(1+8I_1)})}{2} \tag{3.26}$$

In this formula, f_{TP} indicates the TP-frequency.

Using this frequency, the following weighting scheme for term i appearing in document j has been developed.

$$w_{i,j} = \frac{f_{i,j}(k1+1)}{K+f_{i,j}} \cdot e^{-(\frac{f_i - f_{TP}}{p \cdot m})^2} \qquad \text{with: } K = k1((1-b) + b\frac{\sum_{i=1}^{m_j} f_{i,j}}{\sum_{j=1}^{n}\sum_{i=1}^{m_j} f_{i,j}} \tag{3.27}$$

In this equation $w_{i,j}$ is the weighting for term i in document j. $f_{i,j}$ is the raw frequency of term i in document j, and f_i is the sum of the raw frequencies of term i in the domain vocabulary with size m. p is the parameter that defines the region around the transition frequency f_{TP}. K is the scaling factor of the BM25 weighting scheme, thus $k1$ and b are BM25 scaling parameters. $\sum_{i=1}^{m_j} f_{i,j}$ and $\sum_{j=1}^{n}\sum_{i=1}^{m_j} f_{i,j}$ are the length of document j and the average document length.

This weighting scheme is based on the BM25 weighting, wherein the relevance feedback factor (Section 2.2.2.2) is replaced with the Zipf component $e^{-(\frac{f_i - f_{TP}}{p \cdot n})^2}$.

3.3.6 Validation

A series of experiments were conducted in order to evaluate the effectiveness of the second proposed weighting scheme for the support of clustering operations. Different datasets were used in these experiments and uniform preprocessing, clustering and evaluation procedures were applied to these datasets.

3.3.6.1 Datasets

The validation procedure was composed of two steps:

1. In the first scenario several smaller test sets were made of the Ohsumed and Reuters datasets. The properties of these test sets are shown in Appendix A. The rationale behind these test sets was to create different sets with an increasing number of documents and dimensions.

2. In order to test the weighting scheme, a second test scenario was constructed to validate the weighting scheme in a clustering process with larger datasets. The large datasets originate from the TREC dataset collection, and they

were chosen based on the variety of the content in the various articles. The properties for these datasets are also described in Appendix A.

3.3.6.2 Experimental setup

The object of this comparison is to validate the improved cluster performance of the proposed weighting scheme compared to other weighting schemes in datasets with varying dimensionality and size. Each of the constructed test sets therefore results in a series of term-by-document matrices, with growing dimensionality and size. The clustering algorithm selected for these experiments is hierarchical agglomerative clustering (see Section 2.4) with following properties:

- the cosine measure was used to calculate the pairwise distances
- Wards linkage was used to link clusters together
- the number of clusters was set to the number of topics present in the index

The output of the clustering algorithm is a number of document groups, equal to the topics present in the specific data set.

Several weighting schemes are used in this validation procedure. For the smaller datasets in the first test scenario, a comparison between the proposed and TFIDF weighting scheme is obtained. For the larger datasets in the second test scenario, besides TFIDF, raw frequency and the BM25 weighting scheme were also used in the validation procedure.

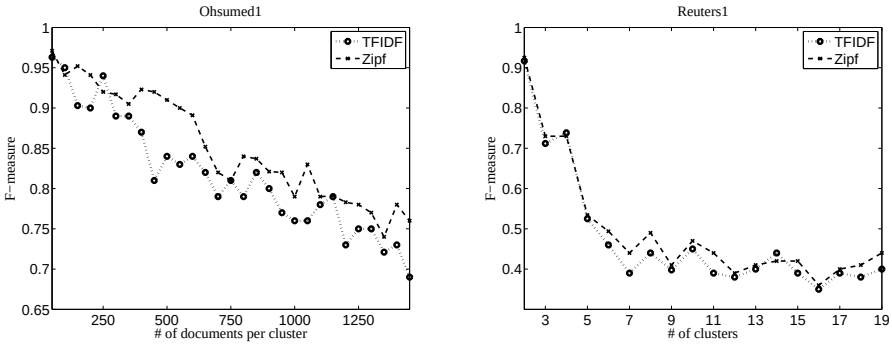
The results will be explained in the next section.

3.3.6.3 Validation measure

For each test set all documents were selected so as not to belong to more than one topic. Therefore it is assumed that each topic is represented in the clustering result by one cluster of documents. Based on this assumption, the F-measure is used to quantify the quality of each cluster in the clustering result.

3.3.6.4 Results

Figure 3.24(a) displays the average cluster quality for the first test scenario based on OHSUMED. The cluster quality of the results obtained with the Zipf-based weighting scheme are noticeably better than the results obtained with the TFIDF scheme. For all test sets, the cluster quality measurement scores higher with the proposed weighting scheme. These results provided an indication that the performance of the new scheme in datasets with low dimensionality (up to 4500 terms) is better than the performance of TFIDF. Also noticeable is that the average F-measure score of the new weighting scheme for this experiments remained higher than 0.75.



(a) Comparison of the proposed weighting scheme with the TFIDF weighting scheme for the Ohsumed1 test environment

(b) Comparison of the proposed weighting scheme with the TFIDF weighting scheme for the Reuters1 test environment

Figure 3.24: Results of the application of the Zipf based weighting scheme on the Reuters1 and Ohsumed1 test environment.

The experiments based on the Reuters based test collection are summarized in Figure 3.24(b). This figure shows a more similar evolution of the F-measure scores between the two weighting schemes. For most of the test sets, however, the new weighting schemes outscored the TFIDF scheme. These results gave an indication that the average performance of proposed weighting scheme in datasets with a higher dimensionality (up to 14000 terms) is better than the performance of the TFIDF weighting scheme, but the difference between both is less pronounced. Noticeable is the low range of the F-measure for this test environment. This is likely due to the much broader topic definition of the overall dataset.

As previously described, the large datasets for the second test scenario were chosen based on the variety of the content. Table 3.3 presents the results of these experiments for these datasets. As indicated in the table, the comparison was performed with four different weighting schemes: besides the proposed scheme, TFIDF, BM25 and raw frequency were used. In this test scenario, Zipf obtains for two datasets the highest scoring: Re0 and K1b. For BM25 and TFIDF, the scoring is the highest in three datasets. For all datasets except the La1, the average F-measure scores within a considerable range for all weighting schemes.

The results thus indicate that the proposed weighting scheme is not longer outscoring TFIDF. Two datasets result in the second highest scoring, while for three datasets the second lowest scoring was obtained. The results indicate that for these datasets the proposed weighting scheme is not outperforming the three other weighting schemes. Further experiments on a larger scale are required to validate this weighting scheme.

Dataset	BM25	Zipf	TFIDF	Raw
La1	0.39	0.34	**0.42**	0.25
La2	**0.54**	0.49	0.50	0.31
Re0	0.59	**0.63**	0.56	0.55
Re1	0.56	0.52	**0.57**	0.37
K1a	0.63	0.64	**0.69**	0.48
K1b	0.69	**0.74**	0.70	0.67
Wap	**0.69**	0.62	0.66	0.66
Tr31	0.66	0.61	**0.69**	0.34
Tr41	**0.70**	0.54	0.64	0.50

Table 3.3: Comparison of the proposed weighting scheme with the BM25, TFIDF and Raw weighting scheme on 9 high dimensional datasets

3.3.7 Conclusions

Two approaches for a new weighting scheme for document clustering were introduced. The main idea of this scheme is to favor the mid-range terms of the domain vocabulary, based on the law of Zipf. Two experimental environments based on two well-known datasets, OHSUMED and Reuters, were constructed to test the resulting weighting scheme. The domain vocabularies ranged from 867 to 10705 stems. For all test sets, the average cluster quality was calculated with the F-measure. Compared to the popular TFIDF weighting scheme, the first experiments on standardized datasets indicate that this new weighting scheme, used with a hierarchical clustering algorithm, results in cluster results of higher quality. Experiments with larger datasets however indicated that the resulting cluster quality is not always better in high-dimensional environments compared to other weighting schemes, such as BM25 and TFIDF. In this manner, the conclusions for the smaller datasets were not confirmed for the larger datasets.

Chapter 4

Applications

4.1 Metadata Extraction from Document Content

4.1.1 Introduction

As stated in the introduction of this dissertation, information and knowledge of a company are considered to appear in two general forms: tacit and explicit knowledge. In the category of explicit knowledge not only the information expressed in the content of the document plays an important role: information about this content-related information also can provide a base for further knowledge gathering. This information, also called metadata, becomes increasingly important in the handling of documents in a KMS or Document Management System (DMS) because it enhances the accessibility of the information and knowledge in these systems.

The identification of metadata is often performed manually by the creator of a document, or automatically by retrieving information from the operating system. Examples of metadata fields of a document are the date of creation or the name of the author. These fields are however solely based on the context or environment of the system. Other metadata fields can be identified based on the content of the document itself.

The objective of the techniques presented in this chapter is to provide a supporting role for the end user in the identification of these 'content-driven' metadata fields. This research objective was set by a commercial research partner, developing solutions for document handling and creation. To adapt their DMS to the need of the customer, the identification process of the metadata fields is, to a large extent, still performed manually. Today this intervention is mainly based on interviews with workers in different roles in order to construct the optimal

document structure. This process can take up to several days, depending on the size of the company. This manual work has some serious drawbacks: apart from the time invested by the consultants and the company employees, the final structures depend on the subjective view of the interviewed employees, and they are also influenced by the experience and the number of interviewed employees. Based on these drawbacks the research was initiated to develop a methodology to support and semi-automate this metadata identification procedure.

4.1.2 Document model

The identification process of the metadata fields for a DMS has two objectives:

1. identifications of document types
2. assigning metadata labels to each document type

A document type is defined as a representative of a group of 'similar' documents. Similar documents mean either similar in the content they describe or in the layout they contain. Documents similar in content are, for instance, invoice documents, sent by different companies. Documents containing the same template are examples for similarity in layout. In Figure 4.1 a representation is given of these concepts.

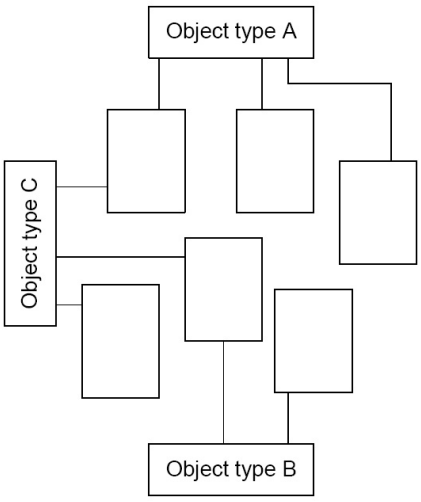


Figure 4.1: Graphical representation of the concept 'Document Type'

This collection contains three document types. With every document type, several actual documents can be linked. These documents are represented as

empty rectangles. As indicated in this figure, a document can belong to more than one document type. The main idea behind these multiple relations is that the system can provide multiple views on the document collection present in the system. These viewpoints result in different ways of structuring the documents in a corpus covered in a KMS. It is possible to first structure the corpus according to the projects currently being executed, or to restructure it according to the clients of the company. These different viewpoints can also be associated with different roles of employees using the system.

The second objective is the identification of a set of metadata labels for every document type. These labels are the properties of the document type, describing the documents in more detail. For example, an invoice is defined by items such as the company name, account number and project type related to this invoice. These relationships are shown in Figure 4.2.

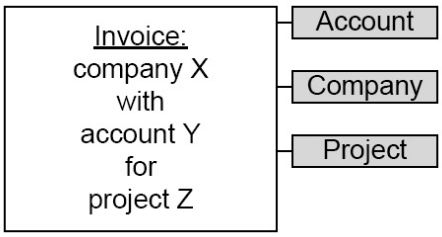


Figure 4.2: Graphical representation of the concept 'Labels'

The combination of the properties and the document type defines a 'document model'.

The metadata labels can also be used to create different viewpoints on the document collection. The documents assigned to one or more document types inherit all the labels defined by these related document types.

For every label, a value needs to be assigned by the user or creator of the document. The complete result of the identification process is given in Figure 4.3. The already mentioned double relationship with the document types in the used example is complemented with actual values for the three identified labels.

4.1.3 Concept of multilayer clustering

As described in Chapter 2, clustering is a possible technique to reveal a structure in unordered data. For documents, a clustering result is governed by a set of important terms. Due to the different weighting of the terms of a document, the higher weighted terms will have a higher influence on the clustering result. This set of terms forms the base for the presented technique: these terms can

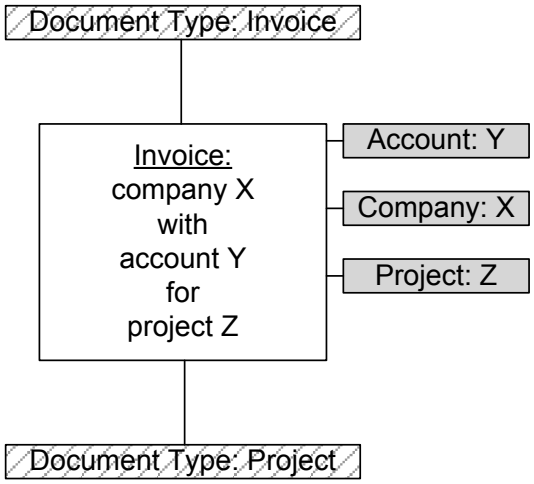


Figure 4.3: Graphical representation of a document connected to two document types and three meta-data labels.

provide indications for the different document types and metadata labels in the collection. If a clustering process is iteratively applied on a document collection, more indications of appropriate document types and related metadata can be obtained.

The proposed technique based on this assumption is called a multilayer clustering process. This process is defined as a repetitive clustering process, whereby in every clustering operation, the terms defining this clustering result are skimmed from the domain vocabulary.

For every clustering step in this process, two types of information are stored:

- the obtained clustering structure
- the set of terms defining the clustering structure

These types of information are the base of the importance concept in this process, the so-called multilayer cluster. A formal definition of this concept is provided in Definition 4.

Definition 4 *Suppose l repeated cluster operations, with C_i an identified cluster in the ith repeated clustering.*
A multilayer cluster MLC in a collection of documents is defined as the collection of documents assigned to a set of cluster results $\{C_i\}_{i=1\rightarrow l}$
with the following condition:
$\forall$ *document doc:* $doc \in MLC \Longleftrightarrow \forall\ i = 1 \rightarrow l$*:* $doc \in C_i$

In other words, a multilayer cluster is the stable or common region of the repeated cluster process. This reasoning is presented in Figure 4.4. In this figure, three

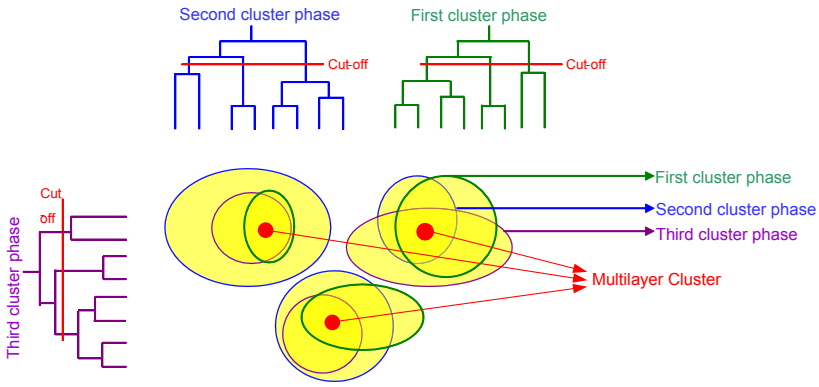


Figure 4.4: Graphical overview of the multilayer clustering concept. Every stable or common region of the clusters is indicated with a dot.

consecutive clustering steps are performed. Every clustering phase results in a different set of clusters. Every stable or common region of the clusters, as a result of three repetitive clustering processes, is a multilayer cluster.

4.1.4 Multilayer clustering process

Figure 4.5 summarizes the followed methodology for identifying a number of document types and a list of possible metadata for each document. In the following paragraphs, the different procedural steps of this methodology are explained.

4.1.4.1 Document by term matrix transformation

As explained in Section 2.2.2.1, a sparse DTM matrix and the related distance matrix, based on the cosine distance measure, can be obtained of the complete document collection. The applied weighting scheme is the TFIDF weighting scheme.

4.1.4.2 Hierarchical clustering

The clustering technique used in this process is the agglomerative hierarchical clustering technique with average linkage (see Section 2.4). In every iteration the clustering process is performed and the resulting dendrogram is intersected according to a predefined number of clusters. This parameter will be described later in the results section.

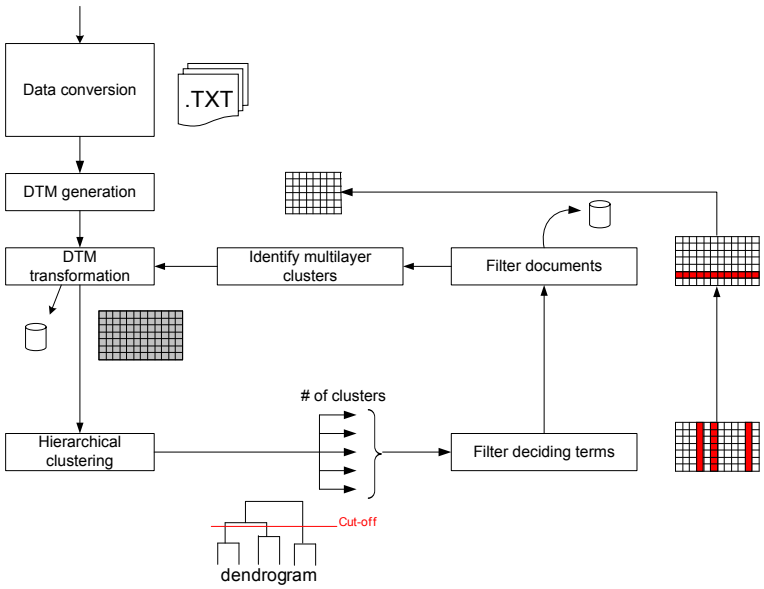


Figure 4.5: Overview of the multilayer clustering methodology.

4.1.4.3 Filtering

Due to the different weighting of the terms of a document, the higher weighted terms will have a higher influence on the clustering result. Once a clustering is determined, these highest weighted terms are filtered from the domain vocabulary and the DTM, wherein the related weights are replaced by zero-values. The selection of the highest weighted terms is based on a weight threshold. The number of iterations is also predefined, or is governed by cluster quality criteria. Many criteria, based on compactness and isolation, are available in literature, such as minimum total distance or separate clusters [Jain et al., 1999]. If this minimal cluster quality is not met, the process is halted.

4.1.4.4 Identification of multilayer clusters

Identifying multilayer clusters involves identifying so-called stable regions of the cluster results of the repeated cluster operations. A stable region of a clustering is defined as a set of documents that systematically cluster together when a clustering process is repeated multiple times, with or without perturbation of the data collection [Ben-hur and Guyon, 2003]. A Multilayer Cluster (MLC) is thus a stable region of multiple repeated clustering processes wherein terms are filtering after each clustering step.

An example of multilayer clusters is shown in Table 4.1. In this example, three multilayer clusters are identified with four repeated cluster operations. The columns L1 to L4 are the four repeated cluster processes. In these columns, the labels of the different clusters in the different layers are shown. These cluster labels are only important in the same layer, e.g. A1 and A2 are different clusters in a different cluster process. The rows are the three identified multilayer clusters. Groups of documents with the same chain of cluster labels thus form a stable clustering. All documents combined in multilayer cluster MLC1 were grouped together in four repeated clusterings with the previously described filtering.

Introducing more layers or iterations in the clustering process, changes the granularity of the identified MLCs. For instance in Table 4.1, the documents related to MLC1 and MCL3 are most likely to be connected because they are clustered together in the first three of the four layers, meaning that in each of these layers the set of important terms defining those clusters was the same. Documents grouped together multiple times can be described by these sets of terms, which thus can provide indications for metadata of these documents. The relationship between these terms is described in the following section.

Table 4.1: This table shows the result of four iterative clustering processes, shown in the columns. From these clustering processes, three multilayer clusters are identified as displayed in the rows.

Cluster ID	**L1**	**L2**	**L3**	**L4**
MLC 1	A1	A2	A3	A4
MLC 2	B1	B2	B3	B4
MLC 3	A1	A2	A3	C4

4.1.5 Term network generation

After the preprocessing and multilayer cluster steps, the stable multilayer clusters combined with an equal number of relevant term collections are available for the generation of a term network. The objective of this term network is to identify the relevant relationships between these terms, based on the co-occurrences of the terms. The generation of the network proceeds as is indicated in Algorithm 23, located in Appendix G.1.1.

Starting from a DTM D obtained from the documents belonging to a multilayer cluster, a co-occurrence matrix is constructed. This matrix contains the term-term relationships, and is the adjacency matrix of the term network (see Section 6.2.2). The matrix construction process is illustrated in Figure 4.6. The rows and columns of the DTM are determined by the documents supporting the particular multilayer cluster and the terms determining this structure. This network plays a crucial role in the metadata identification process described in the following sections.

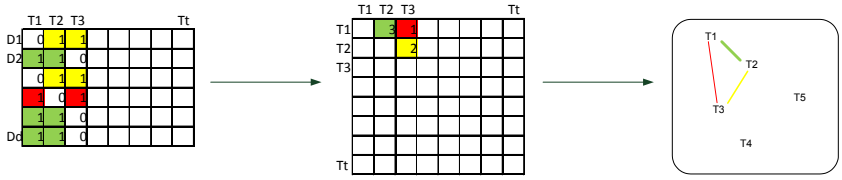


Figure 4.6: Graphical representation of the construction process of the term network.

4.1.6 Ideal conceptual term network

The information expressed by the term network forms the basis for the metadata identification. Ideally, the term network is structured as indicated in Figure 4.7. In this ideal network, the relations between, and the function of, the terms are clearly identifiable. The identification is based on the positions of these terms: the more central the terms are in the network, the more important they become in the identification of the metadata. The position of a term is based on the weights and the number of relationships with other terms. Based on such an ideal term network, context identification is facilitated. The second step, the contextual enrichment, follows from the documents that support the different MLCs.

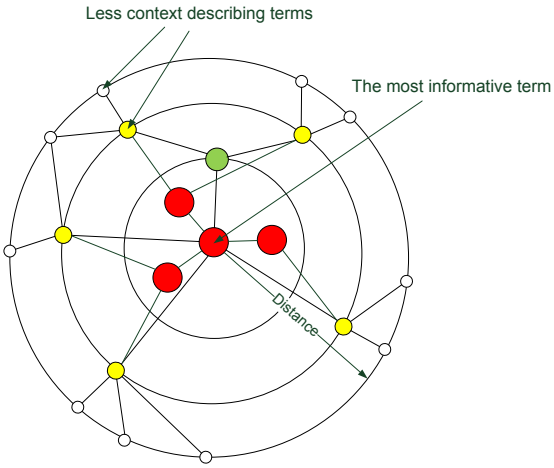


Figure 4.7: Representation of a term network. The more central the terms, the more information they contain concerning the documents they represent.

In practical cases however, these term networks often do not exhibit these properties. The weights of the relations in the network do partly indicate the

positions of the different the terms. Terms that are strongly related to each other should be related closely in the term network, while others should be located farther away. To approximate such an ideal network, the representation technique of Fruchterman-Reingold is used, which is a force-directed graph drawing algorithm [Fruchterman and Reingold, 1991]. This algorithm determines the positions of the vertices by applying attractive and repulsive forces between the vertices. The formulation of these forces is similar to the spring and electrical forces. The advantage of this technique is the ability to display the network in a two- or three-dimensional environment. This network indicates that topological related nodes are located in the same vicinity, while non-related nodes are located far from each other.

4.2 Identification of Document Types

Based on the concepts of MLC and term network introduced in the previous sections, a technique called Semi-Automated Metadata Extraction (SAME) is developed aimed at the identification of document types and models. In this section, several facets of this identification process of document types are explained.

The identification is based on the obtained MLCs. Every MLC is proposed to represent a possible document type for the document collection. In this assumption, the set of documents forming the MLC is represented by the related document type. The collection of terms determining the MLC can be used to identify the document type. In the ideal term network shown in Figure 4.7, the terms located in the center of the term network do suggest the document type of the MLC. These terms are the most important for this MLC, thus they contain the most information of the documents the MLC represents. A document type is thus supported by a number of documents, i.e. the collection of documents that were clustered together, the structure of which is based on a collection of discriminating terms.

The number of clustering layers is an important parameter for the SAME algorithm. The theoretical example shown in Figure 4.8 illustrates the influence of the number of layers on the perceived model structure. When a small number of layers is considered (e.g. 5), only a single document type can be identified with few associated terms in a term network. As the number of layers increases, the size of the term network increases. This increase can lead to a break-point where all important terms determining the current model are filtered. The absence of these important terms causes the document type to split (12 layers in Figure 4.8). This break point, also called 'view threshold', reflects a specific number of cluster layers where the view on the document corpus changes. A practical example for such a view point is a MLC describing a 'communication letter'. Increasing the number of layers beyond this view point, splits the document type into several

new types dealing with different types of communications such as 'order forms' and 'event invitations'. More examples concerning this effect are discussed in the results section.

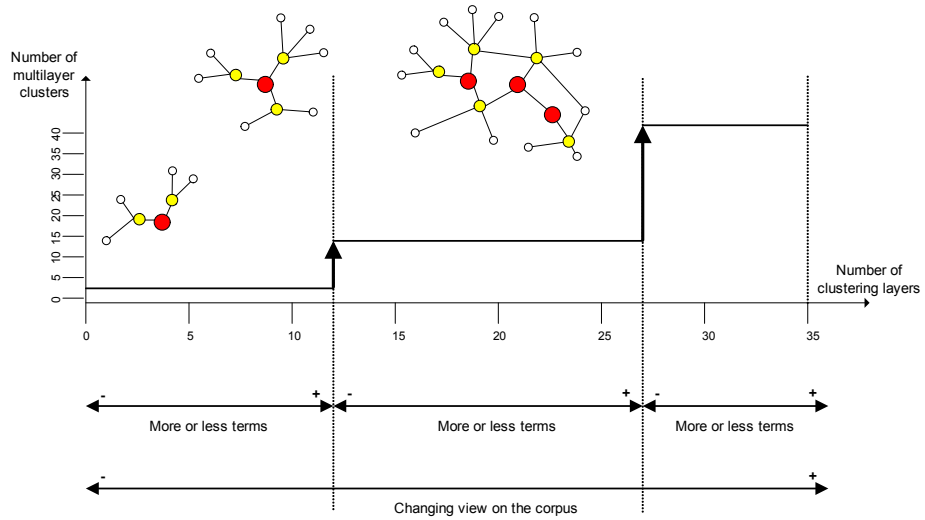
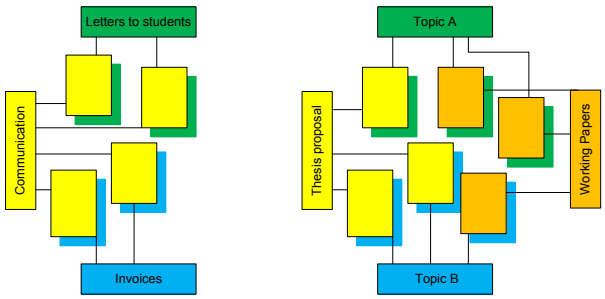


Figure 4.8: The influence of the number of cluster layers on the number of identified models

In the process of varying this parameter, two categories of document types can be identified (Figure 4.9):

- A document type can be split into several related subtypes (Figure 4.9(a)). The previously indicated example of 'communication letter' is an example of this category: for the first type, the terms that all communication documents have in common are identified as relevant. Once these terms are filtered out, the document type is split into several subtypes that are related to the original document type. This example is further discussed in Section 4.5.2.

- The unrelated subtypes do not share this straight subdivision property (Figure 4.9(b)). For example, a document type describes the projects running in the customer company. When the view threshold is reached, the obtained document types represent the different areas of expertise of the employees.



(a) Related models (b) Unrelated models

Figure 4.9: The collection of documents can be divided into two categories: a document can be split up into several related subtypes (Figure 4.9(a)), or no straight subdivision can be identified (Figure 4.9(b))

4.3 Identification of Document Properties and Property Values

The term network corresponding to every document type is also important for the identification of the document properties. These terms were important in the construction of the multilayer cluster, and thus explain the grouping of the documents. The main idea is that, based on the terms and relationships in the ideal term network, indications are provided for the model enrichment phase. These concepts are explained in the following paragraphs.

4.3.1 Model enrichment

The model enrichment process, determining which terms could be metadata labels and values, is based on less central terms in the term network. This identification process requires additional user interaction which is facilitated by the term network. The number of terms that indicates the properties of a document type is subject to the number of layers used in the process. As explained previously, varying this number influences the number of terms and therefore is an important factor in the interpretation phase. Domain knowledge is thus required to assess the appropriate property values. The user has to evaluate and classify these useful terms as model properties or values and to decide whether more detail is required.

An important aspect, however, is the general trend of decreasing relevance of the terms in a sequence of clustering layers. In Figure 4.8, the layer intervals between the viewpoints (i.e. intervals $[1, 11]$, $[12, 26]$ and $[27, 35]$) can be used to identify these terms and properties with varying detail. Changing the number of layers to be analyzed in these intervals, changes the details of the associated

document types. The information of the layers is available as the range of cluster layers is processed before the interpretation phase (as shown in Figure 4.8).

Besides the possibility to vary the number of terms based on the number of layers, applying a filtering parameter on the co-occurrence matrix is another means of influencing this number of terms. A certain co-occurrence frequency might not always provide the desired level of detail to describe the term-term relationships in the term network. By varying this parameter, a functionality is provided to change this level of detail. As stated before, this information is available as the complete range of cluster layers is processed beforehand.

4.3.2 Template identification

Important for the property identification process is the possible embedding of a template in documents. The use of templates results in very similar documents, and they are likely to be clustered together. This multilayer clustering will result in a set of terms describing this template. As an example, all invoices composed of a template have a list of properties in common, such as payment date, amount and account number. In order to identify those MLC, the following equation is used

$$\text{MLC template score} = \frac{1}{n_{MLC}} \sum_{i=1}^{n_{MLC}} \frac{|\text{Document properties present in document i}|}{|\text{Document properties}|} \tag{4.1}$$

In this formula, n_{MLC} represents the number of documents supporting the MLC related to the document model.

An example of this scoring is shown in Figure 4.10. Documents 1 and 3 contain four out of five labels identified for the document model, thus resulting in a score of 80%. Document 2 contains all labels resulting in a score of 100%. Setting a threshold value on the resulting score enables the identification of a template in a document.

The identification of templates is relevant in the context of filtering: the terms defining this model can be included in the filtering list in order to exclude the template from the identification process.

4.4 Validation

In this section, an overview is given of the validation of the SAME technique on three test cases. The structure of these cases is introduced in the first section. The preprocessing step and the applied validation procedure are described in the subsequent sections, followed by the results section. In the last section conclusions

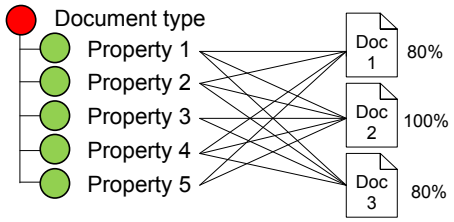


Figure 4.10: Template identification based on a document model

are drawn concerning the effectiveness and future work is discussed of the presented techniques.

4.4.1 Test cases

The used datasets in the validation process are labeled:

- 20Newsgroup
- Research group
- Quotes

In the following paragraphs, more information will be provided concerning these datasets.

4.4.1.1 20Newsgroup

The 20Newsgroup dataset is a collection of approximately 20,000 English newsgroup articles, equally divided over 20 newsgroup items. An extensive description of the structure can be found in Appendix A.

To illustrate the process of the SAME technique, a document collection containing four dissimilar topics was constructed. These topics are

- Crypto (991 documents)
- Atheism (799 documents)
- Automotive (990 documents)
- Baseball (994 documents)

In total, this collection contains 3774 English documents.

4.4.1.2 Research group

The documents in this collection were produced over a period of multiple years in the Centre for Industrial Management at the K.U. Leuven. The selection of these documents was performed by the secretary, without any profound knowledge about the actual research. These 2691 documents were located on the data storage facilities of the department and, as such, the following directory structure of this collection was available:

- Letters
  - order forms (314 documents)
  - general letters (1009 documents)
- Guide for students (7 documents)
- Timetable (135 documents)
- Registration for students (83 documents)
- Thesis proposals & contracts (343 documents)
- Thesis texts (35 documents)
- Working papers (675 documents)
- Meeting reports
  - Educational committee meeting (62 documents)
  - General assembly (15 documents)
  - Advice council (11 documents)
  - Ad hoc meetings (2 documents)

This partitioning does not reveal the actual content of the document collection in detail, for the following reasons:

- The collection of 'general letters' is largely composed of admittance letters (593). The remaining letters concern general issues, such as the planning of courses, arrangements for students concerning exam planning and public relations for the master programme offered by the centre.
- The guide for students is in fact a simple document, composed of 7 components served in separate files.
- The timetable was often published ad valvas for the students of the master programme, due to the changed planning of seminars and company visits. These documents often only contain a table structure.

- The collection of 'registration for students' contains documents concerning the practical arrangements in the registration procedure of the centre: documents such as information concerning the participants of language tests; or statistics of the admitted students reside in this collection.

4.4.1.3 Quotes

The Quotes collection contains project proposals of a company called <company name> in the glass industry. It covers the entire range of glass moulds in markets such as:

- Food
- Beverages
- Spirits
- Pharmaceutics

In total, this collection contains 5241 documents in the context of a large number of projects.

This dataset is structured in the following manner: For every prospect a project or quote is compiled. Examples of such projects are bottles for Bordeaux wines or neckrings. For each quote, all conversations and drawings are grouped together. These conversations are performed by email, and the drawings originate from specialized software or scans. These drawings are saved as a figure in a portable document format (pdf). The nature of the documents complicates the identification of document models: the conversations are frequently very brief, often only repeating the title of the drawing that is included as an attachment. Preprocessing of the figures often results in an empty document, because Optical Character Recognition (OCR) techniques were unable to identify the text in the figure.

The total number of projects included in this dataset is 5000. For around 1500 projects of this dataset, manually identified metadata on the level of the quote are supplied by the commercial partner. In general, three document types are identified for each quotation: request, official quote and drawings.

Summarized, a project or quote is characterized by means of

- Date of entry
- Customer
- Location
- Country
- Address

- Serial number of quote
- Serial number of order
- Title
- Request (date)
- Proceed (date)
- Amount (price)

No further contextual enrichment was available on the level of document types.

4.4.2 Preprocessing

Not all files are readily accessible and/or suitable for further processing. Following steps were performed in order to obtain a processable representation of a document:

- OCR: if the document only exists in a picture format, OCR techniques were applied to convert this format to a plain text format.
- Conversion: in order to obtain a processable ASCII format, all other document formats were converted to this format. For some of these file types, no converters were available. These files were omitted from the dataset.
- Term filtering on the content of a document was performed, as described in Section 2.2.3.2.

Each document of a dataset was subjected to this sequence of preprocessing steps in order to obtain a processable document representation.

4.4.3 Validation measure

As the nature of the proposed technique and the used industrial datasets imply a considerable share of subjective influence when validating the results, a pair of experts was asked to identify the different document models.

These results were validated using the Cohen's free marginal Kappa coefficient [Cohen, 1960]. This coefficient is used to validate the assignment of so-called cases into categories by two raters. This statistical measure of inter-rater agreement κ takes the agreement occurring by chance into account, and is calculated as

$$\kappa = \frac{P(a) - P(e)}{1 - P(e)} \tag{4.2}$$

In this equation, $P(a)$ is the relative observed agreement among the different raters, while $P(e)$ is the hypothetical probability of chance agreement among the

raters. This last probability is calculated based on the number of available cases and the chance that each rater is randomly assigning a case to a category. For the standardized dataset 20Newsgroup the underlying (topical) structure is known as this structure was derived by domain experts. Therefore a validation can be performed based on the precision and recall criteria.

4.4.4 Validation procedure

The following procedure for the industrial datasets was applied to validate the SAME technique:

1. A short introduction was given to the test person concerning the objective of the research, stating the aim of identifying metadata based on the content of documents.

2. Two example graphs of the 20Newsgroup dataset were given to explain the expected answering method.

3. Two question were asked to each test person:

    - Assign three domains, in descending order from general to specific, to a given term network. These three domains are necessary because similar terms are considered as an agreement. Agreement is reached if one of the domains from rater 1 matches with one of rater 2, even if the domains are not assigned on the same level.
    - Assign a number of metadata labels to the identified document type in step 2, and base this assignment on the term network.

A first Kappa score is calculated for the identification of the document types. The cases in the scoring were the different term networks, as each term network had to be related to a document type. The categories (or the possible document types to select from) originated from

- the obtained indications of possible structures in the document collection. Examples of these indications are the directory structure or (parts of) the results of the manual identification process.

- the union of the answers of the raters. This means that all answers of the raters are taken into account as possible document types.

An example is shown in Table 4.2. The two raters have assigned three levels, from a general to a specific level. The first rater related the term network to a document of a university, and more specific, of a department and a council. The second rater regards the same network differently: the answer is more broader as the most general domain is an organisation. The other domains do suggest

the context of a meeting in a university. In this example, agreement is reached as meeting and council are considered as similar terms. In this example the categories are the answers of both raters.

Table 4.2: Example of the first kappa score: the identification of a document type

Domain	**Rater 1**	**Rater 2**
Domain 1	University	Organisation
Domain 2	Faculty	Department
Domain 3	Council	Meeting

A second Kappa score is calculated for the identification of metadata labels. Similar to the first Kappa score, agreement for a metadata label was reached when both raters identified the label or a synonym of this label. This process is clarified in the example shown in Table 4.3. Suppose a document type called 'Report' was obtained in the previous identification phase. In the Table 4.3, the answers of rater A and B are shown. The raters both identified 'date' and 'subject'/'title' as labels of this document type. Label 4 is a blank label for the first rater, as this rater only assigned three labels to this document type.

The cases are the different identified labels assigned by the different raters. The categories are obtained as

- the union of the answers of the raters
- blank categories for every label a rater did not assign (to obtain the same number of raters for every case)

In the previous example, the categories are all the answers of both raters.

Table 4.3: Example of the second kappa score: the identification of the metadata labels

Labels	**Rater 1**	**Rater 2**
Label 1	Date	Subject
Label 2	Author	Date
Label 3	Title	Room
Label 4		Attendee

As indicated in Section 4.1.6, the presented networks to the raters were constructed with the technique of Fruchterman-Reingold. A filtering parameter is set on the weights of the links as to reduce the number of terms in the network. This value was set at 0.2 for the validation procedure, as this figure resulted in limited networks that still contained enough information for the interpretation by the experts.

The number of clusters for each clustering is set in the following manner. Initially, for every test case this parameter was set to 100, which is static in each iteration. This value is chosen in order not to lose any information concerning the document types. As the optimal number of clusters is often lower than this parameter value, a single large MLC and several singleton MLCs are often obtained. After several iterations, the initial large MLC is decomposed into several smaller MLCs in order to converge to a number of interpretable MLCs. Additional experiments are however required to validate this statement. To limit the number of iterations, a value of 50 was set for this parameter for the 20newsgroup and Quotes datasets. In this reasoning, for each case a maximum number of ten layer were calculated. The term networks presented to the raters were obtained from the layer in which the support of the highest MLC dropped below 20 % of the total document collection. As will be shown in the next section, this layer number was reached within a small number of iterations.

4.5 Results

In this section, the results of the three previously described test cases are summarized.

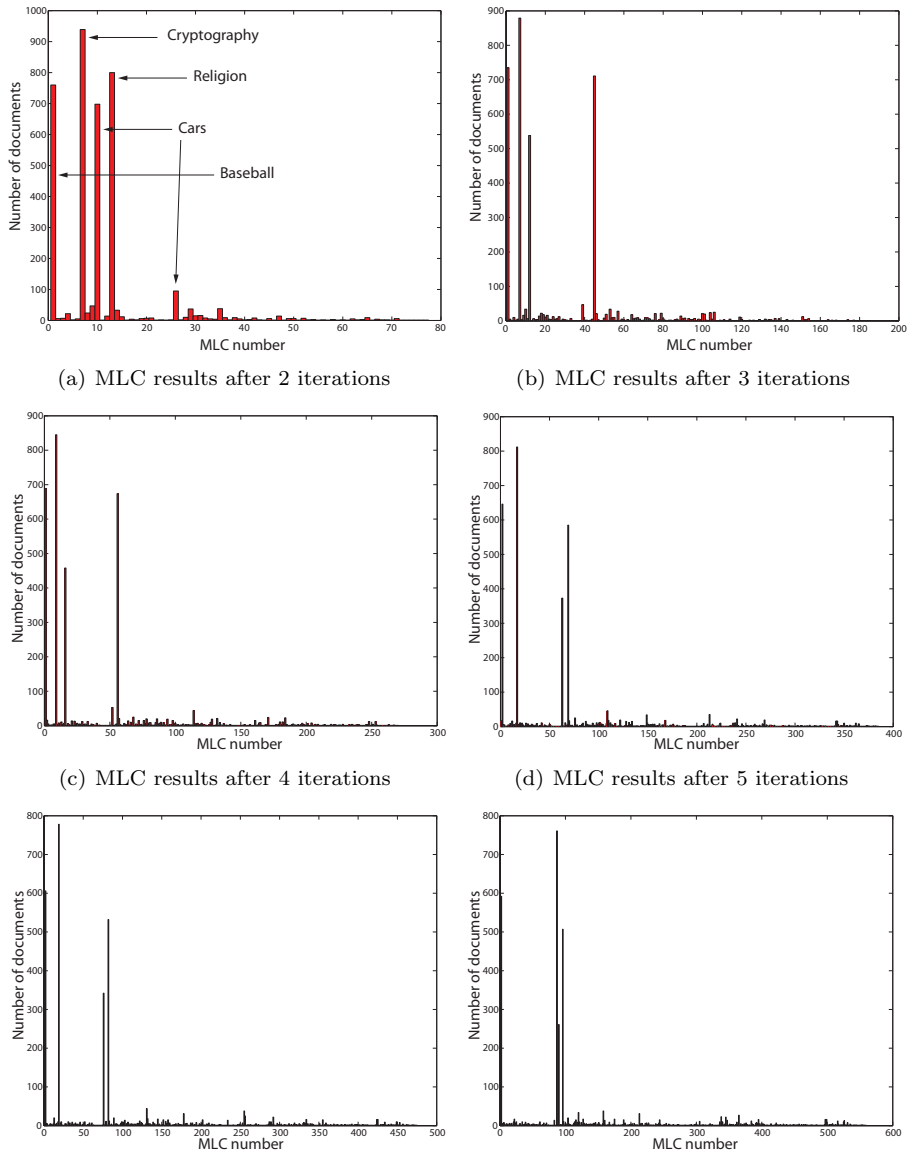
4.5.1 20Newsgroup case

The first objective of applying the SAME technique on this dataset is to identify the document types, based on the different obtained MLCs. As the structure is known beforehand, the expected document types are the selected categories of the Newsgroup dataset. In Figure 4.11, the number of identified MLCs and the related number of documents are shown for the first seven iterations. In these figures, the abscissa represents all identified multilayer clusters, while the ordinate indicates the support for each MLC. This support is expressed as the number of documents assigned to the multilayer cluster, or stated differently, the number of documents that were clustered together in each of the clustering iterations.

Figure 4.11(a) representing the results obtained after two iterations, with a number of 73 identified MLCs. This amount vastly increases up to more than 550 MLCs after seven iterations, as indicated in Figure 4.11(f). For a given multilayer cluster, the MLC number in every figure is different from the corresponding MLC number in other figures: if a MLC breaks down in multiple MLCs, they are inserted at the location of the original MLC. The position or MLC number can thus shift after an iteration.

The ordinate represents the support of the respective multilayer cluster.

In Figure 4.11(a), the apparent presence of the four topics is confirmed, suggesting the presence of four document types. After two iterations, the documents divided over the four largest multilayer clusters are allocated with the



(a) MLC results after 2 iterations

(b) MLC results after 3 iterations

(c) MLC results after 4 iterations

(d) MLC results after 5 iterations

(e) MLC results after 6 iterations

(f) MLC results after 7 iterations

Figure 4.11: MLC results for the 20Newsgroup dataset

following level of recall (the recall is calculated as the support of the MLC divided

over the number of documents in the related topic.)

- Baseball ($760/994 \simeq 0.76$)
- Cryptography ($939/991 \simeq 0.95$)
- Cars ($698/900 \simeq 0.78$)
- Atheism ($800/994 \simeq 0.80$)

After seven iterations, the related recall for each topic becomes

- Baseball ($507/994 \simeq 0.51$)
- Cryptography ($761/991 \simeq 0.78$)
- Cars ($261/900 \simeq 0.29$)
- Atheism ($592/994 \simeq 0.60$)

For all topics, the recall clearly diminishes during the multilayer clustering procedure which can be explained by the repetitive filtering of structure defining terms. However, the recall for the topic 'Cars' is clearly more effected by the MLC procedure. The decrease for this topic indicates that its compactness is less strict. The topic 'Cars' steadily decreases and in iteration 5 and 6 several view thresholds occur in the related multilayer cluster. An interesting view threshold, for example, implies a diversification between general car mechanics, and issues concerning the 'delco' of a car. In the sixth iteration, another split of general car mechanics occurs: posts concerning car cooling form a (small) separate multilayer cluster. A split also occurs based on template information of the newspost itself. Not all of these thresholds are thus as valuable.

Based on these figures and the recall statistics, the four document types can be identified. The identification process of a document type should however be facilitated by the presence of a domain specialist. As an example, the twenty most important terms (i.e. terms determining the clustering based on their weighting) from each of the first five layers of the largest MLC related to Cryptography are shown in Table 4.4. Besides the terms in italics suggesting the topic of cryptography, other terms require the assistance of a specialist. For instance, the term 'clipper' can be linked to 'chip' as the Clipper chip was an important encryption device for telecommunication in the nineties. Another example is 'ripem', which is an acronym for Riordan's Internet Privacy Enhance Mail, which uses public key cryptography to secure email. In the same category, PGP is another algorithm providing cryptographic privacy and authentication.

This analysis is performed for every MLC. The full list of terms obtained at the first layer for each of the MLC is shown in Table G.1 of Appendix G.1.1. In this list, the terms are ordered by decreasing importance, which is defined as the

Table 4.4: Overview of important terms from the first five layers for the MLC related to 'Cryptography'

Layer 1	**Layer 2**	**Layer 3**	**Layer 4**	**Layer 5**
key	law	strnlght	jim	fbihh
clipper	bill	*cryptography*	newsreader	clarinet
chip	information	brad	question	kadie
encryption	symmetric	qualcomm	ripem	tap
government	time	bits	group	rwing
keys	code	tapped	means	online
escrow	posting	house	idea	machine
netcom	amanda	phones	strong	metzger
access	bit	walker	news	cheap
nsa	message	marc	long	user
algorithm	computer	court	agree	*crypt*
phone	nntp	denning	life	omissions
public	technology	holland	robert	rubin
crypto	secret	agencies	post	details
security	org	mail	issue	excepted
sternlight	things	bontchev	hard	president
david	host	mit	technical	feds
system	read	fbi	laws	corporation
pgp	communications	hamburg	faq	speech
intercon	evidence	pat	text	brinich

Table 4.5: Selection of relevant terms for the four identified MLC

Crypto	**Atheism**	**Baseball**	**Car**
key	god	team	car
chip	atheists	game	oil
encryption	morality	baseball	engine
keys	religion	players	dealer
access	atheism	hit	ford
crypto	islam	win	auto
security	islamic	season	drive

average weighted frequency in the specific cluster corpus. Table 4.5 is a subjective selection of a domain specialist of the most relevant terms of Table G.1.

Based on the co-occurrence matrix of a MLC, the term network of this MLC is constructed providing the relevant information between the terms. Figure 4.12 is a part of a filtered term network of the multilayer cluster related to 'cryptography'. In Figure G.1 of Appendix G.1.1 the complete filtered network is shown.

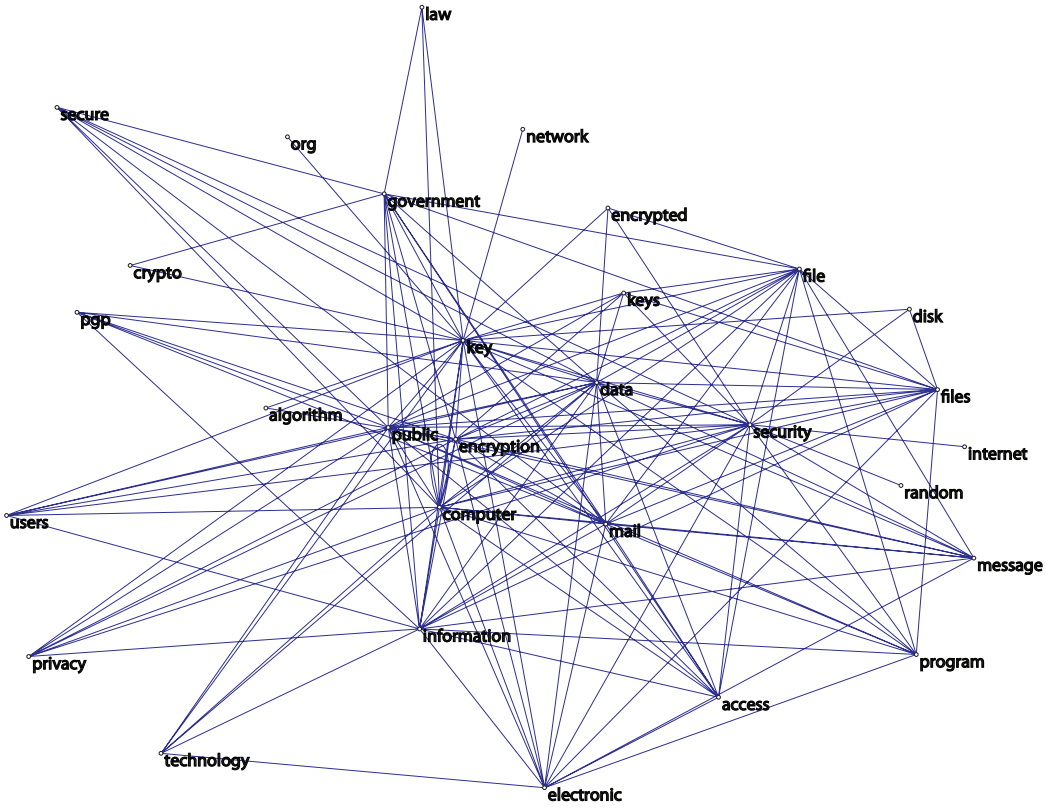


Figure 4.12: Reduced term network of the multilayer cluster related to 'Cryptography'

The obtained network is a practical approach of the ideal network discussed in Figure 4.7. The terms in the dense region, such as 'encryption' in this network, suggest the document type of cryptography. Several terms at the border of this dense region indicate the different possible metadata labels. Based on this reasoning, the list of terms and the term network, the document properties listed in Table 4.6 were identified by the expert for newsposts in the domain of 'cryptography'.

Table 4.6: Identified document models related to three of the largest MLC

Cryptography	**Baseball**	**Cars**
author	team name	brand
date	team suffix	car
email	league	component
organization	statistics	price
domain of application		insurance
type of cryptography		speed limit
algorithm or protocol		

The conceptual network extracted from the previous network and mapped on the ideal network, is displayed in Figure 4.13.

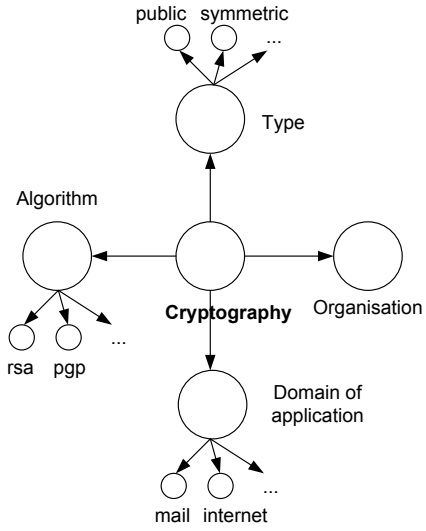


Figure 4.13: Conceptual term network of the multilayer cluster related to 'Cryptography'

From the list of obtained properties in Table 4.6, the first three properties are labels that can be derived without examining the content of the post. Most document systems already capture this category of properties as they can be extracted directly from the operating system.

The other identified labels are based on the following reasonings:

- organization: terms such as 'MIT' or government indicate the presence of the name of an organization in the post. The name of the organization can

be discussed in several contexts: from the development of an algorithm in MIT to the required knowledge of these algorithms in the NSA.

- domain of application: in the network several terms such as 'mail', 'message', 'internet' or 'data' suggest this general label. This label thus indicates the environment or application on which the encryption is applied.

- type of cryptography: the list of terms indicates the different subdomains of cryptography such as symmetric encryptions or public keys. This label thus indicates which type of cryptography is discussed.

- algorithm: related to the previous label, terms such as 'rsa' or 'pgp' describe an algorithm in the context of cryptography. This label thus indicates the type of algorithm that is discussed in the newspost.

For the other identified multilayer clusters, the document models are identified in the same manner. The different multilayer clusters and the related term networks can be found in Appendix G.1.1.

For baseball, the derived model is described in Table 4.6. For the property 'team name', possible values were obtained using the network, such as Phillies, Rockies and Marlines. Furthermore, the suffix of the team name always ends in 'white', 'red' or 'sox' and the team is playing in a minor or major league. The last property is subdivided in number of wins, losses and a so-called 'average'. These relationships are shown in the conceptual network shown in Figure 4.14.

A newspost related to cars can be described using the document model described in Table 4.6. The conceptual network is displayed in Figure 4.15. The most important properties of this model are the 'brand' and 'model', with values such as 'Ford Taurus'. Several newsposts include one or more car components in the discussion, combined with their price or the price of the car itself. A possible adaptation of this model thus includes the duplication of the property 'price' as a part of the property 'component', as is already performed with the property 'insurance'. These new posts often concern young drivers with sport cars so the insurance company and the price for their insurance are important issues in these posts. A different property for a car newspost is the 'speed limit', which often occurs based on the age of the drivers and their type of cars.

For the last MLC, a vague document model is identified. As known beforehand, this MLC is related to the topic of atheism. Only two properties can be identified for this document type, which are 'religion' and 'subject'. The label 'religion' can have 'islam' or 'atheism' as specific values, while the values 'god' or 'morality' can be assigned to the label 'subject'. The terms and the term network, shown in Figure G.2 of Appendix G.1.1, do not convey much specific information to suggest more properties.

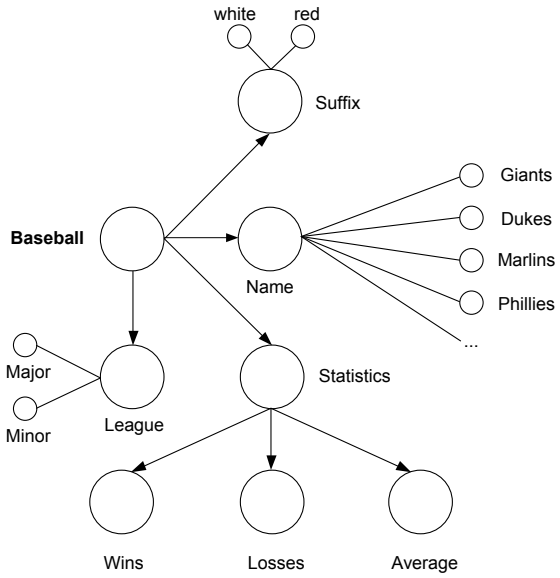


Figure 4.14: The conceptual term network of the multilayer cluster related to 'Baseball'

4.5.2 Research group case

After the preprocessing phase, the resulting DTM of this dataset contains 2571 documents represented in a vector space with a dimensionality of 73858 terms. To validate the obtained results with domain experts, the term networks of the largest MLC were presented to these experts to reach agreement on the document types and the related metadata labels. For this collection, two experts were asked to independently classify and to label these eight different term networks. The first expert is a PhD researcher in the involved division, often confronted with administrative tasks. The second expert is a secretary of the division, as she has a profound knowledge of the administration of the division. In Table 4.7, the answers to the eight term networks are displayed. These networks are displayed in Appendix G.1.2.

The result of this validation step is shown in Figure 4.16. The type of the ten most important MLCs is indicated, obtained after three iterations.

In Table 4.8 these MLC are described by their most discriminative terms.

In the following paragraphs, the identified document types and labels are discussed in more detail.

As indicated in Table 4.7, the first two term networks were assigned in full

Table 4.7: Results of the identified document types of the Research dataset, by two experts

Expert 1	**Expert 2**
Admission letters	Admission letters
Order form/ Invoice	Order form
Thesis proposal	Thesis proposal
Documents concerning processes	Research document concerning maintenance
Timetable	Timetable
Documents concerning logistics	Stock forms
Admission requests	Admission requests
Thesis contracts	Thesis contracts

Table 4.8: Overview of the identified MLCs of the Research dataset, with the descriptive terms for each MLCs

MLC	Selection of important terms
Letters: admittance letters	admittance, academic, program, application, registrar
Letters: Order form	invoice, delivery, person name, afdelingsverantwoordelijke, cib
Thesis proposals	thesis, proposal, students, company, field of study (ict, ese, ppm)
Thesis contracts	agreement, firm, thesis, students, company, field of study (ict, ese, ppm)
Articles: Maintenance & ecodesign	environmental, design, disassembly, cost, waste, cycle, maintenance
Articles: Logistics	magazijn, voorraad, logistieke, processen, levertermijn
Timetables	courses, seminars, semester, elective

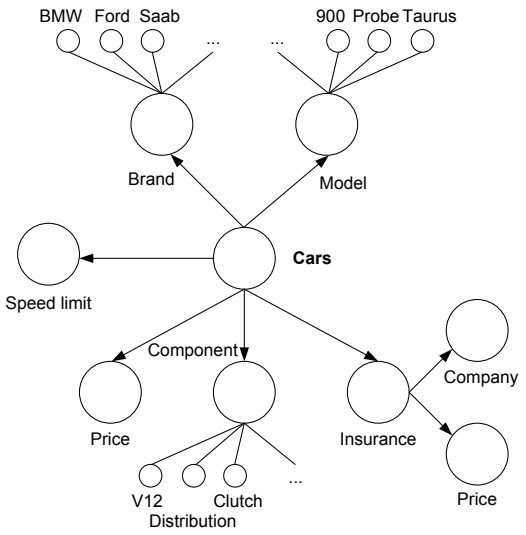


Figure 4.15: Reduced term network of the multilayer cluster related to 'Cars'

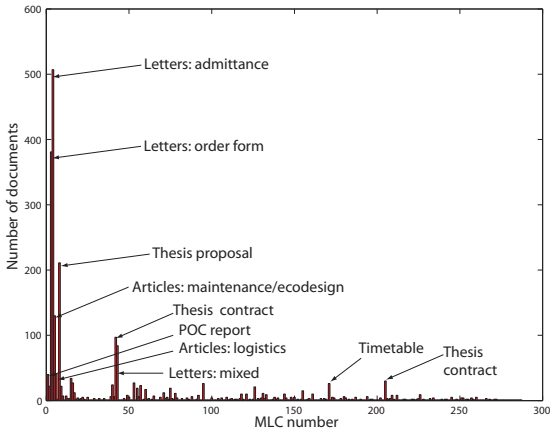


Figure 4.16: MLC results for the research group dataset, after 3 iterations

agreement to admission letters and order forms. Analysis of the MLCs indicated that, besides the largest, two other MLCs exist related to order forms. This remark underlines the required interactive step and the semi-automatic nature of the SAME technique. The difference between these order forms is located in the name of the head of the department at the time the order form was created.

Table 4.9: Overview of the document models for 'Admission letter', 'Order form' and 'Thesis proposals'

Admission letter	**Order form**	**Thesis proposals**
Name student	Name signature	Subject
Email student	Contact person	FoS
Contact person	Reference	Company
Reference	(Adressee)	# of students
(Properties of the education)		Workload
		Attendant

The collection of documents spans several years, wherein three different heads of department were in place. In the first two cluster layers, these three different order forms formed one general MLC. A similar effect was noted in the admittance letters. The largest group of admittance letters is written in English, while the smallest group is written in Dutch.

For the metadata labels concerning the admission letters and order forms, full agreement was again reached between the two experts. Table 4.9 presents the identified metadata for these two document types.

Besides these identified labels, the two experts made for both of the document types a similar suggestion. For the admission letters, this suggestion concerns the properties of the education for which this student is admitted. In this context, the programme is known but for a general document model additional labels, such as the department of university organising the programme, could be interesting according to the experts. For the order forms, an additional label for the addressee could be interesting, but it was not added in the final list because it was not clear for the experts whether the names in the network were the addressees. Analysis of the documents supporting this document type indicated that these names were of the heads of department when the order forms were submitted.

The labeling for the thesis proposals resulted in a free-marginal Kappa score of 0.81. The small difference between the experts was the balance between administrative and research/educational based labels which seems to reflect the role of the experts in the day-to-day operation of the division. In Table 4.9 the final document model is presented. A considerable part of the document collection consists of draft or accepted papers in the different research domains of the group. Therefore it is not surprising that these types were identified, but an assignment of a general document type to a research domain is however difficult. No difference can be made between these papers and project proposals, mostly due to the limited presence of other research related documents such as proposals and project reports. Furthermore the research topics are not always explicitly mentioned, and therefore these terms are not considered as important terms in

the term network. Related to this phenomenon, the cross domain activities of some researchers complicate the identification process. Similar to the document types related to the letters, the names of the people are considered as important terms (see Appendix G.1.2). These terms do provide different possible values for a label. The document type grouping the logistic articles obtained almost no agreement, only one common label was identified: the subdomain of logistics. The researcher gave more technical labels such as model and process description; as well as performance and performance measurement. The secretary identified the labels 'time of document' and 'domain of publication'. This result again suggests the importance of the role of knowledge workers in the company, assisting in the identification process. Similar conclusions are valid for the document type grouping the maintenance and ecodesign articles.

Table 4.10 presents the obtained document models according to the conclusions of the domain experts.

Table 4.10: Overview of the document models for 'Maintenance articles' and 'Logistics articles'

Maintenance & Ecodesign	**Logistics**
Company	Company
Product	Model type
Process type	Domain
	Objective

The last two document models, admission request and thesis contracts, were labeled in full agreement and in a free-marginal Kappa score of 0.84, respectively. In Table 4.11 the resulting document models are shown.

The analysis of several smaller MLCs resulted in interesting results. To limit the test procedure, these results were not presented for a Kappa scoring process. The MLC number 2 in Figure 4.16 seems to suggest a document type of 'invitation to a departmental meeting' with a meeting agenda. The MLC number 171 indicates a collection of timetables, which is confirmed by the documents assigned

Table 4.11: Overview of the document models for 'Admission request' and 'Thesis contracts'

Admission requests	**Thesis contracts**
Institute of education	Company
Nationality	FoS
Diploma	Confidentiality
Annulation	Transportation
TOEFL	# of days in company

Table 4.12: Overview of terms for two small MLCs

MLC 2	MLC 171
'vergadering'	'mgmt'
'uitnodiging'	'aud'
'adviesraad'	'seminars'
'afdelingsvergadering'	'semester'
'uitgenodigd'	'meeting'
'goedkeuring'	'elective'
's02'	'courses'
'vriendelijk'	'timetable'
'uur'	'gelders'
	'workshop'
	'mim'
	'avd'
	'1st'
	'kulak'

to the MLC. Table 4.12 presents an overview of the important terms of these MLCs.

Several templates were expected to be identifiable in this document collection, and Figure 4.17 provides an overview of the results of a test concerning these templates. Because the mapping of the terms to properties is subjective, the following scoring process was maintained: for each of the document types the important terms to each of the properties were selected, and the presence of these terms was compared in the documents supporting the MLC. The selection of terms is given in Table G.1.2.1 in Appendix G.1.2. Filtering of the template defining terms in case of the research related MLC, for example, resulted in an assignment of these documents to research related documents. A graphical representation of this view threshold is shown in Figure 4.18.

As previously mentioned, for each of the above mentioned MLCs the term network was generated. As an example, the obtained conceptual term network for the multilayer cluster related to 'thesis proposals' is displayed in Figure 4.19. A thesis proposal document, as can be derived from its related template, has several properties such as title, company and expectations.

4.5.3 Quotes case

After the preprocessing phase, the Quotes dataset contains 5241 processable documents. Applying the SAME technique on this dataset, resulted in the MLCs shown in Figure 4.20. For the eight highest MLCs obtained after two iterations, the related term networks were presented to the experts. These networks and are

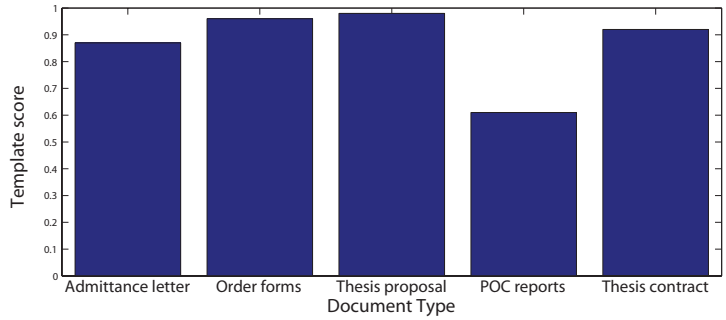


Figure 4.17: Template scores for MLCs from the research group dataset

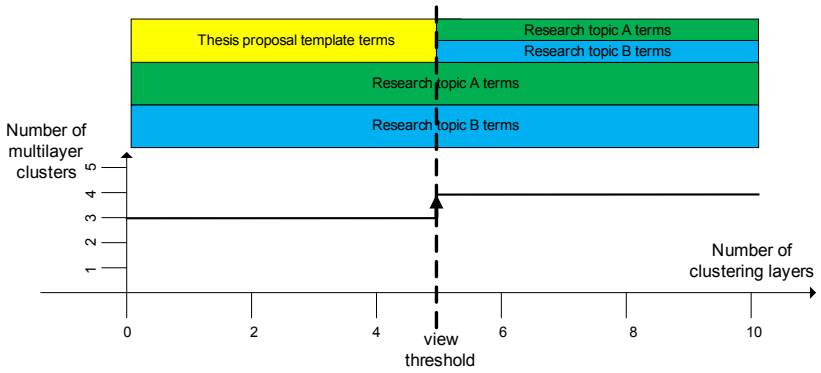


Figure 4.18: View threshold occurring in a multilayer cluster related to research

provided in Appendix G.1.3.

According to the raters, three general document types can be identified based on the presented networks:

- Sales quote (English/French/German)
- Quotation email
- (Autocad) drawing

The document types and the related support are also indicated in Figure 4.21.

These findings are confirmed by the manual identification results, as referenced in dataset description in Section 4.4.1. For every prospect a project or quote is compiled and for each quote, all conversations and drawings are grouped together. Every project thus at least contains

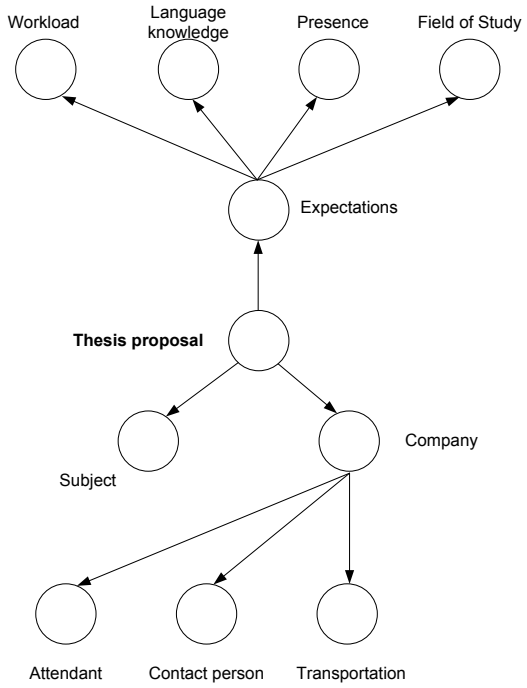
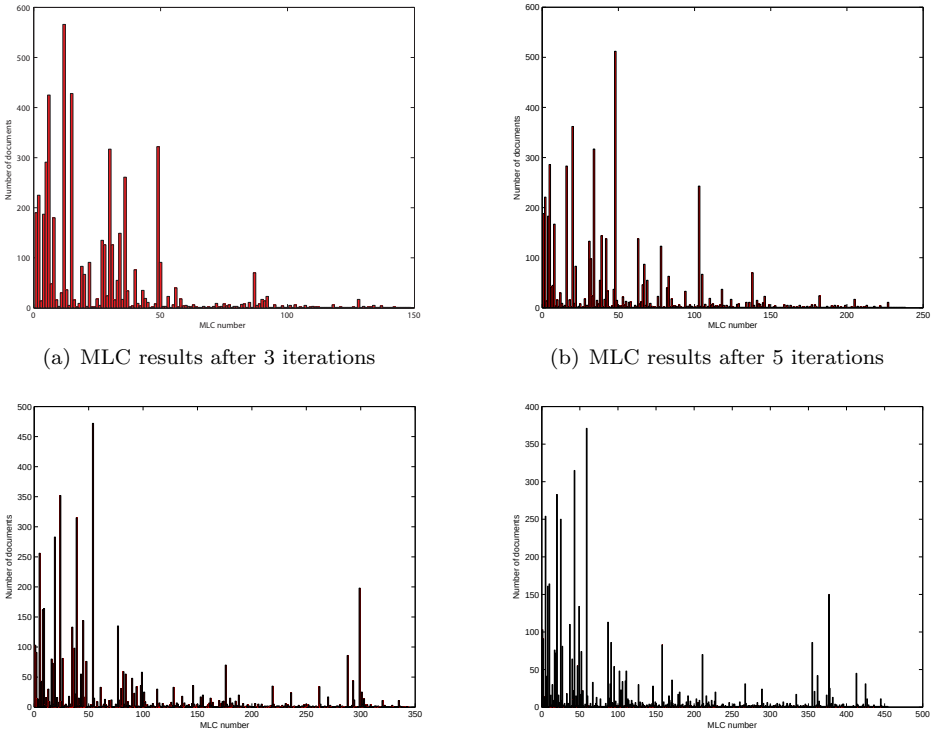


Figure 4.19: Conceptual term network of the multilayer cluster related to 'Thesis proposals'

- An email containing a technical problem statement (quotation tender)
- An email containing order overview
- One or more drawings in attachment in one or both emails

Based on an analysis of the document collection and feedback by the commercial partner, the major part of the dataset is identified as 'sales quote'. An estimation of the size of this fraction lies around 59%. This figure is influenced by the preprocessing and filtering phase and possible reformulations of quotes in a project.

According to the SAME technique, several MLCs were identified in the document collection. The largest identified MLC concerns technical drawings, as indicated by the experts and the composition of documents supporting this MLC (93 % drawings). As can be noticed, not all drawings were grouped in this MLC. The main reason for this effect is that the conversion of these drawings to a workable plain text format results in a small document. The non-included



(a) MLC results after 3 iterations

(b) MLC results after 5 iterations

(c) MLC results after 7 iterations

(d) MLC results after 9 iterations

Figure 4.20: MLC results for the Quotes dataset

drawings are filtered out or reside as singletons in separate MLCs. The important terms for this MLC are listed in Table 4.13.

Table 4.13: Important terms for the MLC related to 'Drawing'

mold	blow	neck
blank	scale	emhart
hard	sheet	data
cooling	edge	cavity
	qty	holes
	depth	pty
		decimal

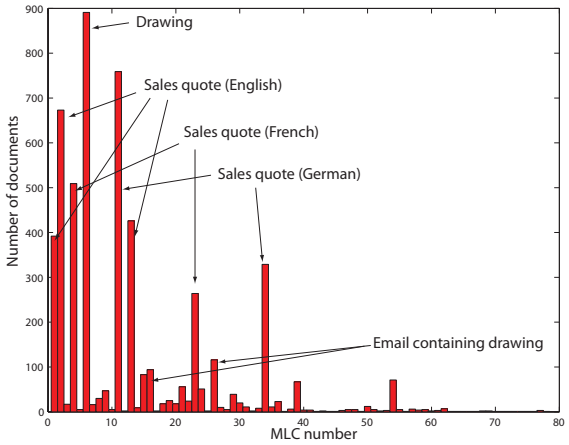


Figure 4.21: Identified document types for the Quotes dataset after 2 iterations

Analyzing six of the eight other term networks by the experts led to the conclusion that the different MLCs are generally grouped based on the language and the contact (sales) person of the company. For the largest MLC (Table 4.14), this contact person is called 'Andrea Gherzi'. For the second largest MLC, this person is called 'Peter Dean'. Other names that appear in the list of terms also often appear in other MLCs, and they are people from the production department. In the email conversations, the related email addresses are often included.

For the largest MLC concerning Sales quotes (MLC 2), the most important terms of the first three layers are shown in Table 4.14. The second largest related MLC only differs in the first iteration, where the name of 'Peter Dean' appears, the other two iterations are identical. Similar terms, either in German or French appear in multilayer cluster three and four. These terms are listed in Appendix G.1.3.

The last identified document type, which contains quotation emails, obtains a low support. The reason for this low support is the presence of the drawings: most quotation information is present in the drawing. The related email often contains very few words, only a reference to the drawing. Therefore, several of these quotation emails are located in MLCs related to sales quotes, due to the presence of company or person names.

In Table 4.15, the resulting documents models are shown as obtained by the analysis of the term networks. The Kappa scores for these labels are 1.00, 0.81 and 0.95 respectively. In this table, the results of one rater is shown as this rater has a profound knowledge of this dataset.

In the term network for quotation emails, two properties are predominantly present: the company and the file type 'pdf'. The terms such as 'sep' and 'nov'

Table 4.14: Selection of important terms appearing in the first three layers for a MLC related to the document type 'Quotes' of the Quotes dataset

Iteration 1	**Iteration 2**	**Iteration 3**
gherzi	base	sep
andrea	welding	acknowledgement
patriek	yioula	<company name>
quoting	olivier	pdf
boecquaert	rapoye	smtp
	glassworks	quote
	nicola	sales
	mazza	public
	material	received

Table 4.15: Overview of the identified document models for the Quotes dataset

Quotation email	**Sales quote**	**Drawing**
Company	Company	Scale
Company contact person	Company contact person	Dimensions
Date	<company> contact person	Quantity
Related drawing	Date	Construction process
	Related drawing	
	Contact properties	

indicate the relevance of a date indication. Less evident is the company contact person. Only the names appearing frequently enough are part of the network. For the sales quotes, the identification of the properties was based on the following reasoning:

- Company names and the names of contact persons are present.
- Similar to the network of quotation emails, some terms indicate the date.
- In the term network of the German and French document type, some terms indicate the contact properties of the company placing the order.
- The file '.pdf' indicates the necessary relationship between the sales quote and the drawing that was needed to generate the quote.

The last document model to complete, namely 'drawing', has a term network that is distinct from the other identified document types or multilayer clusters. From the term network, the scale of the drawing and the dimensions of the mould can be identified. The property 'Quantity' is denoted by 'Qty'. The property 'Construction process' is supported by the terms 'holes', 'cavity' and 'depth'.

4.6 Conclusions

In this chapter, the different components of the SAME technique were tested and validated. Document type identification, supported by the generated important term lists as described in Section 4.2, resulted in useful document types for all three datasets. The manual work, which is now solely based on interviews with experts within the company and preliminary tests based on term frequency, can be semi-automated in this manner. The test cases described in Section 4.5 give an indication of the possibilities of the concept of iterative clustering. Repeating a clustering exercise, while removing those terms important for the cluster structure, reveals a layered structure of the document collection.

As indicated in the previous sections, the presence of a domain expert having contextual knowledge facilitates the metadata identification process. In the previously described manual process, this presence is already a required condition to support the interpretation process. Providing sufficient terms and layers for each MLC is an important aspect in interpreting the results of the SAME technique, especially in the context of an unstructured environment where the inherent level of subjectivity increases. In the case of the Research group dataset, for example, the identified document types related to research domains less clearly indicate one or more document types. The identification tests indicated that the contextual knowledge therefore becomes increasingly important: the secretary is unaware about the detailed research tracks and thus could not identify a document type. While metadata generated by analysis of the first five layers might describe

research area A, the addition of the terms resulting from a sixth clustering layer may suggest terms of another research area B. This can possibly lead to a view threshold as discussed in Section 4.2.

The SAME technique offers indications for the construction of document models in a structured environment. The presence of a so-called general template for a certain document type enhances the performance of the SAME technique. A general template is the representative of a collection of similar templates. An example are the (letter) templates of companies A, B and C. Although they are not identical (i.e. different structure), they all share a similar content. The general template will thus be translated in a document type.

The application described in this chapter is a first step towards an enhanced automated metadata extraction system based on the content of a document. The currently used term network visualization method of the test tool still needs considerable enhancements to enable the deployment of the technique in an industrial setting. Points of attention are the location of the terms in the graph, and a graphical representation of the importance of a term. These issues will be discussed in the chapter concerning future research.

Algorithmic descriptions of the SAME techniques are provided in Appendix G.2.

Part II

Multi-vector Representation of Documents based on Topics

Chapter 5

State-of-the-Art in Multi-vector Representation of Documents Based on Topic Identification

Apart from the issues with clustering techniques in the context of an increasing number of documents, as discussed in Part I, other problems arise. Clustering techniques applied in a VSM regard a document as single piece of information. The retrieval process based on this VSM is however hampered when the documents contain or discuss multiple topics. A document vector in a VSM integrates all topics into one representation format which results in less accessible or irretrievable information. The ability to divide full text documents into their components based on coherent parts or topics can help to bypass this retrieval issue. In this way, each topic can be represented as a single vector in a VSM. Another application of such a partitioning of documents based on topic identification, is text extraction and summarization. These techniques reduce the size of documents and can be used as an input for further information processing in order to decrease the required amount of memory and processing time. The technique of topic-based segmentation has shown its usefulness for improved retrieval accuracy and retrieval of meaningful components of text, for document navigation and text summarization [Angheluta et al., 2002].

5.1 Multiple Vector Representations for a Document

The inspiration of proposing a manner to represent a document in multiple vector representations originates from daily life examples. Perceiving objects from multiple angles, for example, is an advantage to analyze them. Several research efforts can be identified in literature providing a solid base for this statement. Research in the domain of face recognition indicated that, if a person is related to more than one picture, the techniques usually obtain a better performance [Wu and Zhou, 2002]. Vertommen [Vertommen et al., 2008] constructed multiple-vector profiles to represent the tacit knowledge of a knowledge worker. This profiling system outperforms a single vector representation format. The retrieval performance of meta-search, which combines the search results from many different search engines, is considerably better compared to the performance of the result obtained from one single search engine [Gulli and Signorini, 2005]. These observations have motivated the research of a multi-vector representation for a document, discussed in detail in Chapter 6.

To conclude this introduction, the awareness of the need to represent a document in different perspectives has already been mentioned in literature. It is generally understood that in information retrieval the retrieval performance normally increases with the size of the stored documents. However in literature recent research activities can be identified indicating the existence of an upper limit on the size of these documents, and approaches to deal with this limitation [Chen et al., 2006, Kulp and Kontostathis, 2007]. From this viewpoint, the research presented in this chapter and Chapter 6 can be considered as facilitating the required preprocessing step for documents in a KMS in order to accurately preserve the information of a document collection.

In the following Sections 5.1.1 and 5.1.2 a brief overview is presented of the state-of-the-art techniques for a multiple-vector representation of a document, based on structural or content properties of this document. However, no clear difference exists between these two approaches. An example is a set of part-of-speech tags: selecting such a set can be based on structure as well as on its content.

Because the research presented in Chapter 6 is based on a novel topic identification algorithm, Section 5.2 discusses the current state-of-the-art in the domain.

5.1.1 Structural information based approaches

Constructing document representations originating from structural information is an approach that is often applied for reasons of computational and retrieval performance [Larkey, 1999, Verhaegen et al., 2009]. The representation of a

document is often more based on the structure of the document than on its content. By including structural information of a document, often an inherent dimensionality or document length reduction step is performed. The rationale behind this approach is that the author imposed a notion of importance by providing a structure in the document, and therefore it should be included in the further processing.

The pure structure of a document is the most straightforward approach. Examples are the representation of a document by its title, varying font sizes or meta tags [Collins-thompson et al., 2002]. Other approaches exist that specifically depend on the markup language or approach when the document was created. Examples of these approaches are specific HTML or XML tags [Zhang et al., 2002].

All representation formats need to be converted to a usable and common format, e.g. a vector representation in a VSM. Ogilvie [Ogilvie and Callan, 2003] presented a manner to combine multiple representations in the context of meta-search techniques. He proposed a mixture language model [Ponte and Croft, 1998] that combines different structural document representations to perform the retrieval process.

Other research activities can be identified combining different types of structural information. Citations [Glenisson et al., 2005] and other bibliometrics are used to represent documents. For web pages the same reasoning counts for links and related structural information on the web. Phrases, proper nouns and passages are also often used as an input for a combined representation format [Croft, 2000].

5.1.2 Content based approaches

In literature two approaches can be identified that try to connect a document to a multi-vector representation for information retrieval purposes.

Chen [Chen et al., 2006] devised a 'stereo' document representation for a document. The technique is based on the Differential Latent Semantic Indexing (DLSI) projection technique, constructing differential inter- and extra-document vectors for a given set of training documents. This variant of the LSI technique, as discussed in Section 2.2.3.3, uses these vectors to capture the unique characteristic of the individual documents. LSI ignores the relevant information captured in the distance between the original and the reduced term space. By means of these two vector types, two subspaces can be obtained: the extra-DLSI and intra-DLSI spaces. The authors claim that, besides the improved adaptability to the unique characteristics of a document, the application of DLSI leads to an improved retrieval performance in search engines. To obtain a multiple vector representation format of a document, a document needs to be partitioned in several so-called subfiles. An example of such a subfile is (a part of) an abstract. This partitioning is required prior to the start of the construction process, and no fixed definition is given on how to create these subfiles. An example is described in which

every subfile consists of a predefined set of sequential sentences, and a predefined overlap rate for the sentences is given. For every of these subfiles, a term vector is created in a similar manner as described in Section 2.2.2.1, and their related inter- and extra-document vectors are projected into the DLSI space. No further processing is performed on these vectors to obtain an enhanced multiple vector representation.

Creating a multidimensional representation of a document based on user's Information Need (IN) is another approach [Piwowarski et al., 2010]. In the presented framework a document is associated with a set of V (pure IN) vectors spanning the IN space. The relevance of a document to a user's information need is based on the projection of the document in one or more IN dimensions, related to the user's query. In the presented validation methodology the term space is taken as the pure IN space. Any combination of terms, e.g. sentences, can be taken as IN space, but no approach is given on how to create this space.

5.2 Topic Identification Algorithms

A vast collection of definitions for the notion 'topic' can be found in literature [Andrews, 1985]. In this dissertation, a topic is considered as a set of words from the document (part) that describes the content. The algorithms discussed in this section are aimed at identifying one or more topics in the content of a document. This is opposed to other interpretations of this notion, such as topic identification in a complete document collection.

Due to the reasons previously described, several innovative research efforts can be identified for text segmentation. Classifying this collection is a difficult task: roughly they can be separated in two main categories: techniques respectively using statistical information extraction techniques and those exploiting lexical cohesion. This classification is however not strict. Several statistical methods adopt a format of lexical cohesion. In this section difference is made between statistical techniques exploiting this lexical distribution and purely linguistic techniques.

Many topic identification algorithms assume that topically coherent subdocuments are related to text fragments exhibiting a homogeneous lexical distribution (i.e. the usage of words). In literature, several approaches can be identified, based on different descriptions of this distribution in a document. Lexical weighting is one of the most popular approaches in topic identification [Morris and Hirst, 1991, Hearst, 1997, Kathleen and Mckeown, 2003, Sitbon and Bellot, 2007]. Lexical chains and the extended approach, the so-called weighted lexical links, are two techniques often used in a huge collection of identification algorithms. A lexical chain is defined as a relation between repetitions of a lemma if the gap between two repetitions (measured as the number of text fragments in between) is below a certain distance,

also called hiatus [Sitbon and Bellot, 2007]. Using this definition, a lemma can create multiple chains, depending on the inner distance of the chain.

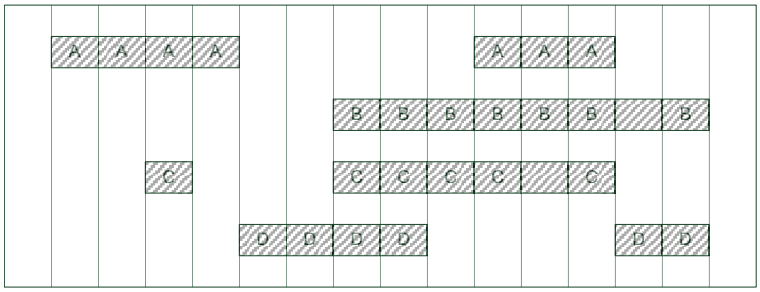


Figure 5.1: Schematic example of four lexical chains, according to the general definition of a lexical chain

Figure 5.1 illustrates the result of the identification process of lexical chains. Four chains are identified, based on the lemmas A, B, C and D. The hiatus is 3, therefore A, C and D are each defining two chains. The classical view on lexical chains has a binary nature, whether a chain is present in a text fragment or not. Lexical chains inhibit more information than this binary view. The nearer the repetitions of a lemma, the more significant this lemma is for the topic [Morris and Hirst, 1991]. To capture this notion of information, weighting is included in evaluating the importance of a lexical chain. The variant of the weighted lexical chains by omitting the hiatus is called a weighted lexical link. If this approach is applied to Figure 5.1, only four links, with varying weights, are identified.

Using these lexical chains a similarity measure between document fragments can be defined to measure the topic connectivity of the document fragments. Most techniques based on this approach are linear topic segmentation algorithms. These algorithms place boundaries inside a text at positions where a topic shift is identified. This identification process is performed in a (fixed size) sliding window, examining lexical variations. The lexical variation often results in a drop of an employed similarity measure. As previously indicated, many algorithms can be described on this generic description. Popular examples are TextTiling [Hearst, 1997], C99 [Choi, 2000], Dotplotting [Reynar, 1999] or Segmenter [yen Kan et al., 1998]. The TextTiling technique segments texts into multiple entities (i.e. sequences of 3 to 5 sentences) or subtopics, the so-called 'tiles' using the cosine similarity between segments. A smoothed curve is calculated expressing the similarity between adjacent entities. Minima in this curve are considered as potential topic boundaries.

Other statistical approaches exist using global information of the text. Malioutov [Malioutov and Barzilay, 2006] presents a graph-theoretic framework.

The text is converted into a weighted undirected graph in which the nodes represent the sentences and the edges quantify the relations. The text segmentation is performed by applying the normalized-cut criterion [Shi and Malik, 1997]. By using this criterion, the similarity within each partition is maximized and the dissimilarity across the partitions is minimized. The graph-based approach extends the local cohesion range of the sliding window by taking into account the long-range lexical cohesion and distribution in a text. The computational techniques for finding the optimal solution to the minimal cut objective are however difficult. The minimization of the normalized cut is NP-complete, but, due to the linearity constraint of this segmentation type, obtaining an exact solution is feasible [Malioutov and Barzilay, 2006].

All of the described algorithms rely on statistical properties of the text. On the other hand, linguistic methods introduce a set of specific rules based on the corpus and use external semantic information such as thesauri and ontologies. This is the main drawback of this type of identification techniques: the results are dependent on the semantic resources available for a specific text [Utiyama and Isahara, 2001] and therefore the setup is limited to the text. Hidden Markov Models and Neural Networks are used as part of the learning process in the technique of Amini [reza Amini et al., 2000]. A probabilistic sequence framework is proposed to estimate symbol or term sequences in a text. This framework should enable processing of more complex information retrieval and extraction tasks. Caillet proposed a machine learning technique based on term clustering [Caillet et al., 2004]. The technique first discovers the so-called different concepts in a text, which are defined as sets of representative terms. The partitioning in coherent paragraphs is performed with the Maximum Likelihood clustering approach, as discussed in Section 2.4.3. Linguistic approaches are known to be efficient, but they depend on the content of the document for the learning process. The approach presented in Chapter 6 is a statistical based technique, because statistical methods appear to obtain the best results on general corpora [Lamprier et al., 2008]. A comprehensive overview of several, statistical and linguistic, topic identification techniques can be found in [Chali, 2005].

5.3 Discussion

The objective of the second major research track was the construction of a multiple vector representation of a document, based on the content of the document. Each vector should represent a topic describing part of the content of the document. The number of vectors should be dynamic, and this number should be defined automatically.

A considerable number of topic identification techniques are identified in literature, but none of the existing content based techniques deals with the following scenario: the possibility that one or more topics are discussed in a

non-contiguous manner. A topic can be discussed in multiple sections of the text, while other topics appear in between these occurrences.

A last desirable property is the ability to deal with multiple languages. Although the capabilities of the existing techniques in a multi-lingual environment are not explicitly mentioned in literature, the rigidity and required manual efforts will not facilitate the process in such an environment.

From the presented overview of the techniques for vector representations and the topic identification, no technique can be identified linking a dynamic representation format to the content of a document. The existing representation formats are either based on structural information, or rigidly based on the content and require considerable manual effort (e.g. creating the IN space). From this list of desired properties and the observed deficiencies of the methods and techniques document in literature, the research for a multiple vector representation format for a document, based on the topics present in its content, was initiated and is presented in the next chapter.

Chapter 6

Multi-vector Representation for Documents

6.1 Introduction

The research introduced in the following sections is focused on a novel technique that automatically subdivides a document in multiple topic-based components based on statistical and spectral properties of the text and the related coherence graph. The underlying idea for this segmentation technique is common to the techniques proposed by Hearst and Choi, as discussed in Section 5.2: when a topic is described in a text, one can assume that a specific set of terms is used in the text fragment describing this topic. When a topic shift occurs, this set is substantially changed. Therefore re-occurrences of these terms indicate the presence of a certain topic, and topic boundaries can be identified. The main objective of topic identification methods is to label groups of document entities with common properties. This means not only identifying the topic shift between units, but also to combine those units describing a similar content.

The difference between the notions of 'topic' and 'subtopic', which are frequently used in the remainder of this chapter, is difficult to the describe. In this chapter, the interpretation of Hearst is used [Hearst, 1997]: subtopic discussions are assumed to occur within the scope of one or more overarching main topics, which span the length of the text.

The presented approach is able to discover non-contiguous connections, which differs from the previously discussed linear text segmentation in two aspects which are explained in the following paragraphs.

Several of these techniques are applied to only identify (sub)topic boundaries between two adjacent document entities. No recombination of segments containing similar content is thus performed. The presented technique uses a novel

combination of a new coherence graph and spectral partitioning techniques to reach this objective.

The second difference between the presented technique and existing techniques is shown when these techniques are applied to the 21-paragraph Stargazers document [Hearst, 1997], a well-known example in the context of topic identification. Subtopic frontiers were identified in this corpus by human judges in order to evaluate the TextTiling technique of Hearst. This subtopic structure is indicated below, together with the paragraph ranges:

- $1 \rightarrow 3$: Intro - the search for life in space
- $4 \rightarrow 5$: The moon's chemical composition
- $6 \rightarrow 8$: How early earth-moon proximity shaped the moon
- $9 \rightarrow 2$: How the moon helped life evolve on earth
- 13: Improbability of the earth-moon system
- $14 \rightarrow 16$: Binary/trinary star systems make life unlikely
- $17 \rightarrow 18$: The low probability of nonbinary/trinary systems
- $19 \rightarrow 20$: Properties of earth's sun that facilitate life
- 21: Summary

While subtopic algorithms try to identify these subtopic frontiers, the proposed technique tries to identify the general topics present in a document taking the overall content into account. Considering the Stargazers document, the presented technique identifies two general topics in this document: the first topic describes the earth-moon interaction (paragraphs 1 - 13) while the second topic concerns the binary/trinary star systems, covering paragraphs 13 to 21.

In Section 6.2 the topic identification process is explained in more detail, followed by the Sections 6.3 and 6.4 discussing the validation method and results. Section 6.5 is based on the identification process, and discusses the conversion of the topics into a multi-vector representation for a document. A technique to identify languages on sentence level is introduced in Section 6.6. The conclusions of this chapter are drawn in Section 6.7.

6.2 Topic Identification Process

Besides the preprocessing stage converting documents into a processable format (see Section 2.2.3), the proposed component identification process is composed of the following parts:

1. Construction of a coherence graph based on a linkage matrix
2. Construction of the normalized symmetric Laplacian of the coherence graph
3. Identification of the number of topics using the calculated Laplacian
4. Spectral partitioning of the coherence graph to obtain the coherent components

The last phase in this process, the partitioning of the coherence graph to obtain these components, is also a combination of two steps:

1. Reducing the feature space of the original document to a reduced subspace (see section 6.2.4)
2. Application of a cluster algorithm in this reduced space to obtain the components

In the following paragraphs, these different phases of the identification process are discussed in more detail.

6.2.1 Coherence graph construction based on a linkage matrix

The main idea of this document decomposition process is based on the presence of lexical chains in a document. A lexical chain is defined as a chain of document entities based on the re-occurrence of a certain informative stem or term in a document [Barzilay and Elhadad, 1997, Morris and Hirst, 1991]. An informative term is a term that is neither a stopword nor a numerical value.

The presence of these chains represents the lexical cohesive structure of the document [Morris and Hirst, 1991]. The term and the identifiers of the document entities are stored as information of the chain. An example of such a lexical chain is given in Figure 6.1. In this figure, the sentences are taken as document entities. Their identifiers are shown between brackets. Several lexical chains are linking the sentences of a BBC article. The occurrences of the term 'copper' link the sentences 1,2 and 4; while the occurrences of term 'Honda' link the sentences 7, 8 and 10 (as graphically shown in Figure 6.1 by the connecting lines). These two lexical chains are only a subset of several possible lexical chains in this small example. Considering all chains present in this example, two larger components can be identified. The first component is the set of sentences from 1 to 6, the second component contains the set of sentences from 7 to 11. In the figure, these two components are separated by a horizontal line.

As already indicated in Section 5.2, the relations between the document entities described by a lexical chain are not evenly informative. The first difference is based on the notion of distance. The notion of distance between document entities in this

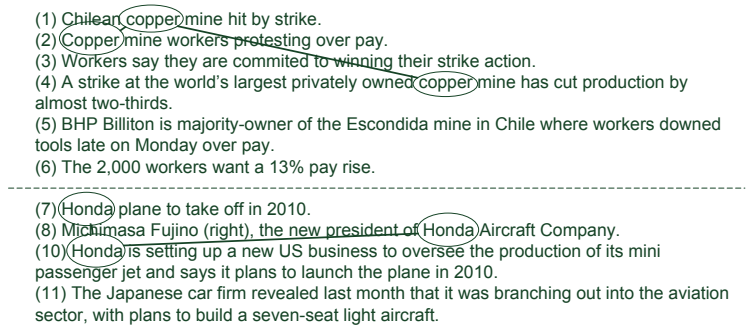


Figure 6.1: Examples of two lexical chains. These lexical chain are based on the terms 'copper' and 'Honda', respectively.

context is defined as the number of entities they span. In Figure 6.1 the distance between the first and third occurrence of 'copper' is 4 (sentence 1 to 4). Intuitively, the nearer the related document entities are, the more chance there is that they are related to a similar content. However, it remains possible that another distant set of entities is related with the same term. These entities are also describing a similar content, and both groups will form one longer chain. However, due to the distance, not all pairwise relationships in this chain will be equally strong.

A second difference between the entities is the informative value of the term itself. Not all words in a document are equally important [Newman, 2005]. An obvious example are stopwords. These carry almost no content, and therefore can be considered as noise when creating stem chains.

Longer chains are also preferred over shorter ones. This reasoning is based on the importance of mid-frequency words originating from the Zipf-curve (see Section 3.3). Considering the possibility that a term occurs multiple times within a document entity, the longest chains are most likely defined by the most frequently occurring terms in the document. This requires an optimal filtering of non-informative terms. Shorter chains can be created based e.g. on typing mistakes or the presence of numerical values.

The previous assumptions are translated into a quantification formula. The linkage $Link_{i,j,t}$ between document entities i and j based on term chain TC_t, formed by term t, is defined as

$$Link_{i,j,t} = w_t * |TC_t| * \frac{dl_{entities}}{1 + (C_i - C_j)} \tag{6.1}$$

w_t is the weight of the term defining the chain C in the document, such as raw or normalized frequency [Jain et al., 1999]. $|TC_t|$ is the number of occurrences in the term chain C, $dl_{entities}$ is the length of the document expressed in number

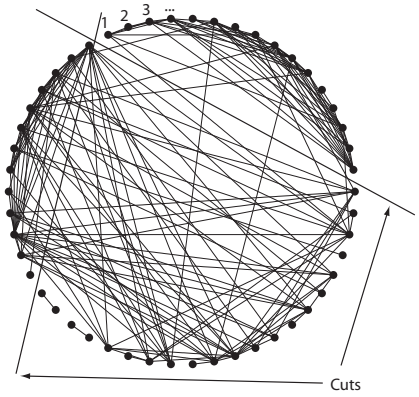


Figure 6.2: Coherence graph of a document containing 50 document entities.

of document entities, and $TC_{t,i} - TC_{t,j}$ is the distance between entities i and j in the text. In a complete document, multiple term chains can be identified and quantified. The overall linkage $l_{i,j}$ between document entities i and j is based on the aggregation of all linkage values for the term chains that connect both entities:

$$l_{i,j} = \sum TC_{t\,t=1}^{m} Link_{i,j,t}$$

The linkage results of all document entities finally results in a square linkage matrix W.

Based on this matrix, a graph structure can be composed to represent the quantified relationships between the document entities. This undirected, weighted coherence graph is defined as

$$G = (N, E) \tag{6.2}$$

comprising a set N of nodes, representing the document entities, together with a set E of edges representing the linkage values. This means that each edge between two nodes N_i and N_j carries a non-negative weight $l_{i,j}$.

An example of a graph structure is given in Figure 6.2. The dots in circular form are the different nodes representing the document entities. The lines represent the coherence between the related pair of document entities. The weight values are omitted for reasons of clarity. The final graph partitioning result is also indicated in this figure. After applying this process, which is explained in the next sections, three topics can be identified. The cuts are indicated with a full line.

6.2.2 Graph Laplacian construction

To obtain a partitioning of a graph, spectral graph partitioning can be employed [von Luxburg, 2007]. Laplacian matrices are the main tool for spectral graph partitioning. With every graph, such a Laplacian matrix can be constructed. In literature multiple definitions of a Laplacian matrix can be found. For reasons of stability and consistency, the symmetric normalized Laplacian matrix is frequently used [von Luxburg, 2007]. This matrix is defined as

$$L = I - B^{-\frac{1}{2}} \cdot A \cdot B^{-\frac{1}{2}} \tag{6.3}$$

A and B respectively denote the adjacency and degree matrix of the graph. I represents the identity matrix. In this context, the following definitions for the adjacency A and degree matrix B are used:

Definition 5 *An adjacency matrix of a weighted, undirected graph is a matrix restricted to the following conditions:*

$$A = \begin{cases} A_{i,j} & \text{if } i \text{ and } j \text{ are connected} \\ 0 & \text{otherwise} \end{cases}$$

Definition 6 *A degree matrix of a weighted, undirected graph is a diagonal matrix restricted to the following conditions:*

$$B = \begin{cases} B_{i,j} = \sum_{a=1}^{n} B_{i,a} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

6.2.3 Number of topics identification

The identification process of the number of topics is summarized in this section. The second largest eigenvalue of the linkage matrix explains a significant amount of variance or structure present in the graph [von Luxburg, 2007]. If the linkage matrix is sorted in the order proposed by this eigenvector, the reordered matrix contains dense segments along the diagonal indicating coherent subgraphs in the coherence graph. The ideal situation is that the segments have a full square form. These squares indicate the presence of complete subgraphs.

The number of components can be determined by identifying the relevant dense segments along the diagonal. In an optimal configuration (i.e. every component has a single relationship with another component), these blocks are dense squares, as indicated in the following matrix:

$$L = \begin{bmatrix} L_1 & 0 & 0 \\ 0 & L_2 & 0 \\ 0 & 0 & L_3 \end{bmatrix}$$

with L_i a full dense matrix.

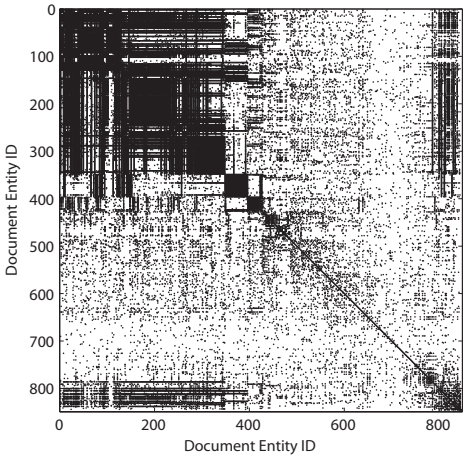


Figure 6.3: Matrix representation of the coherence graph

An example of this sorting is shown in Figure 6.3. The dots in this figure indicate existing relationships between the document entities.

For the further identification process, the notion of a square segment along the diagonal is introduced. This shape is defined to have its upper left and lower right corner on the diagonal, and its size is defined as the number of document entities it spans.

The identification process is composed of two steps. The first step is to extract a number of square segments that possibly indicate the coherent components. The second step is to retain those segments that cover a substantial number of relationships.

In the first step, Algorithm 1, documented hereafter, is used to retrieve dense segments. For every row, the sizes of possible segments are identified. The size of a segment is defined by seg_{size}, the number of adjacent non-zero values in a row of the matrix representation. A threshold value s stipulates how many adjacent zero values, indicated as blank spaces in Figure 6.3, can be taken into account when delineating the segment.

The result of this first step in the process is a list of candidate square segments. The segments can overlap with, or be included in, other segments. The number of segments involved in every non-overlapping list, is a possible result of the number of components problem.

The second step in the identification process uses the notion of coverage ratio

Input: Linkage matrix W sorted in the order of the second largest eigenvector
while $seg_{size} <$ *threshold s # of non-adjacent relations* **do**
  foreach *Document entity i* **do**
    j=i; **if** $W(i,j) > 0$ **then**
      $count++; j++$
    end
    else
      THR–;
    end
    if $s = 0$ **then**
      exit;
    end
    else
      i++;
    end
  end
  foreach *Document entity i* **do**
    Create a square segment with *count* as size, and upper left corner positioned at point i
  end
  foreach *Document entity i* **do**
    Calculate the density of the square segment. The density is defined as the number of related document entities over the total possible number of relationships. If the density of this square segment is above a predefined threshold, retain this square.
  end
end
Output: Enumeration of square segments

Algorithm 1: The Square Segment Identification (SSI) algorithm

of a square segment. This ratio R_i of square segment i is defined as

$$R_i = \frac{S_{i,incl}}{S_{i,incl} + S_{i,excl}} \tag{6.4}$$

$S_{i,incl}$ is the number of relationships that are included in the identified square segment. $S_{i,excl}$ is the number of relationships that are excluded from the square. In Figure 6.4 the coverage percentage of the indicated segment is 0.667. Due to the symmetry of the linkage matrix, the analysis of half the matrix is sufficient. The upper triangle of the square segment, starting at element 2 with a size of 3, contains 6 non-zero values. The total number of non-zero values covered by the rows of the square segment is nine, indicated in Figure 6.4 as hatched cells. For

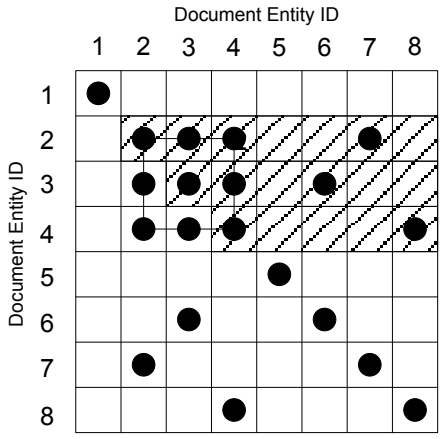


Figure 6.4: Example of the coverage ratio calculation. The ratio for the hatched cells in this example is 0.667.

every identified square segment, this coverage percentage can be calculated. As a result, for every non-overlapping list of square segments, the total coverage ratio can be calculated as the average sum of the coverage of these segments. This score is situated in the range $]0, 1]$. The higher this fraction, the higher the chance is that the corresponding number of components will represent the actual number that can be distinguished in the document. The reasoning is that the larger the group of entities that is covered by a non-overlapping list of square segments, the better this number of square segments will indicate the coherent structure present in the document. These square segments are defined by the variance explained by the second largest eigenvector, and therefore are clustered together based on common properties. A value of one for the coverage indicates a full coverage of the entities by the related number of square segments, a near 0-result indicates the opposite. In Figure 6.3, a number of square segments can be identified. The SSI-algorithm returns 4 as the most likely number of components, followed by 6, 2 and 3. Less likely are 16 and 19.

6.2.4 Spectral partitioning of the graph

The actual partitioning of the coherence graph is based on the 'minimal cut approach'. This approach results in the partitioning of the graph by removing those edges that have the lowest sum of coherence quantification in order to obtain the previously obtained number of subgraphs. Given the graph $G = (N, E)$, minimising a cut of a graph is mathematically similar to the identification of a

non-trivial vector x that minimizes the following function

$$f(x) = \sum_{(i,j)\in N} edge_{i,j}(x_i - x_j) \tag{6.5}$$

In this objective function, x is a bi-partition (P, Q)-vector with $x_i = 1$ if $i \in P$ and $x_j = -1$ if $j \in Q$. $edge_{i,j} \in E$ is the weight of the relation between vertices i and j. This objective function can be reformulated as

$$f(x) = x^T \cdot L \cdot x \tag{6.6}$$

with x the bi-partition vector and L the Laplacian matrix. This formula describes the relationship between the minimal cut of the graph and its spectral properties. The solution of Equation 6.6 is the so-called Fiedler vector. This vector is the eigenvector related to the second-smallest eigenvalue of the Laplacian matrix to the algebraic connectivity of a graph. This eigenvalue is also called the 'algebraic connectivity' of the graph, and is greater than 0 if and only if the graph is connected. A graph is called connected if every pair of distinct vertices in the graph can be connected through some sequential path of vertices.

The partitioning process is a two-phase process. Starting from the representation of the Laplacian, the spectral or Jordan normalized decomposition can be calculated. Selecting the k eigenvectors related to the largest eigenvalues of the Laplacian matrix, with k equal to the identified number of components, a k-dimensional representation in an inner product space is obtained [Skillicorn, 2007]. The mapping of the original data into a new metric space expresses the alignment or coherence of the graph, rather than its structure based on the original similarity.

The second phase is the actual clustering step. Any clustering algorithm can be employed to obtain a partitioning of the coherence graph. In this research, the algorithm of Ng, Jordan and Weiss [Ng et al., 2001] is applied to partition the coherence graph in multiple subgraphs. An overview of this process is given in Algorithm 2. The resulting subgraphs can be converted into the coherent components by reordering the document entities.

6.3 Validation

The validation process is composed of two parts, in accordance with the two phases in the overall identification process:

- validation of the technique concerning the number of identified topics
- validation of the quality of the identified topics

Input: Linkage matrix $L \in \mathbb{R}^{n \times n}$ and k the identified number of components

1. Construct the coherence graph described in section 6.2.1, and its related adjacency and degree matrices A and D
2. Compute the normalized symmetric Laplacian L
3. Compute the first k eigenvectors $\Lambda_1, \ldots, \Lambda_k$ of L
4. Create $\Lambda \in \mathbb{R}^{n \times k}$ containing the vector $\Lambda_1, \ldots, \Lambda_k$ as its columns
5. Normalize the rows of matrix Λ with the 1-norm
6. For each component i, the reduced feature space is the i-th row of the Λ matrix
7. Deploy a clustering algorithm in the new coordinate space

Output: Subgraphs containing the document entity identifiers of the coherent components

Algorithm 2: Graph partitioning process based on its spectral properties

The first test environments aim to validate the novel technique of identifying the number of components, as well as to compare the retrieval performance of the extracted components considering the known structure of the documents. The second validation part compares the quality of the presented technique to two well-known subtopic identification techniques:

- Choi (C99)[Choi, 2000]
- TextTiling [Hearst, 1997]

These techniques were chosen based on their consistent performance under various circumstances [Dias et al., 2007, Stokes, 2003].

This second validation step aims to stress the resemblances and the differences between the topic and the selected subtopic identification techniques.

6.3.1 Validation datasets

For the experimental validation of these techniques, the following datasets were used to create the required test environments:

- Reuters RCV1 document dataset
- OHSUMED medline dataset

Properties of these datasets are listed in Appendix A.

6.3.1.1 Reuters

Six categories were selected from the Reuters RCV1 dataset with the following labels:

- GCRIM: Craw, law enforcement
- GDEF: Defense
- GDIP: International relations
- GDIS: Disasters and accidents
- GENT: Arts, culture, entertainment
- GSPO: Sports

This selection is based on the size and the number of documents present in each category.

6.3.1.2 Ohsumed

From the OHSUMED test collection, only the documents were withheld rated as definitely relevant with one of the standardized queries, as stated in the OHSUMED descriptions. This resulted in a document collection containing 101 topics (five queries returned no documents), with in total 1985 documents. All the documents in this dataset are truncated to a maximum of 250 words.

6.3.2 Validation experimental setup

Two test scenarios were constructed using the previously described standardized datasets OHSUMED and Reuters resulting in four experimental setups. Both scenarios create artificial documents by concatenating original documents. The goal of these scenarios is to identify the topics present in the artificial documents.

The first test scenario is the sequential adding of a random number of randomly selected original documents. All document entities of the selected original documents are concatenated to form the artificial documents. This idea is shown in Figure 6.5 as the left object. The document entities in this test environment are paragraphs delimited by blank lines in the original document. Each paragraph is limited to ten sentences, so larger paragraphs were split. Every newly constructed document contains a random number of categories of the applied dataset. The number of selected documents varies between two to ten documents. Duplicates and multiple documents from the same directory are allowed. For each standardized dataset, a collection of these artificial documents is created.

The second test scenario differs in the order in which the document entities are concatenated. In this test scenario the document entities are concatenated

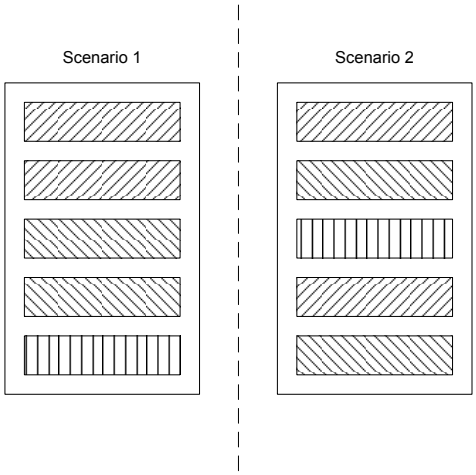


Figure 6.5: Schematic representation of the two different test scenarios. Similar hatching indicates document entities derived from a single document.

in a random manner to effectively distribute the different topics present in the document. In Figure 6.5, this idea is shown as the right object. The other conditions are also valid for this test scenario.

These two test scenarios resulted in four different test sets, two for each standardized dataset. Each generated dataset contained 1000 artificial documents. The results of the topic identification techniques on these test sets are discussed in Section 6.4.

The following steps were automatically performed on each document to obtain a suitable input format:

- punctuations and other marking symbols were removed
- stopword removal

6.3.3 Validation measure

Comparing two types of topic identification techniques, identifying topics at different levels of refinement can pose a problem. Since the focus of the proposed techniques is a reordering the document entities according to their spectral properties, an adapted F-measure based validation is used. The notion of topic in this validation measure is related to the category the document entity originates from. To obtain this measure the following scenario is used:

- Find the largest group of document entities on the same topic within a document

- Label this group of entities as the group entities covering that topic
- Mark this group as labeled, and the topic label as assigned
- Repeat with the remaining groups and topics that are left
- Check that every group is covering a different topic. If not, the group containing the smaller group of document entities on the same topic is set to be relabeled to a different topic.
- Repeat until all groups are covering a different topic.

Based on this process, a F-measure value can be obtained for every processed artificial document [Chen et al., 2004].

6.4 Results

In this section the results are presented of the three topic identification techniques applied on the four test environments. In the overview of the results, the proposed topic identification algorithm is abbreviated to TID (acronym of Topic IDentification).

For each artificial document, the initial composition or topics are known. Therefore comparisons can be made based on the identification of the topic boundaries. These boundaries were obtained from the three topic identification techniques, thus indicating the functional differences between the two existing techniques, and the newly proposed topic identification technique. This difference is situated on the level of detail on which the presented techniques identifies topics, and the ability to identify non-continuous topics. The clustering technique used in the second phase of the topic identification process was the unsupervised hierarchical agglomerative clustering method using average linkage. For every test environment the results are summarized as boxplots (Figures 6.6 ff.) [Tukey, 1977] in which the five represented numerical values are defined as:

- Smallest observation
- First quartile Q1 (25 %)
- Median Q2
- Third quartile Q 3(75 %)
- Largest observation

Observations that are identified as outliers are also individually indicated with a '+'.

6.4.1 Sequential test environments

Figure 6.6 presents the boxplot of the obtained F-measures for the first Ohsumed test set. The indicator values determining this boxplot are given in Table 6.1. A difference in average of 8 % can be noted between the proposed technique and the technique of Choi. The range of obtained F-measures for the proposed technique is significantly smaller compared to the Choi and TextTiling techniques. However, the positions of the boxplots of Choi and the proposed technique are positioned similar along the range of F-measures, indicating a slightly better performance for the proposed technique. This indicates that the identification techniques of Choi and the proposed techniques do not differ significantly for this sequential test environment.

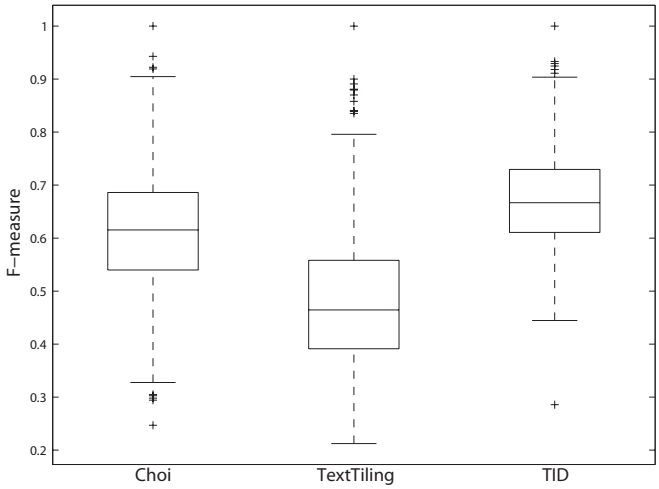


Figure 6.6: Boxplot of the sequential Ohsumed test environment

Table 6.1: Quality results of the sequential Ohsumed test environment

Dataset	**Choi**	**TextTiling**	**TID**
First quartile	0.5398	0.3913	0.6106
Median	0.6154	0.4644	0.6667
Third quartile	0.6892	0.5581	0.7296
Average	0.6122	0.4766	0.6774

The results for the second sequential test environment are summarized in Figure 6.7. The related figures determining this boxplot are given in Table 6.2. Similar conclusions can be drawn as in the first sequential test environment

Table 6.2: Quality results of the sequential test set

Dataset	**Choi**	**TextTiling**	**TID**
First quartile	0.3218	0.4152	0.4672
Median	0.4096	0.4806	0.5217
Third quartile	0.5275	0.5396	0.5817
Average	0.4327	0.4784	0.5310

considering the position of the boxplot of the proposed technique compared to TextTiling and Choi. No significant difference in position of the boxplot is notable between the three techniques. The differences in averages are considerably larger: the average of the proposed technique is 22,71 % higher than Choi and 10.9 % than TextTiling. Also the range of the boxplot of the proposed technique is the smallest of all three obtained boxplots.

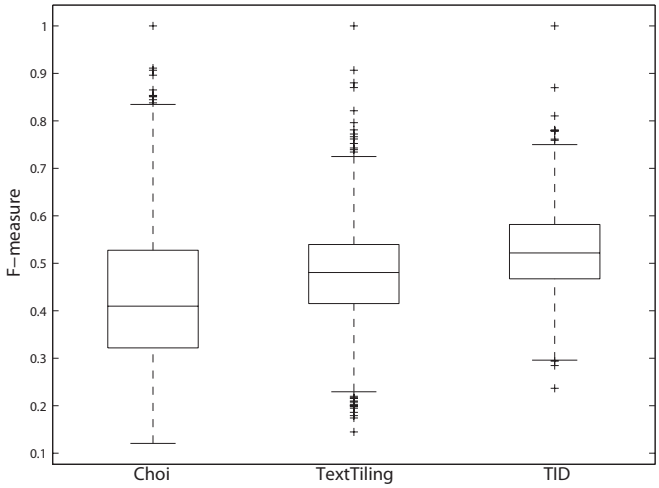


Figure 6.7: Boxplot of the sequential Reuters test environment

Based on the results of these 2000 artificial documents, sequentially composed of the Reuters and Ohsumed datasets, the conclusion can be drawn that the proposed technique performs similar or better compared to two well-known topic identification techniques in the context of the identification of the topic boundaries.

6.4.2 Randomized test environment

The results of the second test scenario, in which artificial documents are composed in a random manner, are presented in this section. Figure 6.8 presents the boxplot of the obtained F-measures for the first randomized Ohsumed test set with the related scores displayed in Table 6.3.

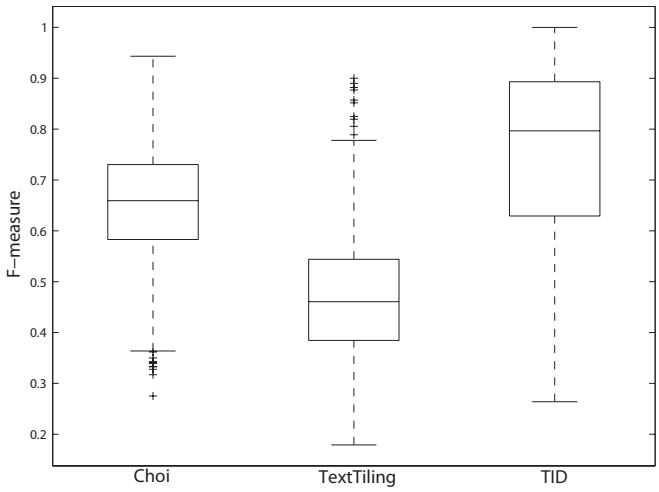


Figure 6.8: Boxplot of the Ohsumed test environment

The first significant difference between the three boxplots is the position of the boxplot of the proposed technique. The boxplot range of 0.4672 to 0.5817 is significantly higher than the other two boxplots. Also the complete ranges of all three boxplots are notably larger than in the first test environments, the largest range being for the presented technique. This is mainly due to the nature of the technique. Since some documents, chosen randomly out the datasets without any content knowledge, mostly contain only numerical information, the proposed technique fails to identify the expected structure. Choi and TextTiling are less error prone in this situation compared to the proposed technique, indicated by the range of the boxplots.

The results of the last test environment, composed of Reuters articles, are summarized in Figure 6.9 with the related scores of the boxplot shown in Table 6.4. These results are in line with the results of the previous randomized test environment: a clear distinction can be noted between the position of the boxplot of the presented technique and the position of the second highest boxplot of Choi.

Table 6.3: Quality results of the random Ohsumed test set

Dataset	Choi	TextTiling	TID
First quartile	0.5829	0.3846	0.6292
Median	0.6592	0.4606	0.7966
Third quartile	0.7304	0.5439	0.8932
Average	0.6553	0.4694	0.7457

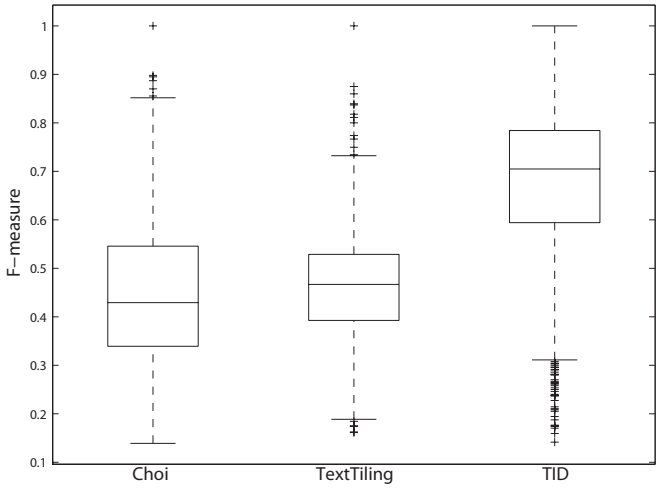


Figure 6.9: Boxplot of the Reuters test environment

6.5 Multi-vector Representation based on Topic Identification

The partitioning of a document based on the coherence or topics present in the document, forms the base of the multi-vector representation. The following considerations were taken into account when constructing the multiple vectors:

- Number of vectors:
  - Are all topics relevant for conversion to a vector?
  - Are there related topics?
- Is there a need to take all dimensions of the original document vector into account, or is an optimal selection possible?

Table 6.4: Quality results of the random Reuters test set

Dataset	Choi	TextTiling	TID
First quartile	0.3391	0.3925	0.5942
Median	0.4292	0.4667	0.7049
Third quartile	0.5457	0.5287	0.7842
Average	0.4516	0.4642	0.6712

- Is there a need to differ the weighting for every topic vector, as cited in [Reynar, 1999]?

In the further process of the research, the obtained coherent documents are considered to be independent new documents. This means that the documents are preprocessed, weighted and normalized as if they are not related to the original documents. These assumptions have the following implications:

- All topics are considered to be relevant. No similarity measurement is performed after the topic identification. It is assumed that the topics are sufficiently distinct based on the nature of lexical chains. If two topics are closely related, they share a considerable set of terms and therefore they should be linked with a set of lexical chains.
- Because every coherent document is treated as an independent document, the sum of raw frequencies of every coherent document should be equal to the raw frequencies of the original document. When normalization and weighting is applied, the resulting figures differ from the original document.
- An optimal selection of terms for every topic vector is left as future work. In the process, apart from the preprocessing, an additional filtering step can be applied for terms known to occur frequently, such as company names or labels for document parts. This filtering step requires considerable a-priori knowledge of the document collection.

Figure 6.10 gives a schematic view of the result of the multi-vector identification process. Section 7.1 of the following chapter will discuss several results comparing the usage of a single and multi-vector representation in a search engine.

6.6 Identification and Extraction of Multiple Language Sections

The set of techniques presented in Section 6.2 can also be utilized to identify and extract different language parts in a document. The process to identify and

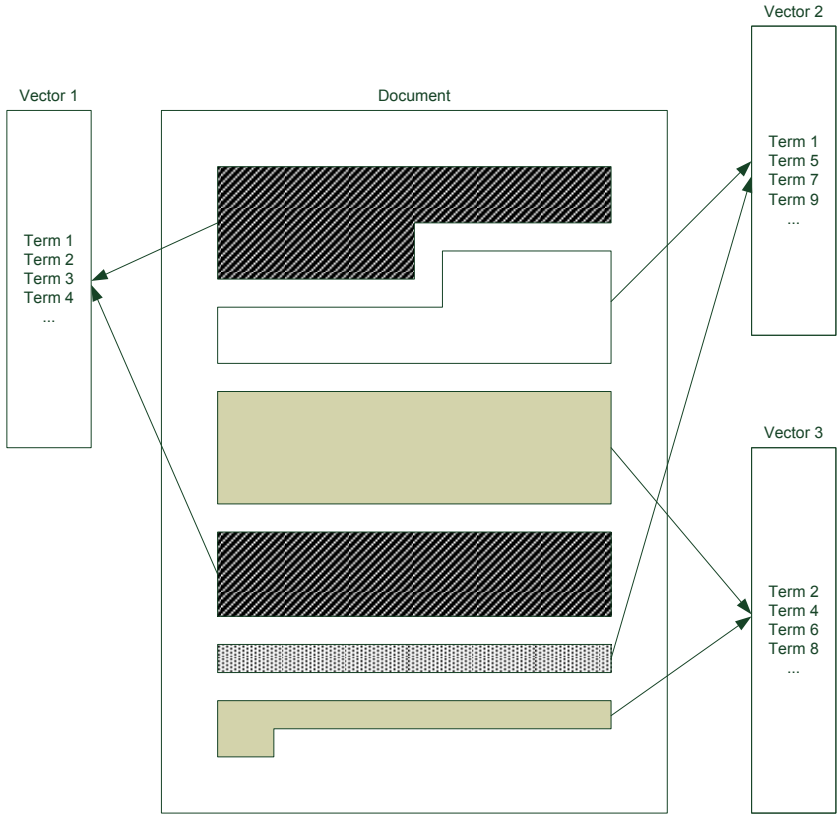


Figure 6.10: Schematic overview of the relations between a multiple vector representation of a document and its content

extract the different language sections differs from the original process in the identification of the lexical chains. Instead of using all possible lexical chains, only a specific set of chains based on stopwords is retained. All stopwords of one language form one long lexical chain. This chain can be obtained by converting all language-specific stopwords to a common term or language identifier, e.g. 'EnglishTD' for English. The other chains are omitted from further processing, resulting in a sparse coherence matrix. This sparseness enables the usage of this technique on sentence level. This enables the identification of citations or sayings in different languages. In Figure 6.11, a small document is shown composed of three languages: English, Dutch and French. The left figure displays the original text, the right figure displays the converted text containing the language identifiers. In Appendix D.1, the stopwords are displayed for the languages Dutch, French,

German and English. Important in the selection of stopwords for this technique, is the requirement of a stopword occurring in a single language. Words such as 'das' occur in Dutch and German.

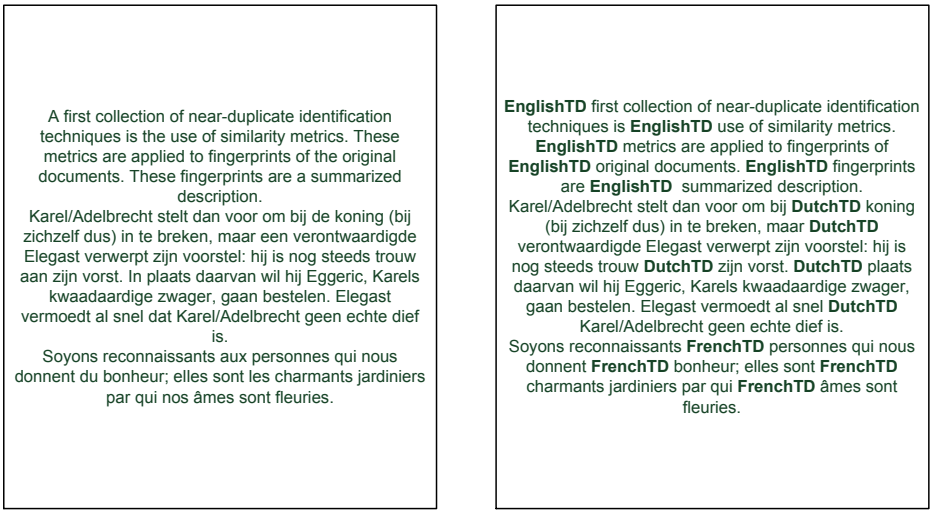


Figure 6.11: Example of three language chains in a document based on a set of stopwords. The used identifiers are 'EnglishTD', 'DutchTD' and 'FrenchTD'.

6.7 Conclusions

A novel technique was introduced to automatically identify the topics present in a document, based on the presence of lexical chains. Since every topic can be related to a specific set of words, the coherence of a document can be quantified using these lexical chains and the proposed similarity measure. The application of spectral graph partitioning techniques on the coherence graph transforms the original input documents to a new metric space. This transformation has two benefits in the topic identification process. First, the required number of topics can be identified based on the reordering of the Laplacian matrix of the graph. Second, a clustering technique can be applied on this new metric space in order to obtain the required number of components, each representing a topic.

This topic identification technique differs from existing approaches because these current techniques identify subtopic document parts. Two large experiments based on standardized datasets were performed to validate these techniques. The results indicate that the techniques resulted in similar to better results compared to the subtopic identification in a sequential test scenario, whereas in

a randomized test scenario the proposed technique outperforms the two other subtopic identification techniques. This last result indicates that the application of these graph techniques enables the identification of non-contiguous topics in a document.

Apart from the identification of topics, a variant of this technique enables the identification of multiple languages in a document. When only the lexical chains are retained covering a specific language-dependent set of stopwords, a coherence graph is constructed indicating the language structure of a document.

Following the topic identification process, the topics are used to represent the document in multiple vectors. Each vector describes a topic of this document, and therefore provides a link to this document. This topic is also related to a part of the document, so that part can also be returned as part of the result set of a query as shown in the next chapter.

In Appendix D.2, the related algorithmic descriptions are provided.

Chapter 7

Applications

7.1 Search Engine

Nowadays the importance of search engines is commonly recognized. In large unstructured document collections, the application of a search engine is often the only manner to obtain the required information within a reasonable time span. The average performance of a search engine is often not high, according to the user's IN, and this is depending on the characteristics of the document collection used in the search engine [Bar-Ilan, 2002, Liu et al., 2007b]. Constructing a multi-vector representation of a document aims to improve the retrieval performance of a search engine, because the information present in the content of the document is more accessible. The average representation of one single vector is converted to a set of more specific vectors.

In the following sections, the validation procedure for the improved retrieval performance based on the multivector document representation described in Chapter 6 is explained, and the obtained results are discussed.

7.1.1 Experimental setup

Each document subjected to the explained techniques, is preprocessed in the following manner:

- any headers or footers describing the project or dataset were deleted from the files.
- single-sentence paragraphs were merged to a larger paragraph.
- any tables and captions were merged to a larger paragraph.
- all words were converted to lower-case words.

- stopword removal was applied to each document, using a list of 622 English stopwords.
- stemming using the Porter stemming algorithm to reduce the number of terms
- the TFIDF weighting scheme was used.

The merging process is required for the following two reasons:

- computational complexity: the computational complexity of these techniques is quadratic. Performing these techniques on individual sentences of a book, for example, can complicate the performance.
- chains: when working on sentence level, several sentences will only be linked by a few or even no term chains. Document entities without any chain, cannot be included in the Laplacian because calculating an eigenvalue decomposition of a rank deficient matrix is not possible.

The clustering algorithm used after obtaining the reduced subspace by the spectral properties of the coherence graph, is the unsupervised hierarchical agglomerative clustering method using average linkage. The number of clusters in the clustering result was set to the identified number of components.

7.1.2 Datasets

For the experimental validation of these techniques, three datasets are used to create the required test environments to obtain validation results:

- Reuters RCV1 document dataset
- OHSUMED medline dataset
- Gutenberg

In the following sections, an overview of these datasets is given. More details are provided in Appendix A.

7.1.2.1 Reuters

Six categories were selected from the Reuters RCV1 dataset with the following labels:

- GCRIM: Craw, law enforcement
- GDEF: Defense
- GDIP: International relations

- GDIS: Disasters and accidents
- GENT: Arts, culture, entertainment
- GSPO: Sports

7.1.2.2 Ohsumed

From the OHSUMED test collection, only the documents were withheld rated as definitely relevant for one of the standardized queries, as stated in the OHSUMED descriptions. This resulted in a document collection containing 101 topics (five queries returned no documents), with in total 1985 documents. All the documents in this dataset are truncated to a maximum of 250 words.

7.1.2.3 Gutenberg

Two books are selected from the Gutenberg Project [Gutenberg, 1992]. This project is the largest single collection of free books with the aim to encourage the creation and distribution of eBooks. The books were selected, without in depth knowledge of their content or story, from the top 100 of most popular books:

- Encyclopaedia Britannica, 11th Edition, Volume 4.3:
  The Encyclopaedia Britannica Eleventh Edition is a 29-volume reference work, written between 1910-1911. The part used of this encyclopaedia is volume 4, part 3. This document discusses subjects ranging from "Brescia" to "Bulgaria". After the pre-processing step, this document has the properties listed in Table 7.1.
- A Christmas Carol:
  This book was written by Charles Dickens, and was first published in 1843. The book contains five large parts describing an old and bitter miser, Ebenezer Scrooge, who experiences a profound experience on Christmas Eve. The properties of this novel after the pre-processing phase are also shown in Table 7.1.

Table 7.1: Properties of the two books of the Gutenberg project after the preprocessing phase

Document	# of entities	# of words
Britannica	1599	305516
Christmas Carol	198	157052

7.1.3 Validation using standardized datasets

Apart from the Gutenberg books, a 'random test scenario' was constructed using the previously described datasets OHSUMED and Reuters. This scenario consists of the concatenation of a random number of random documents to generate a series of artificial documents, similar as described in Section 6.3.2.

The document entities in this test environment are paragraphs delimited by blank lines in the original document. Each paragraph is limited to ten sentences, so larger paragraphs were split. Every newly constructed document contains a random number of categories of the applied dataset. The number of selected documents varies between two to ten documents. Duplicates and multiple documents from the same directory are allowed in the construction process of the artificial documents.

The objective of this test scenario is twofold. In a first phase, the number of components within a random document is identified. In the optimal solution, this number equals the number of topics of the original documents. With the previously described set of techniques, one or more components are obtained. The second step is to compare the retrieval of the extracted components with the retrieval of random documents. This comparison is performed with the average F-measure, as described in Section 2.7.1.1.

7.1.4 Results

7.1.4.1 Reuters

The first step in this process is to validate the technique to identify the number of components. As described in the previous section, an artificial document is composed of different articles originating from different topics. Table 7.2 shows the results of applying this technique on 1000 artificial documents. Since the artificial documents do not necessarily contain the same number of articles as topics, the sum in this table is not equal to 1000.

The second step in the validation process is the validation of the retrieval performance. Twenty queries are constructed in the following manner: the five highest TFIDF weighted terms are selected from twenty random components. Five of these queries are shown in Table 7.3.

Table 7.2: Results of the identification of the number of components

Matching number of documents	126
Matching number of topics	572
Faulty identification	202

Table 7.3: Subset of 20 composed Reuters queries based on the highest TFIDF weights of 20 random documents

Query ID	Query
1	turkey, commit, vocational, turkish, energy
5	prodi, italy, emu, italians, deficit
10	internet, glass, telephone, tax, government
15	achieve, production, vista, buena, films
20	young, military, army, men, service

These queries are executed twice to compare the retrieval performance of the extracted components as separate vector representations with the artificial documents containing a selected component. The results are shown in Table 7.4. The highest F-measure obtained in the collection of extracted components was 0.72. The lowest F-measure was 0.24. In the collection of artificial documents, the highest F-measure was 0.71. The lowest F-measure was 0.1. These figures indicate the improved accessibility and retrievability in a search engine, when the multi-vector representation techniques are applied.

Table 7.4: Average F-measure retrieval performance in the Reuters and Ohsumed test environments

Dataset	Artificial documents	Components
REUTERS	0.34	0.61
OHSUMED	0.42	0.69

7.1.4.2 Ohsumed

The same two steps as described in the previous section, are performed on the Ohsumed test environment. Table 7.5 gives an overview of the results of the SSI-technique for identifying the number of components.

The identification technique is able to correctly identify the imposed structure of almost 66% of the documents. In only 6% of these tests, the difference between the identified number of components and the real structure comes to more than two. For the second step, the retrieval performance is validated using the 96 queries of the OHSUMED environment. The average results were described in Table 7.4. In this test environment 91 out of 96 F-measures of queries executed with the extracted components score higher than executing the same queries on the collection of artificial documents. These five lower scoring queries were all

related to the previously described difference between the calculated number of components and the real number of documents.

Table 7.5: Results of the identification of the number of components

Matching number of documents	158
Matching number of topics	654
Faulty identification	184

7.1.4.3 Encyclopaedia Britannica

The original Britannica Encyclopedia was divided into 1599 document entities. The application of the previously described techniques resulted in 35 components. This number is mostly affected by the presence of so-called isolated components, components without one or more chains connecting it to another part in the document.

A first remarkable extracted component is a component containing the paragraphs explaining the production and legislation of beer. Several chains link these paragraphs: apart from the expected terms such as 'brewing' or 'beer', the traditional 'production'-related terms such as 'fermenting', 'cleansing' and 'malt'. Less obvious are the chemical terms and the names of different countries.

Another large component explains the role of bricks in construction. Apart from the expected paragraphs about styles of bricks and construction-related items, some cities or towns such as Budapest are related to this paragraph. The presence of descriptions of well-known buildings relates these paragraphs to the larger part of construction. Other remarkable components are components concerning painters or the act of painting; and religion or religious people.

Not all paragraphs are correctly placed: in the first larger component, a person called Brewer is mentioned. Except for his name, he has no (described) relationship with beer or the process of beer making. The part describing the life of this person contains few links to other paragraphs, therefore it is linked to the component of beer.

To indicate the increased retrieval performance, two queries for the two largest components were constructed in the manner described in Section 7.1.4.1. These two queries for the respective components are:

- brew, cleans, barrel, amount, tax
- load, brick, stress, centuri, princip

The cosine dissimilarity for these two queries performed on the two components and the full books are shown in Table 7.6. These values indicate that the two extracted components are more coherent than its related full text.

Table 7.6: Comparison of the cosine dissimilarity values of two queries performed on an extracted component and on the full book

Query ID	Book	Component
1	0.22	0.61
2	0.31	0.57

7.1.5 Conclusions

Using the technique discussed in Chapter 6, an information retrieval step is performed to compare the performance of the developed techniques with retrieval on full text. The standardized datasets Reuters and Ohsumed and two books of the Gutenberg project are used to validate these techniques. From the results, it can be concluded that the developed techniques from Chapter 6 can be integrated in a search engine environment, improving the accessibility of information. In a search engine environment, a document needs to be partitioned in coherent document parts to make them retrievable, independent of other information in the document that can distort this retrieval. The statistical approach to subdivide a document in multiple coherent subtexts offers also other possibilities. Apart from the indicated retrieval performance in the validation process, the set techniques can also deliver an input to summarization techniques or other text mining tools.

7.2 Near-duplicate Identification

7.2.1 Introduction

The rapidity of document creation is one of the other possibilities the new information world offers. Often a new document is composed of existing components of other documents, e.g. manuals or presentations, or multiple versions of the same documents are created, such as emails. Therefore, in document management, the problem of (near-)duplicate documents is related to the issue of versioning. This unavoidable presence of near-duplicate documents has a serious impact on the performance of the supporting KMS, as well as on the performance of the knowledge workers themselves. The presence of redundant files in the document collection requires additional computational and memory capacity to process these often very large collections. The retrieval of documents is hampered due to the presence of these files because, besides the confusion of the most recent version, the quality of the returned cluster of documents is biased when a large number of near-duplicates are present. Identifying near-duplicate documents thus benefits many document-based applications. In a retrieval process, identifying the near-duplicates enhances the quality of the

results. Clustering processes performed for mining objectives, such as expert location [Vertommen et al., 2008] or knowledge identification, obtain more precise results.

Obviously two documents are considered to be near-duplicates if their content is highly similar. However detecting this similarity is more complicated than a byte-by-byte match. For instance, a new document can be created by omitting selected parts of the original, such as templates or copyright statements. Many other causes explain the existence of near-duplicate documents, e.g. typographical errors, plagiarism or versioning.

The definition of near-duplicates according to Pugh [Pugh and Henzinger, 2003] is as follows:

> documents that have more than l features in common are considered to be, at least, near-duplicates.

Other definitions include a limit on length difference [Conrad and Schriber, 2004]. As stated in [Yang and Callan, 2005], simple percentage-based definitions neglect the structural differences between the different types of duplicates. Besides exact duplicates, Yang [Yang and Callan, 2005] distinguishes seven different types of duplicates:

- Block Edit
  When one or more paragraphs are added to, or removed from, another document to create a new document.

- Key Block
  Adding one or more paragraphs, typically to complete a template or another formal layout type.

- Minor Change
  The altering of a small set of words within a few paragraphs.

- Combination of Minor Change and Block Edit

- Block Reordering
  The shuffling of a set of paragraphs to create a new document.

- Bag-of-word similar
  If more identical words are identified above a predefined threshold e.g. 80%, and non of the above categories comply, then these documents are considered to be bag-of-word similar. This notion originates from the use of indexed lists and the cosine dissimilarity (see Section 2.2.2.1).

An extensive overview of these different types is given in [Yang and Callan, 2005].

7.2.2 Existing approaches

The research domain of exact duplicate identification is a domain that has been explored extensively. This detection is frequently performed by calculating a unique hash value, based on well-known hash functions such as MD2, MD5 or SHA [Knuth, 1973]. These hash functions have three desired properties [Skala and Hrdek, 2009] that make them the favorite approach to identify exact duplicates. These techniques however are not able to cope with slight formatting changes or variations in the content. Near-duplicate detection requires a different set of techniques. Central in the process of identifying near-duplicates, is the notion of distance. As mentioned in the Section 2.2.2.1, a large set of distance measures exists to compute the distance between two documents.

A first collection of near-duplicate identification techniques uses previously described similarity metrics. These metrics are applied to fingerprints of the original documents. These fingerprints are a summarized description of a document (part), e.g. a bag of words or hash values, so that a full term by term comparison of the documents can be avoided. These measurement approaches are similar to weighted document clustering. Every fingerprint of a document (part) is compared to every other fingerprint to group potentially near-duplicates. The near-duplicate similarity is based on the number of common fingerprints. Several optimizations were suggested to this approach based on advanced feature selection or a minimum amount of term overlap [Heintze, 1996]. In literature, several metrics for the detection of near-duplicate documents can be found. Resemblance, cosine measure or Edit distance [Ukkonen, 1985] are some of the available measures.

Another popular collection of near-duplicate detection techniques uses shingles [Pugh and Henzinger, 2003] and is often used in web search engines. A shingle is a set of contiguous terms of the document, often transformed to numerical values. A full document is thus represented by a collection of shingles, also called a sketch, storing the shingles itself or their hash values. Documents that share a certain number of shingles are considered to be near-duplicate.

Collection statistics is a recent advent in the identification research [Chowdhury et al., 2002]. This set of techniques uses the statistics of a document, e.g. inverse document frequency, as a manner to identify a similar appearance of terms in different documents. A filter with a predefined threshold value is applied to identify the relevant terms.

The research proposed in this chapter is a combination of fingerprinting with collection statistics. As stated in Chapter 6, a document is considered to be composed of document entities. These document entities are related to each other by term repetitions. The assumption is that these repetitions are an indication of the coherence of a document, and that they also can be used to detect near-duplicates. In Section 7.2.2, a brief summary is given of the existing state-of-the-art approaches in the domain of (near)-duplicate

identification. Section 7.2.3 presents an overview of the near-duplicate process and its three phases to identify the near-duplicateness of two documents. The next sections present the validation process on two large books of the Gutenberg project, and summarize the presented technique.

7.2.3 Identification process of near-duplicate documents

The identification process of near-duplicate documents is composed of three phases:

1. Identification of coherent document components of a document (see Chapter 6)

2. Identification of related coherent document components of a pair of documents

3. Identification of near-duplicate document entities of a pair of documents, based on common lexical chains

7.2.3.1 Identification of coherent document components of a document

A first phase in the process of identifying near-duplicate document parts is to subdivide a document based on its coherence, as discussed in Chapter 6. To summarize, the main steps in this phase are

1. Construction of a coherence graph based on a linkage matrix

2. Construction of a normalized symmetric Laplacian of the coherence graph

3. Identification of the number of coherent components using the obtained Laplacian

4. Partitioning of the coherence graph using its spectral properties to obtain the components.

After the identification of the different coherent document components, for each of these components a fingerprint in the form of lexical chains is available. Besides the fingerprint, the multiple vector representation is obtained. These two component representations are used in the second and third phase of the near-duplicate identification process. Similar to the coherent component identification process, this process is based on common lexical chains in the different identified components.

7.2.3.2 Identification of related coherent document components

The identification of near-duplicate document entities could be performed by pair-wise comparing all document entities of both documents. This processing step however would be computationally prohibitive, and can be optimized by only comparing those coherent document parts that are statistically closely related. This relationship is quantified with e.g. the cosine dissimilarity measure between the term frequency vectors in the common term space representing the different coherent document components. Only those closely related coherent components (above threshold value) will be further examined in the second phase.

7.2.3.3 Identification of near-duplicate document entities based on common lexical chains

When two different document components share multiple lexical chains in their document structure, a probability can be calculated indicating the possible near-duplicateness between this pair of document components. In Figure 7.1, this idea is illustrated. Two document components are displayed, the right component is an adaptation of the original left component. In these components, two lexical chains are indicated. While the right text is considerably adapted, the lexical chains remain to a large extent intact.

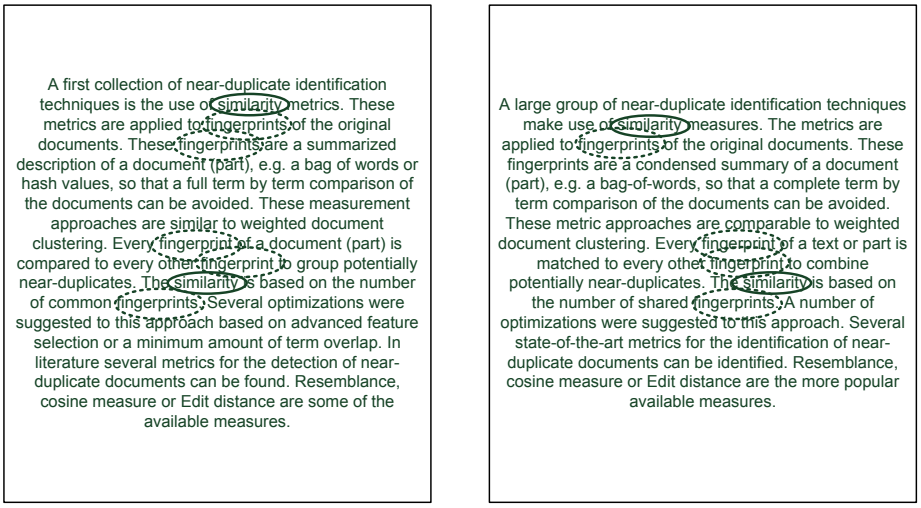


Figure 7.1: Graphical representation of two near-duplicate documents. Although the right document is an adapted version of original document (on the left), the indicated chains remain almost unchanged.

A similarity measure to assess near-duplicateness for a pair of document

components, based on the number of common lexical chains, can be obtained. For chain TC_i this measure is defined as

$$s(i) = \begin{cases} min(|TC_{i,1}|, |TC_{i,2}|) & \text{if } |TC_{i,1}| \text{ and } |TC_{i,1}| > 0 \\ \text{-}min(doc1(i), doc2(i)) & \text{otherwise} \end{cases}$$

$|TC_{i,1}|$ and $|TC_{i,2}|$ are the number of occurrences of term i in document entities 1 and 2. The overall similarity measure between the document entities doc1 and doc2 is defined as the average sum over all lexical chains.

$$sim(doc1, doc2) = \frac{\sum_i s(i)}{min(n, CL2)} \tag{7.1}$$

$\sum_i TC_{i,1}$ and $\sum_i TC_{i,2}$ are the number of chains in document entities 1 and 2. Not all lexical chains are suited for this comparison process. Besides the filtering step performed in the preprocessing phase that converts a document into a processable list of terms (see next section), an additional filtering step is performed based on the Zipf law. Other approaches include the selection of terms that at least obtain a weight above a predefined threshold value.

7.2.4 Validation

In this section, the experimental results of the near-duplicate identification techniques are presented. These techniques were applied to two large books from the Gutenberg dataset. These books were selected based on the results of the topic identification technique on these books (see Section 7.1.4.3). These results indicated that the book entitled 'Christmas Carol' could not be subdivided according to an underlying topic structure, while the volume of the Britannica Encyclopaedia was judged to contain 35 different topics. As such, these two books offered two collections of substantial size, and two different topic scenarios to validate this technique. The datasets and applied validation procedure will be explained in the following sections.

7.2.4.1 Experimental setup

Each document subjected to the explained techniques, is pre-processed in the following manner:

- any headers or footers describing the project or dataset were deleted from the files.
- single-sentences paragraphs were merged to a larger paragraph.
- any tables and captions were merged to a larger paragraph.
- all words were converted to lower-case words.

- stop words were removed using a list of 622 English stop words.
- stemming was performed using the Porter stemming algorithm to reduce the number of terms [Porter, 1980].
- the TFIDF weighting scheme was used.

The clustering algorithm used after obtaining the reduced subspace by the spectral properties of the coherence graph, is the unsupervised hierarchical agglomerative clustering method using average linkage [Jain et al., 1999]. The number of clusters in the clustering result was set to the previously identified number of components. The goal of the following scenario is to validate the near-duplicate identification techniques on one of the more complex near-duplicate environments, i.e. variation on 'bag-of-words similarity' [Yang and Callan, 2005].

In a first phase, a percentage of random terms in the original document are randomly changed into words from the British National Corpus (BNC) word list [Clear, 1993]. No prior selection is made on the type of words from the original document or on the words of the BNC word list. In the second phase of the process, the new document is subdivided into its coherent components. In the third phase, the near-duplicate technique explained in Section 7.2.3 is performed on the coherent components identified in the original document and the distorted document. This scenario is repeated for different percentages of changed terms to validate the robustness of the near-duplicate identification process. The percentages of distortion are 10%, 25% and 50%. This distortion has no effect on the number of document entities in each of the documents.

7.2.4.2 Results

The results are quantified based on the identified relationships between document entities:

- A correctly identified relationship between document entities is a relationship between the corresponding entities in the original document and the distorted document (e.g. entities numbered 35 in the original document and distorted document).
- A false negative relationship is a non-identified relationship between corresponding entities.
- A false positive is a relationship identified between non-corresponding entities. (e.g. entity 35 in the original document and entity 17 in the distorted document).

Encyclopaedia Britannica

The original document is subdivided in 35 components. This number is mostly defined by the presence of so-called isolated components, i.e. components containing no important chains connecting it to another part in the document. Distorting 10% of the terms of the original document has a minor influence on the number of identified components. In this distorted document, 33 components are identified mostly due to creation of new chains connecting isolated components. Changing 25% and 50% of the terms increases the number of identified components to 44 respectively 57. The reason for this increasing figure is the lower chance of reselecting the same term from the word list, resulting in a lower coherence between the document entities. In Table 7.7, an overview is given of the identified relationships between the entities.

Table 7.7: Summary of the result of the near-duplicate detection techniques on the book titled 'Encyclopaedia Britannica'

	10% distorted	25% distorted	50% distorted
Correct	99%	78%	47%
False negative	1%	19%	48%
False positive	0%	3%	5%

A distortion of 10% returns an almost full identification of the correct relationships between the corresponding document entities. In the second scenario, the overall scoring of correctly identified near-duplicate relationships remains on a high level. The presence of false positives for this book is related to smaller document entities containing few important chains. Breaking one of these chains falsely relates these document entities to other entities based on a too low number of common terms. This implies that the size of the document entities should be selected carefully. The near-duplicate identification technique still rates the document as almost 50% near-duplicate when 50% of the terms are different. The reason for the higher score for false negatives is again the smaller size of the document entities, due to the isolated entities.

A Christmas Carol

The SSI-algorithm of Section 6.2.3 identifies the original book as one fully coherent component. Several important chains such as 'scrooge', 'ghost' and 'christmas' result in a dense coherence graph. Only the distortion of 50% of the terms in this book results in a higher number of components, namely three. Two of these components are isolated document entities. Table 7.8 summarizes the identified relationships. For the first two scenario's, similar conclusions can be drawn for this book. Remarkable is the higher scoring for the false positives for the third scenario. This indicates that the presence of a large number of important,

long term chains result in a high number of false positives. These overlapping chains are broken in different entities, omitting the differences in chains between a pair of non-related document entities.

Table 7.8: Summary of the result of the near-duplicate detection techniques on the book titled 'A Christmas Carol'

	10% distorted	25% distorted	50% distorted
Correct	96%	77%	49%
False negative	2%	21%	41%
False positive	2%	2%	10%

7.2.5 Conclusions

The preceding sections introduce a novel near-duplicate identification technique based on the coherence of the compared documents. The assumption is that, similar to topic identification, near-duplicate documents can be identified by the repetitive use of a selection of terms, so-called lexical chains. In a first step of the process the documents are subdivided in, if present, multiple coherent document components. This is performed by quantifying the coherence based on the presence of common lexical chains in the document. The result of this coherence quantification is an undirected, weighted graph. Using the spectral properties of this graph, the number of coherent components is identified and the document is divided in components. The second step in the near-duplicate identification process is the comparison of the components of the different documents based on the number of common lexical chains. A similarity measure is employed to quantify this relationship. The validation dataset were two books of the Gutenberg Project. The validation procedure was to distort the content of these books, by changing a varying number of terms originating from the BNC word list. The near-duplicate techniques were applied on the original document and the obtained adapted versions of these books. The results indicated that in the case of a large number of interchanged terms the techniques were still able to identify a considerable set of these near-duplicate document entities. This effect indicates that several chains need to be broken, before a near-duplicate relationship remains unidentified.

Chapter 8

General Conclusions and Future Work

8.1 Conclusions

Two main objectives were formulated at the outset of the research presented in this dissertation. The first objective was to provide an overview of, and develop advances for, clustering techniques to support the handling of large document collections in a KMS. Also a technique should be developed to assist the identification process of metadata of a document, based on the content of this document. The general conclusions for this clustering part are described in Section 8.1.1.

The second objective was to devise techniques to improve the accessibility and retrievability of information in a KMS. An approach for this objective was suggested at the start of the project which aims to represent the content of a document in multiple vectors. The impact of the expected improved accessibility should be demonstrated in an application context. The general conclusions concerning this representation format are described in Section 8.1.2. The applications based on the advances and gained insights of parts I and II are discussed in Section 8.1.3.

8.1.1 Cluster techniques

The initial research activities were concentrated around the clustering context and the collection of existing clustering techniques that are suitable for application in KMSs. A document collection included in a knowledge base often has a considerable dimensionality, which hampers the processing of these datasets due to the so-called curse of high dimensionality. Several research attempts are discussed

in this dissertation, providing different insights in the domain of clustering and their behavior in large and high dimensional environments.

The overview of clustering techniques inspired three research tracks, which were profoundly explored:

- An incremental clustering algorithm
- A dissimilarity measure
- A weighting scheme

The motivation behind the tree-based incremental clustering algorithm is twofold: the first interesting aspect is the desired incremental property for the updating process of the document tree. In contrast of the batch updating combined with a full clustering process which often occurs nowadays, an incremental clustering effort results in a partial recalculation of an existing clustering solution. The second motivation is the interesting computational and memory complexity of tree-based search and insert algorithms.

The clustering algorithm was tested on several standardized datasets, with different sets of parameters and parameter values. The results of this clustering algorithm were in overall not convincing. For some test sets, the incremental clustering algorithm resulted in a better cluster quality compared to the well-known agglomerative hierarchical clustering algorithm, while for other datasets this performance was not reached.

Conclusion: For tree-based incremental clustering algorithms, a term vector representing the documents of the subgraph is not improving the cluster quality when matching this vector to the term vector of a new inserted document. The influence of the curse of high dimensional is not reduced, even when feature filtering techniques are applied on this vector.

The second approach, the adaptive dissimilarity measure, is based on a selection of common features of the compared document vectors. Two types of this measure were constructed: a static or dynamic number of features which are selected for a distance measurement. For every distance calculation, a new subspace is created wherein this measure is applied. The measure was validated in several test sets with varying collection size and dimensionality.

Conclusion: The approach of feature selection combined with a dynamic selection of the features has been integrated in a dissimilarity measure, which proved to provide better cluster quality results in large datasets.

The last research track concerns the weighting scheme based on the theory of Zipf. According to this theory, the terms appearing in the mid-range when these terms are sorted by decreasing frequency are the most informative ones. This reasoning formed the basis to construct a scheme assigning the highest weights to these terms. The results of this weighting scheme conducted on several datasets however did not indicate a significant improvement compared to well-known weighting schemes.

Conclusion: The theory of Zipf, selecting the most informative terms in a document or document collection, is not directly related to the cluster structure. According to the performed tests, the most informative terms are not directly the most discriminative terms for the cluster structure.

8.1.2 Multi-vector representation of documents

The second group of research activities was situated in the domain of a multiple vector representation of a document in a VSM. Each vector is related to a topic present in the content of the document. These topics are identified based on the reoccurrences of topic-defining terms. These reoccurrences are used to define so-called lexical chains in the document, describing the coherence of the document. The quantification of these coherence relationships thus results in a graph representation. Using techniques of the domain of spectral analysis, the underlying connections can be exploited for the partitioning of the graph, and the identification of the number of topics in the document. Each identified subgraph describes a topic of the document, relating different document entities that together constitute a topic. Each of these topics is then converted to a vector representation by indexing the newly obtained document parts.

Conclusion: By using lexical chains and a coherence quantification function, it is possible to describe the coherence of a document. Using a graph partitioning technique, such as spectral analysis, a non-successive reordering and subdivision of a document according to its topics can be obtained.
Conclusion: The proposed SSI technique is able to estimate the number of topics present in the document according to the lexical chains and the coherence quantification.

8.1.3 Applications

The last objective was the development of applications based on the experiences and advances developed in the two previously described research tracks.

In the first part, a semi-automated approach named SAME was presented

that aims to support the identification of metadata of a document, based on the content of this document. This identification process results in so-called document models, which are the aggregation of document types and their related metadata labels. These models are the representatives of clusters of documents in a KMS. This approach should replace the manual identification process that currently is used.

Conclusion: The principle for the SAME technique, which is the process of filtering of the cluster-defining terms of a repeated clustering process, results in a useful layered collection of MLCs and terms supporting the context identification and enrichment for documents defining the knowledge base for a KMS.

In the second part, two functionalities based on the multi-vector representation format were presented. The first functionality is the integration of this representation format in a search engine. The objective of this integration is to improve the accessibility of information in these systems, because small topics are discriminated in the single vector representation.

The second functionality is the near-duplicate identification of documents. Based on the developed topic identification technique, parts of different documents describing a similar topic can be compared based on the lexical chains they contain.

Conclusion: The developed multiple-vector representation can be integrated in, at least, two functionalities. The integration of this representation format in the traditional document search functionality enhances the retrievability of the required information. By comparing the query vector to the different topic based vectors of a document, an enhanced retrievability for these topics is obtained. The improved availability has been verified based on comparisons with the single-vector representation of a document.
The second functionality is the near-duplicate identification technique. Based on the performed validation tests, near-duplicateness based on bag-of-words similarity can be identified with this technique.

8.2 Main Contributions

As already indicated, the contributions of this research can be located in two domains.

- Several research efforts were located in the domain of clustering. These clustering techniques were applied on document collections, which provided insights in the complexity of these techniques and identified several opportunities for further research.

- The representation of a document with multiple vectors is located in the domain of topic identification, which is part of the domain of NLP. The lexical information of the individual terms is omitted, in order to integrate the technique in existing applications.

In the following sections, the contributions in these two domains on the technological and application level are briefly summarized.

8.2.1 Contributions on the technological level

The applied techniques in the presented research are, or are related to, clustering techniques and the application of these techniques on a document or document collection. In the following paragraphs, a short overview is provided of the main contributions on technological level in the domain of clustering and topic identification.

High dimensionality

The often high dimensionality of indexed document collections requires an adapted approach in order to reduce the effect of the curse of high dimensionality. Besides the research efforts invested in an adapted weighting scheme and incremental clustering algorithm, the largest advance is obtained with a specialized distance measure. The Pairwise Adaptive Dissimilarity measure, which selects a limited set of features to construct a unique subspace for each distance calculation, is a novel technique which integrates feature selection. No other technique is known in literature to counteract the curse of high dimensionality in this way.

Topic identification in a document

A novel technique is introduced to identify topics in a document. Based on lexical chains, the coherence is expressed in a graph which is partitioned by means of its spectral properties. This new combination of techniques is able to identify non-contiguous topics in a document, an approach which, to the best of our knowledge, is not known in literature.

Number of topics in a document

To identify topics in a document, an essential parameter for this process is the expected number of topics present in the content. To assess this number, a new technique named Square Segments Identification (SSI) is deduced from the spectral properties of the graph related to the document. This technique is able to produce multiple results, each assigned with an occurrence probability.

Multi-vector representation of a multilingual document

Based on the topics present in a document, a multi-vector representation for a

document is presented in a VSM. In literature, no research effort can be identified that combines the vector representation of a document with the topics in the content of the document itself. Besides the topics, the technique is also able to identify and locate the different languages in a document.

8.2.2 Contributions on the application level

The research conducted in the two research domains resulted in a new application and interesting advances for at least three applications. In the next paragraphs, these existing and new applications are briefly summarized.

Search engine
The construction of a multi-vector representation for a document enhances the retrievability of information in the context of a search engine. As a single vector is an average representation of a complete document, the smaller topics are underrated in this vector. By discovering and separating the topics in a document, the information becomes more accessible.

Near-duplicate identification of documents and document parts
By describing and quantifying the coherence of a document, a new approach for near-duplicate identification became possible. The identified lexical chains can serve as fingerprints of documents and document parts, and they allow a near-duplicate comparison based on the topics present in the content of the documents. By identifying these topics, the similarity or dissimilarity between the compared document parts is more pronounced.

Semi-automatical identification of metadata of a document
The SAME technique assists in the construction of document models to represent documents. In literature, no such application can be identified that aids such a manual metadata identification process. The results of this technique assist the identification of document types and the related metadata based on co-occurrences of terms in different documents.

8.3 Future Work

A number of items were left unexplored in the research presented in this dissertation, and this could motivate further research. In this section, an overview is provided for possible future research paths for Parts I and II of this dissertation. The main item that should be considered for future work for both parts is the roll-out of the presented techniques in an industrial environment for a long time span. For the presented research tracks, it was not always possible, mainly due to

confidentiality issues, to have the disposal of large industrial datasets for validation purposes.

8.3.1 Part I: Clustering techniques

The future research for part I discussing the clustering techniques is focused on the incremental clustering algorithm and the weighting scheme. The suggested future research for the incremental clustering algorithm in general can be summarized to two items. The first item concerns the insertion procedure of the algorithm. Instead of inserting a document in the root node of the tree, insertion at the level of the leaf nodes is an option that has been suggested in literature for tree-based incremental clustering. The advantages include an improved invariability of the input order and improved cluster quality. As main disadvantage, in practical applications, an augmented computational time was recorded. The second item concerns the integration of the Pairwise Adaptive Dissimilarity measure in the different processes. As these two research tracks were conducted in the reverse order in time, the influence is not studied yet. For the Zipf-based weighting scheme, this future research includes the application of the weighting scheme on other datasets to assess the applicability in search engine environments.

For the SAME technique, a number of future research tracks are identified:

- The current SAME technique requires several parameter values during its process. The required number of clusters during each iteration and the required term frequency (to assess the importance) are the most important parameters for which a more profound value assessment is desirable. Other parameters, such as the filtering parameter in the co-occurrence matrix, can be included in the user interface as the results for varying parameter values can provide valuable information to the end-user.

- The assessment for relationships between the terms is currently calculated via a first order co-occurrence calculation. Using the PCA technique, higher order relationships could be obtained.

- Considering the results of the Pairwise Adaptive Dissimilarity measure, the integration of this measure in the SAME technique is expected to provide improved results.

- The recognition of synonymy and polysemy by means of dimensionality reduction techniques such as LSI could provide new insights in the metadata identification process.

- Besides a commercial implementation of the presented SAME tool, enhancements are required to suggest values for metadata labels for a specific document. In the current tool, this analysis needs to be performed by the creator of the document.

The last important research track concerns the design of a visualization tool for the end-user. During the validation phase of the SAME techniques, it became clear that the current testing tool was not able to adequately present the extracted information of the term network, and thus the construction of a well conceived user interface in this context is suggested.

8.3.2 Part II: Multiple Vector Representation of documents

In part II, the influence of weighting requires further analysis. Especially the weighting of terms based on information of the global document collection requires considerable attention. Optimizations steps in the SSI algorithm are required to improve the computational performance of this technique. Also tests on larger datasets are needed to further validate the performance of the developed multiple vector techniques. Especially in the context of the search engine, the apparent promising trend should be confirmed when the technique is applied on larger datasets.

For the near-duplicate identification technique, several properties of the lexical chains can be incorporated in the similarity function used in the second phase:

- The positions of the terms in a lexical chain
- The distance between, and the length of the lexical chains
- The weighting of the lexical chain

Adaptations of the similarity function using these properties should enhance the near-duplicate identification.

Bibliography

[Abney et al., 1996] Abney, S., Klavans, J., and Resnik, P. (1996). Statistical Methods and Linguistics. *The Balancing Act: Combining Symbolic and Statistical Approaches to Language*, pages 1–26.

[Achtert et al., 2007] Achtert, E., Bhm, C., peter Kriegel, H., Krger, P., Mller-gorman, I., and Zimek, A. (2007). Detection and visualization of subspace cluster hierarchies. In *In Proc. DASFAA*.

[Aggarwal et al., 1999] Aggarwal, C. C., Wolf, J. L., Yu, P. S., Procopiuc, C., and Park, J. S. (1999). Fast algorithms for projected clustering. In *SIGMOD '99: Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, pages 61–72, New York, NY, USA. ACM.

[Agrawal et al., 1998] Agrawal, R., Gehrke, J., Gunopulos, D., and Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. *SIGMOD Rec.*, 27(2):94–105.

[Aliguliyev, 2009] Aliguliyev, R. M. (2009). Performance evaluation of density-based clustering methods. *Inf. Sci.*, 179(20):3583–3602.

[Anderberg, 1973] Anderberg, M. R. (1973). *Cluster analysis for applications*. Academic Press, New York.

[Andrews, 1985] Andrews, A. (1985). *The major functions of the noun phrase*, volume 1, pages 62–154. Cambridge University Press.

[Angheluta et al., 2002] Angheluta, R., Busser, R. D., and Moens, M.-F. (2002). The use of topic segmentation for automatic summarization. In *In Workshop on Text Summarization in Conjunction with the ACL 2002 and including the DARPA/NIST sponsored DUC 2002 Meeting on Text Summarization*, pages 11–12.

[Astrahan, 1970] Astrahan, M. (1970). Speech analysis by clustering, or the hyperphoneme method. Stanford a i project memo, Stanford University.

[Baayen, 1991] Baayen, H. (1991). A stochastic process for word frequency distributions. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics*, pages 271–278, Morristown, NJ, USA. Association for Computational Linguistics.

[Baayen, 2002] Baayen, R. H. (2002). *Word Frequency Distributions (Text, Speech and Language Technology)*. Springer, 1 edition.

[Baeza-Yates and Ribeiro-Neto, 1999] Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. Addison Wesley.

[Ball and Hall, 1965] Ball, G. H. and Hall, D. J. (1965). Isodata, a novel method of data analysis and pattern classification. Technical report, Stanford University.

[Bar-Ilan, 2002] Bar-Ilan, J. (2002). Methods for measuring search engine performance over time. *J. Am. Soc. Inf. Sci. Technol.*, 53(4):308–319.

[Barzilay and Elhadad, 1997] Barzilay, R. and Elhadad, M. (1997). Using lexical chains for text summarization. In *In Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization*, pages 10–17.

[Becker and Kuropka, 2003] Becker, J. and Kuropka, D. (2003). Topic-based vector space model. In *Proceedings of the 6th International Conference on Business Information Systems*, pages 7–12, Colorado Springs.

[Beckmann et al., 1990] Beckmann, N., Kriegel, H.-P., Schneider, R., and Seeger, B. (1990). The r*-tree: an efficient and robust access method for points and rectangles. *SIGMOD Rec.*, 19(2):322–331.

[Beeman et al., 1994] Beeman, M., Friedman, R. B., Grafman, J., Perez, E., Diamond, S., and Lindsay, M. B. (1994). Summation priming and coarse semantic coding in the right hemisphere. *J. Cognitive Neuroscience*, 6(1):26–45.

[Bellman, 1961] Bellman, R. E. (1961). *Dynamic Programming*. Princeton University Press.

[Ben-Dor et al., 2002] Ben-Dor, A., Chor, B., Karp, R., and Yakhini, Z. (2002). Discovering local structure in gene expression data: the order-preserving submatrix problem. In *RECOMB '02: Proceedings of the sixth annual international conference on Computational biology*, pages 49–57, New York, NY, USA. ACM.

[Ben-hur and Guyon, 2003] Ben-hur, A. and Guyon, I. (2003). Detecting stable clusters using principal component analysis. In *In Methods in Molecular*, pages 159–182. Humana Press.

[Berkhin, 2002] Berkhin, P. (2002). Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA.

[Berry and Linoff, 2004] Berry, M. J. A. and Linoff, G. S. (2004). *Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management*. *Wiley Computer Publishing.

[Bezdek and Pal, 1998] Bezdek, J. C. and Pal, N. R. (1998). Some new indexes of cluster validity. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 28(3):301–315.

[Bohm et al., 2004] Bohm, C., Kailing, K., Kriegel, H.-P., and Kroger, P. (2004). Density connected clustering with local subspace preferences. In *ICDM '04: Proceedings of the Fourth IEEE International Conference on Data Mining*, pages 27–34, Washington, DC, USA. IEEE Computer Society.

[Booth, 1967] Booth, A. D. (1967). A law of occurrences for words of low frequency. *Information and control*, 10(4):386–393.

[Borghoff and Pareschi, 1998] Borghoff, U. M. and Pareschi, R. (1998). Information technology for knowledge management. *Journal of Universal Computer Science*, 3:835–842.

[Boyd-Graber et al., 2007] Boyd-Graber, J., Blei, D. M., and Zhu, X. (2007). A topic model for word sense disambiguation. In *Empirical Methods in Natural Language Processing*.

[Brinkley, 2008] Brinkley, I. (2008). The knowledge economy: How knowledge is reshaping the economic life of nations. Technical report, The Work Foundation, London.

[Brinkley et al., 2009] Brinkley, I., Fauth, R., Mahdon, M., and Theodoropoulou, S. (2009). Knowledge workers and knowledge work. Conclusions of the knowledge economy programme, The Work Foundation, London.

[Caillet et al., 2004] Caillet, M., francois Pessiot, J., reza Amini, M., and Gallinari, P. (2004). Unsupervised learning with term clustering for thematic segmentation of texts. In *Proceedings of RIAO*, pages 648–656.

[Chali, 2005] Chali, Y. (2005). Topic detection of unrestricted texts: Approaches and evaluations. *Applied Artificial Intelligence*, 19(2):119–135.

[Chen et al., 2006] Chen, L., Zeng, J., and Tokuda, N. (2006). A "stereo" document representation for textual information retrieval. *J. Am. Soc. Inf. Sci. Technol.*, 57(6):768–774.

[Chen et al., 2004] Chen, T. Y., Kuo, F.-C., and Merkel, R. G. (2004). On the statistical properties of the f-measure. In *QSIC*, pages 146–153.

[Cheng et al., 1999] Cheng, C.-H., Fu, A. W., and Zhang, Y. (1999). Entropy-based subspace clustering for mining numerical data. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 84–93, New York, NY, USA. ACM.

[chiu Wong and chee Fu, 2000] chiu Wong, W. and chee Fu, A. W. (2000). Incremental document clustering for web page classification. In *In IEEE 2000 Int. Conf. on Info. Society in the 21st*, pages 5–8.

[Cho et al., 2004] Cho, H., Dhillon, I. S., Guan, Y., and Sra, S. (2004). Minimum sum-squared residue co-clustering of gene expression data. In *Proceedings of The fourth SIAM International Conference on Data Mining*.

[Choi, 2000] Choi, F. Y. Y. (2000). Advances in domain independent linear text segmentation. In *In Proceedings of NAACL*, pages 26–33.

[Chowdhury et al., 2002] Chowdhury, A., Frieder, O., Grossman, D., and McCabe, M. C. (2002). Collection statistics for fast duplicate document detection. *ACM Trans. Inf. Syst.*, 20(2):171–191.

[Church and Gale, 1995] Church, K. and Gale, W. A. (1995). Inverse document frequency (idf): A measure of deviations from poisson. In *Proceedings of the Third Workshop on Very Large Corpora*, pages 121–130.

[Clear, 1993] Clear, J. H. (1993). *The British national corpus*. MIT Press, Cambridge, MA, USA.

[Cohen, 1960] Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

[Collins-thompson et al., 2002] Collins-thompson, K., Ogilvie, P., Zhang, Y., and Callan, J. (2002). Information filtering, novelty detection, and named-page finding. In *In Proceedings of the 11th Text Retrieval Conference*.

[Comer, 1979] Comer, D. (1979). Ubiquitous b-tree. *ACM Comput. Surv.*, 11(2):121–137.

[Conrad and Schriber, 2004] Conrad, J. G. and Schriber, C. P. (2004). Constructing a text corpus for inexact duplicate detection. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 582–583, New York, NY, USA. ACM.

[Cordeiro and Brazdil, 2004] Cordeiro, J. and Brazdil, P. (2004). Learning text extraction rules, without ignoring stop words. In *PRIS*, pages 128–138.

[Craven et al., 2000] Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., and Slattery, S. (2000). Learning to construct knowledge bases from the world wide web. *Artificial Intelligende*, 118:69–113.

[Croft, 2000] Croft, W. B. (2000). *Combining approaches to information retrieval*. Kluwer.

[Davenport and Prusak, 1998] Davenport, T. H. and Prusak, L. (1998). *Working Knowledge: How Organizations Manage What They Know*. Harvard Business School Press, Boston.

[Deerwester et al., 1989] Deerwester, S. C., Dumais, S. T., Furnas, G. W., Harshman, R. A., Landauer, T. K., Lochbaum, K. E., and Streeter, L. A. (1989). Computer information retrieval using latent semantic structure.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum-likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society B*, 39:1–38.

[DeShon and Alexander, 1996] DeShon, R. P. and Alexander, R. A. (1996). Alternative procedures for testing regression slope homogeneity when group error variances are unequal. *Psychological Methods*, 1:261–277.

[Dias et al., 2007] Dias, G., Alves, E., and Lopes, J. G. P. (2007). Topic segmentation algorithms for text summarization and passage retrieval: an exhaustive evaluation. In *AAAI'07: Proceedings of the 22nd national conference on Artificial intelligence*, pages 1334–1339. AAAI Press.

[Drucker, 1969] Drucker, P. (1969). *The Age of Discontinuity; Guidelines to our changing society*. Harper and Row, New York.

[Dunning, 1994] Dunning, T. (1994). Statistical identification of language. Technical report, Computing Research Lab (CRL), New Mexico State University.

[Eckart and Young, 1936] Eckart, C. and Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218.

[Ester et al., 1998] Ester, M., Kriegel, H.-P., Sander, J., Wimmer, M., and Xu, X. (1998). Incremental clustering for mining in a data warehousing environment. In *In Proc. 24th Int. Conf. Very Large Data Bases, VLDB*, pages 323–333.

[Ester et al., 1996] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231.

[Everitt et al., 2009] Everitt, B. S., Landau, S., and Leese, M. (2009). *Cluster Analysis*. Wiley Publishing.

[Forgy, 1965] Forgy, E. (1965). Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics*, 21:768–780.

[Fraley and Raftery, 1998] Fraley, C. and Raftery, A. E. (1998). How many clusters? which clustering method? answers via model-based cluster analysis. *The Computer Journal*, 41(8):578–588.

[Fruchterman and Reingold, 1991] Fruchterman, T. M. J. and Reingold, E. M. (1991). Graph drawing by force-directed placement. *Softw. Pract. Exper.*, 21(11):1129–1164.

[Gan and Wu, 2004] Gan, G. and Wu, J. (2004). Subspace clustering for high dimensional categorical data. *SIGKDD Explor. Newsl.*, 6(2):87–94.

[Ghosh and Strehl, 2006] Ghosh, J. and Strehl, A. (2006). Similarity-based text clustering: A comparative study. *Grouping Multidimensional Data*, pages 79–97.

[Glenisson et al., 2005] Glenisson, P., Glänzel, W., Janssens, F., and De Moor, B. (2005). Combining full text and bibliometric information in mapping scientific disciplines. *Inf. Process. Manage.*, 41(6):1548–1572.

[Graepel, 1998] Graepel, T. (1998). Statistical physics of clustering algorithms. Technical report, FB PHYSIK, INSTITUT FR THEORETISHE PHYSIK.

[Green et al., 1989] Green, P. E., Carmone, F. J., and Smith, S. M. (1989). *Multidimensional Scaling: Concepts and Applications*. Allyn & Bacon.

[Grimm et al., 2007] Grimm, S., Hitzler, P., and Abecker, A. (2007). Knowledge representation and ontologies. In *Semantic Web Services: Concepts, Technologies, and Applications*, chapter 3, pages 51–105. Springer-Verlag.

[Guha et al., 1998] Guha, S., Rastogi, R., and Shim, K. (1998). Cure: an efficient clustering algorithm for large databases. In *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 73–84, New York, NY, USA. ACM.

[Gulli and Signorini, 2005] Gulli, A. and Signorini, A. (2005). Building an open source meta-search engine. In *WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1004–1005, New York, NY, USA. ACM Press.

[Gutenberg, 1992] Gutenberg (1992). Project Gutenberg. Web Site. http://www.gutenberg.org. Founded by M. Hart.

[Hahn and Subramani, 2000] Hahn, J. and Subramani, M. R. (2000). A framework of knowledge management systems: Issues and challenges for theory and practice. In *in Proceedings of ICIS 2000*, pages 302–312.

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

[Hamerly and Elkan, 2002] Hamerly, G. and Elkan, C. (2002). Alternatives to the k-means algorithm that find better clusterings. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 600–607, New York, NY, USA. ACM.

[Han, 2005] Han, J. (2005). *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[Han et al., 2001] Han, J., Kamber, M., and Tung, A. K. H. (2001). *Spatial Clustering Methods in Data Mining: A Survey*, pages 1–29. Taylor and Francis.

[Hartigan, 1975] Hartigan, J. (1975). *Clustering Algorithms*. John Wiley and Sons, New York.

[Hartigan, 1972] Hartigan, J. A. (1972). Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337):123–129.

[Hearst, 1997] Hearst, M. A. (1997). Texttiling: segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.

[Heintze, 1996] Heintze, N. (1996). Scalable document fingerprinting. In *In Proc. USENIX Workshop on Electronic Commerce*.

[Hersh et al., 1994] Hersh, W., Buckley, C., Leone, T. J., and Hickam, D. (1994). Ohsumed: an interactive retrieval evaluation and new large test collection for research. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 192–201, New York, NY, USA. Springer-Verlag New York, Inc.

[Hill and Lewicki, 2006] Hill, T. and Lewicki, P. (2006). *Statistics: Methods and Applications*. StatSoft, Inc.

[Jaccard, 1901] Jaccard, P. (1901). Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579.

[Jain and Dubes, 1988] Jain, A. K. and Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[Jain et al., 1999] Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: a review. *ACM Computer Survey*, 31(3):264–323.

[Jancey, 1966] Jancey, R. C. (1966). Multidimensional group analysis. *Australian Journal of Botany*, 14(1):127–130.

[Jing et al., 2009] Jing, L., Ng, M., and Huang, J. (2009). Knowledge-based vector space model for text clustering. *Knowledge and Information Systems*.

[Kailing et al., 2004] Kailing, K., Kriegel, H.-P., and Kroeger, P. (2004). Density-connected subspace clustering for high-dimensional data. In *In: Proceedings of SDM 2004*, pages 246–257.

[Kathleen and Mckeown, 2003] Kathleen, M. G. and Mckeown, K. (2003). Discourse segmentation of multi-party conversation. In *in 41st Annual Meeting of ACL*, pages 562–569.

[Kaufman and Rousseeuw, 1990] Kaufman, L. and Rousseeuw, P. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley Interscience, New York.

[Khmaladze, 1988] Khmaladze, E. (1988). The statistical analysis of large number of rare events. Technical report, Department of Mathematical Statistics, CWI.

[Knuth, 1973] Knuth, D. E. (1973). *Fundamental Algorithms*, volume 1 of *The Art of Computer Programming*. Addison-Wesley, Reading, Massachusetts, second edition.

[Korenius et al., 2007] Korenius, T., Laurikkala, J., and Juhola, M. (2007). On principal component analysis, cosine and euclidean measures in information retrieval. *Information Sciences Sciences*, 177(22):4893–4905.

[Kriegel et al., 2005] Kriegel, H.-P., Kroger, P., Renz, M., and Wurst, S. (2005). A generic framework for efficient subspace clustering of high-dimensional data. In *ICDM '05: Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 250–257, Washington, DC, USA. IEEE Computer Society.

[Kriegel et al., 2009] Kriegel, H.-P., Kröger, P., and Zimek, A. (2009). Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Trans. Knowl. Discov. Data*, 3(1):1–58.

[Kroeger, 2005] Kroeger, P. R. (2005). *Analyzing Grammar*. Cambridge University Press, New York.

[Kulp and Kontostathis, 2007] Kulp, S. and Kontostathis, A. (2007). Improving search and retrieval performance through shortening documents and pruning out jargon. Technical report, Ursinus College, Department of Computer Science.

[Lamprier et al., 2008] Lamprier, S., Amghar, T., Levrat, B., and Saubion, F. (2008). Toward a more global and coherent segmentation of texts. *Appl. Artif. Intell.*, 22(3):208–234.

[Larkey, 1999] Larkey, L. S. (1999). A patent search and classification system. In *DL '99: Proceedings of the fourth ACM conference on Digital libraries*, pages 179–187, New York, NY, USA. ACM.

[Lengnick-Hall and Lengnick-Hall, 2002] Lengnick-Hall, M. L. and Lengnick-Hall, C. A. (2002). *Human Resource Management in the Knowledge Economy: New Challenges, New Roles, New Capabilities*. Berrett-Koehler Publishers.

[Lewis et al., 2004] Lewis, D., Yang, Y., Rose, T., and Li, F. (2004). Rcv1: A new benchmark collection for text categorization research.

[Liu et al., 2007a] Liu, G., Li, J., Sim, K., and Wong, L. (2007a). Distance based subspace clustering with flexible dimension partitioning. *Data Engineering, International Conference on*, 0:1250–1254.

[Liu et al., 2007b] Liu, Y., Fu, Y., Zhang, M., Ma, S., and Ru, L. (2007b). Automatic search engine performance evaluation with click-through data analysis. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 1133–1134, New York, NY, USA. ACM.

[Lu et al., 2004] Lu, Y., Lu, S., Fotouhi, F., Deng, Y., and Brown, S. J. (2004). Incremental genetic k-means algorithm and its application in gene expression data analysis. *BMC Bioinformatics*, 5(172).

[Lundvall and Johnson, 1994] Lundvall, B. and Johnson, B. (1994). The learning economy. *Journal of Industry Studies*, 1(2):23–42.

[MacQueen, 1967] MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In Cam, L. M. L. and Neyman, J., editors, *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press.

[Madeira and Oliveira, 2004] Madeira, S. C. and Oliveira, A. L. (2004). Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1:24–45.

[Malioutov and Barzilay, 2006] Malioutov, I. and Barzilay, R. (2006). Minimum cut model for spoken lecture segmentation. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 25–32. Association for Computational Linguistics.

[Mandelbrot, 1961] Mandelbrot, B. (1961). *The structure of language and its mathematical aspects*, chapter On the theory of word frequencies and on related Markovian models of discourse, pages 129–190. American Mathematical Society.

[Manning et al., 2008] Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press, 1 edition.

[Manning and Schtze, 1999] Manning, C. D. and Schtze, H. (1999). *Foundations of Statistical Natural Language Processing*. The MIT Press.

[McGill, 1979] McGill, M. (1979). An evaluation of factors affecting document ranking by information retrieval systems. Technical report, Syracuse University, School of Information Studies.

[Mcinnis et al., 1990] Mcinnis, M. L., Larson, L. L., and Vavra, M. (1990). Classifying herbivore diets using hierarchical cluster analysis. *Journal of Range Management*, 43(3):271–274.

[McRae, 1971] McRae, D. J. (1971). Mikca: A fortran iv iterative kmeans cluster analysis program. *Behavioral Science*, 16(4):423–424.

[Menzel, 1995] Menzel, W. (1995). Robust processing of natural language. In *KI '95: Proceedings of the 19th Annual German Conference on Artificial Intelligence*, pages 19–34, London, UK. Springer-Verlag.

[Mercer, 2003] Mercer, V. P. (2003). Clustering large datasets. Technical report, Linacre College.

[Milenova and Campos, 2002] Milenova, B. L. and Campos, M. M. (2002). O-cluster: Scalable clustering of large high dimensional data sets. In *ICDM '02: Proceedings of the 2002 IEEE International Conference on Data Mining*, page 290, Washington, DC, USA. IEEE Computer Society.

[Miller et al., 1990] Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. J. (1990). Introduction to wordnet: An on-line lexical database. *Journal of Lexicography*, 3(4):235–244.

[Milligan, 1981] Milligan, G. W. (1981). A monte carlo study of thirty internal criterion measures for cluster analysis. *Psychometrika*, 46(2):187–199.

[Mladenic and Grobelnik, 1999] Mladenic, D. and Grobelnik, M. (1999). Feature selection for unbalanced class distribution and naive bayes. In *In Proceedings of the 16th International Conference on Machine Learning (ICML*, pages 258–267. Morgan Kaufmann Publishers.

[Moise et al., 2008] Moise, G., Sander, J., and Ester, M. (2008). Robust projected clustering. *Knowl. Inf. Syst.*, 14(3):273–298.

[Morris and Hirst, 1991] Morris, J. and Hirst, G. (1991). Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1):21–48.

[Moyotl-Hernández and Héctor, 2004] Moyotl-Hernández, E. and Héctor, J.-S. (2004). An analysis on frequency of terms for text categorization. In para el Procesamiento del Lenguaje Natura, S. E., editor, *Congreso de la Sociedad Espaola para el Procesamiento del Lenguaje Natural No20, Barcelona , ESPAGNE*, number 33 in Procesamiento del lenguaje natura, pages 141–146.

[Newman, 2005] Newman, M. E. J. (2005). Power laws, pareto distributions and zipf's law. *Contemporary Physics*, 46.

[Ng et al., 2001] Ng, A. Y., Jordan, M. I., and Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, pages 849–856. MIT Press.

[Ng and Han, 1994] Ng, R. T. and Han, J. (1994). Efficient and effective clustering methods for spatial data mining. In *VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases*, pages 144–155, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Nguyen and Rayward Smith, 2008] Nguyen, Q. H. and Rayward Smith, V. J. (2008). Internal quality measures for clustering in metric spaces. *International Journal of Business Intelligence and Data Mining*, 3(1):4–29.

[Ning et al., 2010] Ning, H., Xu, W., Chi, Y., Gong, Y., and Huang, T. S. (2010). Incremental spectral clustering by efficiently updating the eigen-system. *Pattern Recognition*, 43(1):113–127.

[Nonaka and Takeuchi, 1995] Nonaka, I. and Takeuchi, H. (1995). *The knowledge-creating company: How japanese companies create the dynamics of innovation.* Oxford University Press, New York.

[Ogilvie and Callan, 2003] Ogilvie, P. and Callan, J. (2003). Combining document representations for known-item search. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 143–150, New York, NY, USA. ACM.

[Olui-Vukovic, 2001] Olui-Vukovic, V. (2001). From information to knowledge: some reflections on the origin of the current shifting towards knowledge processing and further perspective. *J. Am. Soc. Inf. Sci. Technol.*, 52(1):54–61.

[Orengo and Huyck, 2001] Orengo, V. M. and Huyck, C. (2001). A Stemming Algorithm for Portuguese Language. In *Proceedings of Eigth Symposium on String Processing and Information Retrieval (SPIRE 2001) - Chile*, pages 186–193.

[Otoo et al., 2001] Otoo, E. J., Shoshani, A., and Hwang, S.-W. (2001). Clustering high dimensional massive scientific datasets. *J. Intell. Inf. Syst.*, 17:147–168.

[Parsons et al., 2004] Parsons, L., Haque, E., and Liu, H. (2004). Subspace clustering for high dimensional data: a review. *SIGKDD Explor. Newsl.*, 6(1):90–105.

[Patrikainen and Mannila, 2004] Patrikainen, A. and Mannila, H. (2004). Subspace clustering of high-dimensional binary data - a probabilistic approach. In *In Workshop on Clustering High Dimensional Data and its Applications, SIAM International Conference on Data Mining*, pages 57–65.

[Pei et al., 2003] Pei, J., Zhang, X., Cho, M., Wang, H., and Yu, P. S. (2003). Maple: A fast algorithm for maximal pattern-based clustering. In *ICDM '03: Proceedings of the Third IEEE International Conference on Data Mining*, page 259, Washington, DC, USA. IEEE Computer Society.

[Peirce, 1885] Peirce, C. S. (1885). On the algebra of logic. *American Journal of Mathematics*, 7(3):197–202.

[Piwowarski et al., 2010] Piwowarski, B., Frommholz, I., Lalmas, M., and van Rijsbergen, K. (2010). Exploring a multidimensional representation of documents and queries. In *Proceedings of the 9th RIAO Conference (RIAO 2010)*.

[Plato. and Levett, 1928] Plato. and Levett, M. J. (1928). *The Theaetetus of Plato / translated by M.J. Levett.* Jackson, Wylie & Co., Glasgow :.

[Platzer, 2008] Platzer, M. D. (2008). Driving the global knowledge economy. Technical report, SIIA.

[Polyvyanyy and Kuropka, 2007] Polyvyanyy, A. and Kuropka, D. (2007). *A quantitative evaluation of the enhanced topic-based vector space model.* Universitat Potsdam.

[Ponte and Croft, 1998] Ponte, J. M. and Croft, W. B. (1998). A language modeling approach to information retrieval. In *Research and Development in Information Retrieval*, pages 275–281.

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

[Procopiuc et al., 2002] Procopiuc, C. M., Jones, M., Agarwal, P. K., and Murali, T. M. (2002). A monte carlo algorithm for fast projective clustering. In *SIGMOD '02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 418–427, New York, NY, USA. ACM.

[Pugh and Henzinger, 2003] Pugh, W. and Henzinger, M. (2003). Detecting duplicate and near-duplicate files (united states patent 6,658,423)'. Assignee: Google patent, United States Patent.

[Puzicha et al., 1999] Puzicha, J., Buhmann, J. M., Rubner, Y., and Tomasi, C. (1999). Empirical evaluation of dissimilarity measures for color and texture. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 2:1165–1172.

[Qian et al., 2004] Qian, G., Sural, S., Gu, Y., and Pramanik, S. (2004). Similarity between euclidean and cosine angle distance for nearest neighbor queries. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 1232–1237. ACM.

[Ramakrishnan, 2004] Ramakrishnan, S. (2004). A fast, coarse filtering method for database lookup of peptide fragmentation spectra. Technical Report CH391L, University of Texas at Austin.

[Rand, 1971] Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850.

[Reynar, 1999] Reynar, J. C. (1999). Statistical models for topic segmentation. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 357–364, Morristown, NJ, USA. Association for Computational Linguistics.

[reza Amini et al., 2000] reza Amini, M., Zaragoza, H., and Gallinari, P. (2000). Learning for sequence extraction tasks. In *Content-Based Multimedia Information Access, RIAO2000*, pages 476–490.

[Robertson et al., 1992] Robertson, S. E., Walker, S., Beaulieu, M. H., Gull, A., and Lau, M. (1992). Okapi at trec. In *Text REtrieval Conference*, pages 21–30.

[Rosario, 2000] Rosario, B. (2000). Latent semantic indexing: An overview. Technical report, University of California, Berkeley.

[Rousseeuw, 1987] Rousseeuw, P. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20(1):53–65.

[Rowley, 2007] Rowley, J. (2007). The wisdom hierarchy: representations of the dikw hierarchy. *Journal of Information Science*, 33(2).

[Sarwar et al., 2001] Sarwar, B. M., Karypis, G., Konstan, J. A., and Reidl, J. (2001). Item-based collaborative filtering recommendation algorithms. In

Proceedings of the 10th international conference on World Wide Web, pages 285–295.

[Schaeffer, 2007] Schaeffer, S. (2007). Graph clustering. *Computer Science Review*, 1(1):27–64.

[Sheikholeslami et al., 2000] Sheikholeslami, G., Chatterjee, S., and Zhang, A. (2000). Wavecluster: a wavelet-based clustering approach for spatial data in very large databases. *The VLDB Journal*, 8(3-4):289–304.

[Shi and Malik, 1997] Shi, J. and Malik, J. (1997). Normalized cuts and image segmentation. In *CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, page 731, Washington, DC, USA. IEEE Computer Society.

[Silva et al., 2004] Silva, I. R., ao Nunes Souza, J., and Santos, K. S. (2004). Dependence among terms in vector space model. *Database Engineering and Applications Symposium, International*, 0:97–102.

[Sinka and Corne, 2005] Sinka, M. P. and Corne, D. W. (2005). The banksearch web document dataset: investigating unsupervised clustering and category similarity. *J. Netw. Comput. Appl.*, 28(2):129–146.

[Sitbon and Bellot, 2007] Sitbon, L. and Bellot, P. (2007). Topic segmentation using weighted lexical links (wll). In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 737–738, New York, NY, USA. ACM.

[Skala and Hrdek, 2009] Skala, V. and Hrdek, J. (2009). Efficient hash function for duplicate elimination in dictionaries. In *Algoritmy 2009 18th Conference on Scientific Computing*, pages 382–391, Bratislava. Slovensk technick univerzita.

[Skillicorn, 2007] Skillicorn, D. B. (2007). *Understanding Complex Datasets: Data Mining with Matrix Decompositions.* Chapman & Hall,CRC.

[Sneath and Sokal, 1973] Sneath, P. H. A. and Sokal, R. R. (1973). *Numerical taxonomy: The principles and practice of numerical classification.* W.H. Freeman, San Francisco.

[Snijders et al., 1990] Snijders, T., Dormaar, M., Schuur, W., Dijkman-Caes, C., and Driessen, G. (1990). Distribution of some similarity coefficients for dyadic binary data in the case of associated attributes. *Journal of Classification*, 7(1):5–31.

[Sproat and Emerson, 2003] Sproat, R. and Emerson, T. (2003). The first international chinese word segmentation bakeoff. In *Proceedings of the second*

SIGHAN workshop on Chinese language processing - Volume 17, pages 133–143, Morristown, NJ, USA. Association for Computational Linguistics.

[Stokes, 2003] Stokes, N. (2003). Spoken and written news story segmentation using lexical chains. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 49–54, Morristown, NJ, USA. Association for Computational Linguistics.

[Sydänmaanlakka, 2000] Sydänmaanlakka, P. (2000). Understanding organizational learning through knowledge management, competence management, and performance management. In *Prooceedings of 2nd Annual Knowledge Management and Organizational Learning Conference.*, pages 329–341. London: Linkage International.

[Trefethen and Bau, 1997] Trefethen, L. N. and Bau, D. (1997). *Numerical Linear Algebra*. SIAM: Society for Industrial and Applied Mathematics.

[Tukey, 1977] Tukey, J. W. (1977). *Exploratory Data Analysis*. Addison-Wesley.

[Turney and Pantel, 2010] Turney, P. D. and Pantel, P. (2010). From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

[Ukkonen, 1985] Ukkonen, E. (1985). Algorithms for approximate string matching. *Information and Control*, 64(1-3):100–118.

[Utiyama and Isahara, 2001] Utiyama, M. and Isahara, H. (2001). A statistical model for domain-independent text segmentation. In *In Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics*, pages 491–498.

[Verhaegen et al., 2009] Verhaegen, P.-A., D'hondt, J., Vertommen, J., Dewulf, S., and Duflou, J. (2009). Relating properties and functions from patents to triz trends. *CIRP Journal of Manufacturing Science and Technology*, 1(3):126 – 130. Design Synthesis.

[Vertommen et al., 2008] Vertommen, J., Janssens, F., Moor, B. D., and Duflou, J. R. (2008). Multiple-vector user profiles in support of knowledge sharing. *Information Sciences*, 178(17):3333–3346.

[von Luxburg, 2007] von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416.

[Wall et al., 2003] Wall, M. E., Rechtsteiner, A., and Rocha, L. M. (2003). *Singular Value Decomposition and Principal Component Analysis*, chapter 5, pages 91–109. Kluwel, Norwell, MA.

[Wang et al., 2002] Wang, H., Wang, W., Yang, J., and Yu, P. S. (2002). Clustering by pattern similarity in large data sets. In Franklin, M. J., Moon, B., and Ailamaki, A., editors, *SIGMOD Conference*, pages 394–405. ACM.

[Wang et al., 1997] Wang, W., Yang, J., and Muntz, R. R. (1997). Sting: A statistical information grid approach to spatial data mining. In *VLDB '97: Proceedings of the 23rd International Conference on Very Large Data Bases*, pages 186–195, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Waterson and Preece, 1999] Waterson, A. and Preece, A. (April 1999). Verifying ontological commitment in knowledge-based systems. *Knowledge-Based Systems*, 12:45–54(10).

[Wenger, 1999] Wenger, E. (1999). *Communities of Practice: Learning, Meaning, and Identity*. Cambridge University Press.

[Wishart, 1969] Wishart, D. (1969). Fortran ii programs for 8 methods of cluster analysis (clustan i). Computer contribution 38, Kansas Geological Survey, University of Kansas, Lawrence, KS, USA.

[Wolfe, 1970] Wolfe, J. H. (1970). Pattern clustering by multivariate mixture analysis. *Multivariate Behavioral Research*, 5:329–350.

[Wong et al., 1985] Wong, S. K. M., Ziarki, W., and Wong, P. C. N. (1985). Generalized vector space model in information retrieval. In *SIGIR '85*. ACM-SIGIR, ACM.

[Woo et al., 2004] Woo, K.-G., Lee, J.-H., Kim, M.-H., and Lee, Y.-J. (2004). Findit: a fast and intelligent subspace clustering algorithm using dimension voting. *Information and Software Technology*, 46(4):255–271.

[Wu et al., 2009a] Wu, J., Xiong, H., and Chen, J. (2009a). Adapting the right measures for k-means clustering. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 877–886, New York, NY, USA. ACM.

[Wu et al., 2009b] Wu, J., Xiong, H., and Chen, J. (2009b). Towards understanding hierarchical clustering: A data distribution perspective. *Neurocomputing*, 72(10-12):2319 – 2330. Lattice Computing and Natural Computing (JCIS 2007) / Neural Networks in Intelligent Systems Designn (ISDA 2007).

[Wu and Zhou, 2002] Wu, J. and Zhou, Z.-H. (2002). Face recognition with one training image per person. *Pattern Recogn. Lett.*, 23(14):1711–1719.

[Yager and Filev, 1992] Yager, R. and Filev, D. (1992). Approximate clustering via the mountain method. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(8):1279–1284.

[Yang and Callan, 2005] Yang, H. and Callan, J. (2005). Near-duplicate detection for erulemaking. In *dg.o 2005: Proceedings of the 2005 national conference on Digital government research*, pages 78–86. Digital Government Society of North America.

[Yang et al., 2002] Yang, J., Wang, W., Wang, H., and Yu, P. (2002). Delta-clusters: Capturing subspace correlation in a large data set. *Data Engineering, International Conference on*, 0:0517.

[yen Kan et al., 1998] yen Kan, M., Klavans, J. L., and Mckeown, K. R. (1998). Linear segmentation and segment significance. In *In Proceedings of the 6th International Workshop on Very Large Corpora*, pages 197–205.

[Yiu and Mamoulis, 2003] Yiu, M. L. and Mamoulis, N. (2003). Frequent-pattern based iterative projected clustering. *Data Mining, IEEE International Conference on*, 0:689.

[Zaane et al., 2002] Zaane, O. R., Foss, A., hoon Lee, C., and Wang, W. (2002). On data clustering analysis: Scalability, constraints and validation. In *In Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD*, pages 28–39.

[Zhang et al., 2002] Zhang, M., Song, R., Lin, C., Ma, S., Jiang, Z., Jin, Y., Liu, Y., and Zhao, L. (2002). Thu trec2002 web track experiments. In *In Proceedings of the Eleventh Text Retrieval Conference, TREC*, pages 586–594.

[Zhang and Couloigner, 2005] Zhang, Q. and Couloigner, I. (2005). A new and efficient k-medoid algorithm for spatial clustering. *Computational Science and Its Applications ICCSA 2005*, pages 181–189.

[Zhang et al., 1996] Zhang, T., Ramakrishnan, R., and Livny, M. (1996). Birch: an efficient data clustering method for very large databases. *SIGMOD Rec.*, 25(2):103–114.

[Zhang et al., 1997] Zhang, T., Ramakrishnan, R., and Livny, M. (1997). Birch: A new data clustering algorithm and its applications. *Data Min. Knowl. Discov.*, 1(2):141–182.

[Zhao and Karypis, 2002] Zhao, Y. and Karypis, G. (2002). Evaluation of hierarchical clustering algorithms for document datasets. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 515–524, New York, NY, USA. ACM.

[Zipf, 1949] Zipf, G. K. (1949). *Human Behavior and the Principle of Least Effort.* Addison-Wesley (Reading MA).

Appendix A

Clustering: Datasets Descriptions

In this appendix, the different test collection are described which are used in several experiments presented in this dissertation. All the datasets or the derived test collections are standardized datasets for which a reference structure is available:

- Ohsumed: this dataset is a subset of a MEDLINE collection, which consists of 348566 references covering 270 medical journals collected over a five-year period, from 1987 to 1991.
- Reuters-RCV1: the Reuters datasets is a collection of Reuters press articles, written in the English language, covering articles that appeared in the one-year period, starting from 20th of August, 1996. In total, this collection contains 810000 articles.
- 20 Newsgroup: 20 different newsgroup topics constitute this dataset. It contains approximately 20000 newsgroup documents, wherein every newsgroup more or less contains an equal amount of documents.
- Banksearch: This dataset is composed of 11 different categories, containing on average 11000 web pages.

A.1 Ohsumed based test collections

A.1.1 Ohsumed 1

From the standardized Ohsumed dataset, a first Ohsumed test collections was constructed. The documents were selected from the 20 categories listed in Table

Table A.1: Overview of the selected OHSUMED topics

OHSU 1	OHSU 18	OHSU 29	OHSU 37	OHSU 48
OHSU 3	OHSU 19	OHSU 30	OHSU 38	OHSU 49
OHSU 5	OHSU 21	OHSU 32	OHSU 40	OHSU 51
OHSU 11	OHSU 23	OHSU 33	OHSU 42	OHSU 55
OHSU 13	OHSU 24	OHSU 35	OHSU 44	OHSU 63
OHSU 14	OHSU 26	OHSU 36	OHSU 47	

A.1. These topics were chosen randomly, without knowledge about their actual content. From each topic, 15 documents were selected, and 29 subsets were constructed as follows:

- Subset 1 contains 30 documents covering two topics, OHSU 1 & OHSU 3
- Subset 2 contains 45 documents covering three topics, OHSU 1, OHSU 3 & OHSU 5
- ...
- Subset 29 contains 450 documents covering all topics

Table A.2 provides an overview of the dimensionalities of the different subsets.

A.1.2 Ohsumed 2

A second test collection was constructed based on the standardized Ohsumed dataset, wherein every test set describes 4 topics containing an equal number of documents. The following 4 topics were randomly chosen from the Ohsumed description:

- OHSU 3
- OHSU 11
- OHSU 30
- OHSU 33

From these topics, 20 subsets were constructed as follows:

- Subset 1 contains 200 documents, 50 from each of the above topics
- Subset 2 contains 400 documents, 100 from each of the above topics

- ...

- Subset 20 contains 4000 documents, 1000 from each of the above topics

For every test set, the properties is described in Table A.3.

Table A.2: Detailed description of the Ohsumed 1 test collection

Test set ID	# of topics	# of docs	# of terms
1	2	30	645
2	3	45	852
3	4	60	1138
4	5	75	1368
5	6	90	1524
6	7	105	1724
7	8	120	1862
8	9	135	2026
9	10	150	2168
10	11	165	2288
11	12	180	2409
12	13	195	2575
13	14	210	2777
14	15	225	2906
15	16	240	3050
16	17	255	3150
17	18	270	3244
18	19	285	3367
19	20	300	3458
20	21	315	3602
21	22	330	3700
22	23	345	3781
23	24	360	3884
24	25	375	3967
25	26	390	4063
26	27	405	4147
27	28	420	4231
28	29	435	4332
29	30	450	4439

Table A.3: Detailed description of the Ohsumed 2 test collection

Test set ID	# of topics	# of docs	# of terms
1	4	200	4120
2	4	400	6210
3	4	600	7810
4	4	800	9167
5	4	1000	10286
6	4	1200	11262
7	4	1400	12202
8	4	1600	13034
9	4	1800	13875
10	4	2000	14557
11	4	2200	15218
12	4	2400	15940
13	4	2600	16551
14	4	2800	17168
15	4	3000	17726
16	4	3200	18255
17	4	3400	18723
18	4	3600	19231
19	4	3800	19664
20	4	4000	20104

A.2 Reuters based test collections

A.2.1 Reuters 1

Similar constructed as the first Ohsumed test collection, a test collection Reuters1 is constructed from the standardized Reuters dataset. This test set consists of documents selected from the Reuters topics given in Table A.4.

From each topic, 15 documents were selected, and 18 subsets were constructed as follows:

- Subset 1 contains 30 documents covering two topics, 'Defense' and 'Disasters & accidents'
- Subset 2 contains 45 documents covering three topics
- ...
- Subset 18 contains 285 documents covering all topics

Table A.5 provides an overview of the dimensionalities of the different subsets.

Table A.4: Overview of the selected Reuters topics

Defense	Human Interest
Disasters & Accidents	Religion
Arts, Culture & Entertainment	Science & Technology
Environment & Natural World	Sports
Fashion	Travel & Tourism
War & Civil War	Weather
Crime & Law Enforcement	International Relations
Health	Labor issues
Domestic Politics	Personalities & People
Welfare & Social Services	

Table A.5: Detailed description of the Reuters 1 test collection

Test set ID	# of topics	# of docs	# of terms
1	2	30	2736
2	3	45	3921
3	4	60	4616
4	5	75	5272
5	6	90	5990
6	7	105	6541
7	8	120	7086
8	9	135	7648
9	10	150	8249
10	11	165	8987
11	12	180	9488
12	13	195	9967
13	14	210	10447
14	15	225	12255
15	16	240	12685
16	17	255	13184
17	18	270	13633
18	19	285	13897

A.2.2 Reuters 2

This test set was constructed in the same manner as Ohsumed 2. This test set contains single-topic documents, selected from 4 topics of the Reuters RCV1

dataset.

These four topics are listed in Table A.6.

Table A.6: The topics used for the Reuters 2 test collection

C13	C31
C24	C42

For every test set, the properties is described in Table A.7.

Table A.7: Detailed description of the Reuters 2 test collection

Test set ID	# of topics	# of docs	# of terms
1	4	200	8308
2	4	400	11912
3	4	600	14801
4	4	800	17051
5	4	1000	19406
6	4	1200	21480
7	4	1400	23057
8	4	1600	24711
9	4	1800	26312
10	4	2000	27759
11	4	2200	28922
12	4	2400	29968
13	4	2600	30909
14	4	2800	31936
15	4	3000	32855
16	4	3200	33772
17	4	3400	34639
18	4	3600	35393
19	4	3800	36153
20	4	4000	36890

A.3 Banksearch

This multilevel dataset consists of the topics given in Table A.8. Several combination of these topics resulted in 21 test sets. For every test set a random number of topics and documents were selected.

For every test set, the properties is described in Table A.9.

Table A.8: Overview of the used Banksearch categories

Commercial banks	Building societies
Insurance agencies	Java
C/C++	Visual Basic
Astronomy	Biology
Soccer	Motor sports
Sport	

Table A.9: Detailed description of the Banksearch test collection

Test set ID	# of topics	# of docs	# of terms
Banksearch1	10	1257	16108
Banksearch2	5	490	21014
Banksearch3	4	305	36326
Banksearch4	4	331	17346
Banksearch5	5	268	23450
Banksearch6	5	245	15317
Banksearch7	2	151	12953
Banksearch8	10	430	18942
Banksearch9	3	226	17531
Banksearch10	4	197	18660
Banksearch11	4	218	9739
Banksearch12	4	256	8643
Banksearch13	5	386	14743
Banksearch14	2	165	9373
Banksearch15	3	346	9029
Banksearch16	5	592	13395
Banksearch17	10	927	17136
Banksearch18	4	309	9086
Banksearch19	5	437	7615
Banksearch20	3	281	7388
Banksearch21	9	2124	10679

A.4 20Newsgroup

The Newsgroup 20 dataset is a collection of approximately 20000 newsgroup articles, equally divided over 20 newsgroup items. The topics are described in Table A.10.

Table A.10: Overview of the topics present in the 20Newsgroup

comp.graphics	rec.autos
comp.os.ms-windows.misc	sci.crypt
comp.sys.ibm.pc.hardware	sci.med
comp.sys.mac.hardware	sci.space
comp.windows.x	misc.forsale
rec.motorcycles	talk.politics.misc
rec.sport.baseball	talk.politics.guns
rec.sport.hockey	talk.politics.mideast
sci.electronics	talk.religion.misc
alt.atheism	soc.religion.christia

A.5 TREC based datasets

In Table A.11, the properties of several datasets of different tracks of the TREC conferences are listed. These datasets were used in the validation of the Zipf weighting scheme. The content of these datasets is listed below:

- la1 and la2, two datasets used for the TREC-5 conference, contains articles of the Los Angeles Times. The topics of these articles are financial, domestic, international, sports, public transport and entertainment.
- re0 and re1 are two different subsets from Reuters-21578 text, each with a standardized structure.
- the different datasets k1a, k1b, and Wap originate from the WebACE project. Each of these datasets contain web pages that are assigned to specific topics in the subject hierarchy of the Yahoo search engine.
- Tr31 and Tr41 are datasets constructed for TREC-5, TREC-6 and TREC-7 conference and originate from various datasets. The underlying structure of these datasets is based on queries applied on these datasets.

Table A.11: Detailed description of the TREC related datasets

Data	Source	# of documents	# of terms	# of classes
la1	LA Times (TREC-5)	6279	21604	6
la2	LA Times (TREC-5)	6279	21604	6
re0	Reuters-21578	1504	2886	13
re1	Reuters-21578	1657	3758	25
k1a	WebACE	2340	13879	20
k1b	WebACE	2340	13879	6
Wap	WebACE	2340	13879	6
Tr31	TREC-5 & TREC-6	927	10128	7
Tr41	TREC-5 & TREC-6	878	7454	10

Appendix A

Appendix B

Language identification process

B.1 Stopwords for language identification

Dutch	French	English	German
aan	mais	about	aus
dat	aux	after	auf
de	les	all	bei
deze	par	and	bis
die	pour	as	das
dit	tout	at	der
door	tous	by	ein

Appendix B

Appendix C

Advances in Clustering

C.1 Incremental clustering

In this section, the algorithmic descriptions of three operations used in the incremental clustering algorithm are provided. In these descriptions, the following notation conventions are used:

- The comments in the code are in italic font, and proceeded with the '%'sign.
- The variables in these comments are marked with '' (e.g. 'variable').
- The use of a function 'function-name' with parameter 'parm' is denoted with function-name(parm).
- The index 'i' in a structure 'array' is indicated as array(i).
- The concatenation of items 'items' to a structure 'array' is denoted as array = [array items]

In the following sections, the insertion operation of a document in a leaf or non-leaf node are discussed. The last section describes the splitting procedure which is applied when the number of child nodes exceeds a predefined parameter.

C.1.1 Insertion operation for non-leaf node

The following algorithm describes the insertion of a new document *doc* in a non-leaf node i. The first step in the process is a pairwise comparison with the leaf child nodes of the node i. If the highest obtained similarity is above threshold $s2$, the document *doc* will be inserted in one of the leaf node. If not, a similar approach is performed with the non-leaf child nodes.

Node(i) = **InsertionNonLeaf**(Node(i),doc)

```
S ⇐ pairwiseSimmilarity(doc,Child(i)) % S array with similarities
(maxSim,j) ⇐ max(S) % j-th position in the S array
if maxSim > s2 then
  Child(i,j) ⇐ InsertionLeaf(Child(i,j),doc); %leafs only
else
  if maxSim > s3 then
    Child(i,j) ⇐ InsertionNonLeaf(Child(i,j),doc);%nonleafs only
  else
    Child(i,j+1) ⇐ doc;
  end if
  if Child(i)> B then
    Node(i) ⇐ split(Child(i));
  end if
end if
```

Algorithm 3: Insertion algorithm for a document in a non-leaf node

C.1.2 Insertion operation for leaf node

If document *doc* is inserted in node i, a pairwise comparison is performed with the documents in this leaf node. If the highest similarity is above a predefined threshold $s1$, the document *doc* will form a basket with the related document. Otherwise, the document *doc* will form a new child node. If the maximum number of child nodes (parameter B) is exceeded, the splitting procedure is started.

Node(i) = **InsertionLeaf**(Node(i),doc)

```
% S array with similarities
S ⇐ pairwiseSimmilarity(doc,Child(i))
%j-th position in the S array
(maxSim,j) ⇐ max(S)
if maxSim > s1 then
  documents of Child(i,j) ⇐ doc; % Assign document to this child
else
  document of Child(i,j+1) ⇐ doc; % Assign document to a new child
end if
if Child(i)> B then
  Node(i) ⇐ split(Child(i)); % If too many childs, the node is split
end if
```

Algorithm 4: Insertion algorithm for a document in a leaf node

C.1.3 Splitting operation for leaf node

The splitting procedure is applied on a node when the maximum number of child nodes (parameter B) is exceeded. If this parameter is exceeded, the node is split in two nodes based on pairwise similarities. The pair of child nodes resulting in the lowest similarity is used as seeds. The other child nodes are assigned occurring to their similarity with the seeds. The adaptations suggested in Section 3.1.2.2 for the splitting procedure are not shown.

```
Node(i) = split(Node(i))
  %Calculation of pairwise distances between the children of the node
  S ← pairwiseSimmilarity(children of Node(i))
  The two child nodes resulting in the lowest similarity are selected as seeds
  (Seed1,Seed2) ← min(S);
  j ← 1;
  for (j ≤ number of child nodes) do
    Child = child(j) of Node(i);
    if (Child not equals Seed1 or Seed 2) then
      S1 = Similarity(Child,Seed1);
      S2 = Similarity(Child,Seed2);
      if (S1>S2) then
        Children of Seed1 ← Child; % Child more similar to Seed1
      else
        Children of Seed2 ← Child; % Child more similar to Seed2
      end if
    end if
    Children of Node(i) ← [Seed1 Seed2];
  end for
```

Algorithm 5: Splitting procedure for a leaf node in the incremental clustering algorithm

C.2 Pairwise adaptive dissimilarity measure

In this section, the algorithmic descriptions of the Pairwise Adaptive Dissimilarity measure are given. In these descriptions, the following notation conventions are used:

- The comments in the code are in italic font, and proceeded with the '%'sign.
- The variables in these comments are marked with '‘ (e.g. 'variable‘).
- The use of a function 'function-name‘ with parameter 'parm‘ is denoted with function-name(parm).

- The index 'i' in a structure 'array' is indicated as array(i).
- The concatenation of items 'items' to a structure 'array' is denoted as array = [array items]

This Pairwise Adaptive Dissimilarity measure constructs a varying subspace, one per dissimilarity calculation. Two possible configurations exist: a static number of terms is taken of each document vector (variable static terms), or a variable number of terms is taken equal to $p * min(doc1, doc2)$ wherein p is the percentage of the minimum of the number of terms $doc1$ and $doc2$.

This description is provided in two procedures. The first procedure, which is the main procedure, is given in Algorithm 6. In this algorithm, the second procedure is used for the creation of a subspace per distance calculation. This algorithm is described in Algorithm 7.

dissimilarity = **padist**(DTM, static terms, dynamic)

if static terms > 0 **then**
  K ⇐ static terms; *% K is static.*
else
  K ⇐ 0; *% K will be determined dynamically.*
end if
i=0;
while $i \leq$ number of documents **do**
  j = i+1;
  Doc_i = A(i);
  while $j \leq$ number of documents **do**
    Doc_j = A(j);
    space ⇐ subspace(Doc_i,Doc_j,K); *%Subspace algorithm*
    similarity ⇐ cosine(space);
    dissimilarity ⇐ 1-similarity;
    j ⇐ j+1;
    i ⇐ i+1;
  end while
end while

Algorithm 6: Pairwise Adaptive Dissimilarity algorithm

space = **subspace**(doc1,doc2,static terms)

K ⇐ static terms; *%If K is assigned a value, it is used.*
if K = 0 **then**
K ⇐ p * min(length of doc1, length of doc2); *% p is a given parameter, K is dynamic.*
end if
ind1 ⇐ sort(doc1); *% Sort the first document vector.*
ind2 ⇐ sort(doc2); *% Sort the second document vector.*
indices ⇐ union(ind1(K),ind2(K));
%Take the union of the K indices of the first and second document.
space1 ⇐ doc1(indices); *% Construct the document representation in the new subspace.*
space2 ⇐ doc2(indices);
space ⇐ [space1;space2]; *% A matrix size 2* x *length wherein the documents are represented.*

Algorithm 7: Subspace creation for the pairwise adaptive dissimilarity

Appendix C

Appendix D

Multivector Representation of Documents

D.1 Stopwords for language identification

The process to identify and extract the different language sections differs from the original multivector identification process in the identification of the lexical chains. Instead of using all possible lexical chains, only a specific set of chains based on stopwords is retained. The set of stopwords for four languages is shown in the Table D.1.

Table D.1: Selection of terms used in the topic identification technique to identify varying language sections of a document

Dutch	**French**	**English**	**German**
aan	mais	about	aus
dat	aux	after	auf
de	les	all	bei
deze	par	and	bis
die	pour	as	das
dit	tout	at	der
door	tous	by	ein

D.2 Algorithmic descriptions

This section overviews the process that is used for the multi-vector representation process of a document. The diagram in Figure D.1 overviews this entire process.

In the following paragraphs, the most important algorithms in this process are described.

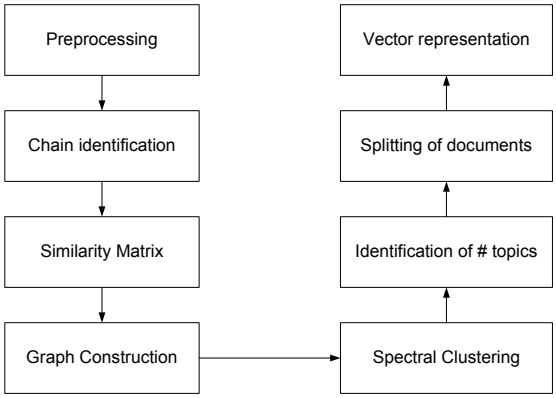


Figure D.1: General overview of the stages in the multivector representation process.

In the described algorithms, the comments in the code are in italic font and proceeded with the '%'sign. The variables in these comments are marked with '' (e.g. 'variable').

As explained in Chapter 6, the coherence of a document is based on the lexical chains in the content of this document. The algorithm used to identify these lexical chains is explained in Algorithm 9. After obtaining the different chains of a document, the similarity matrix based on these chains is calculated. This construction process is explained in Algorithm 10. The Algorithm 12 uses the Algorithm 11 to obtain the normalized Laplacian. This matrix is built based on the degree and adjacency matrix of the similarity matrix, which is the basis of the graph. Algorithm 12 describes the algorithm known as *normalized Jordan spectral clustering*. In this algorithm a spectral decomposition of the similarity matrix is performed. Based on a predefined or calculated number of clusters, a new subspace is created wherein a clustering algorithm is applied (e.g. hierarchical agglomerative clustering). This clustering algorithm returns an index, wherein the component label for each document entity is given. These labels are then used to regroup the original document in order to obtain the subdocuments. These subdocuments are tokenized and preprocessed to obtain the final multiple vectors representing the document (see Algorithm 13).

A = **multiple-vectors-of**(document)

document ⇐ preprocessing(document); *% Necessary preprocessing steps to convert a document to its ASCII format, and to include the delimiters in the content of the document.*
chains ⇐ chainIdentification(document); *% Identification of the relevant lexical chains in the document.*
S ⇐ similarityMatrix(chains); *% Construction of the similarity matrix based on the lexical chains.*
L ⇐ normLap(S); *% Construction of the normalized Laplacian.*
indices = ⇐ normJordan(E, number of clusters); *% Clustering process in a subspace based on the spectral properties of the graph (similarity matrix).*
A = extract-vectors(indices, document); *% The recombination procedure of the 'document' based on the array 'indices', and the tokenization of the new subdocuments to obtain the vectors. The vectors are stored as columns in the matrix 'A'.*

Algorithm 8: Construction of a multiple vector representation of a document

chains = **chainIdentification**(document)

tokens = tokenizer(document); *% Assumed is that the document is converted into a ASCII format, and that delimiters (variable 'delimiter') are present to indicate the different document entities. % The document is broken into 'tokens' (i.e. list of individual terms, omitting the document layout.)*
tokens = preprocessing(tokens); *% The necessary preprocessing steps, as explained in Chapter 6 are applied on the token list.*
chains ⇐ []; *% An empty array is assigned to this variable.*
delimiters = find(tokens,delimiter); *% The variable 'delimiters' contains the indices of the delimiters. % The index of a term is defined as the number of terms before this term. % With these delimiters, the collection of terms per document entity is identified.*
for i=1:size(tokens,1) **do**
  i ⇐ find(tokens(i),tokens); *% The array 'i' contains all indices of the document entities wherein 'tokens(i)' occurs.*
  chains ⇐ [tokens(i);i]; *% The structure chains contains per 'tokens(i)' (i.e. term defining a lexical chain) the indices 'i' wherein this token occurs.*
end for

Algorithm 9: Identification of lexical chains in a document

S = **similarityMatrix**(chains)

A ⇐ zeros(size(chains,2));
for i=1:size(chains,1) **do**
dei ⇐ document-entities-indices(chains(i)); *% per chain the indices of the document entities involved in the chain are stored in the array 'dei'.*
co = pairs-of-indices(dei); *% all combinations of two indices of document entities are stored in the array 'co'*
for j=1:size(co,1) **do**
A(co(j,1),co(j,2)) ⇐ A(co(j,1),co(j,2)) + sim; *% for every combination of two involved document entities, the similarity based on this chain is added in the similarity matrix;*
A(co(j,2),co(j,1)) ⇐ A(co(j,1),co(j,2));*% necessary step to create a symmetric matrix*
end for
end for
isolated ⇐ []; *% Document entities that have no relations with other document entities are stored in this array.*
for i=1:size(A,1) **do**
if (size(find(A(i,:)),2) = 0) **then**
isolated ⇐ [isolated i]; *% If no similarity is added to the matrix for this entity. Add the index to the 'isolated' entities array;*
end if
end for
A(isolated,:) ⇐ []; *% Remove the isolated entities from the similarity matrix.*
A(:,isolated) ⇐ []; *% This removal is necessary because the Laplacian has to be calculated on a matrix of full rank.*

Algorithm 10: Construction of the similarity matrix

L = **normLap**(S)

A ⇐ adjacencyMatrix(S); *% A is the adjacency matrix of the similarity matrix S.*
D ⇐ degreeMatrix(S); *% D is the degree matrix of the similarity matrix S.*
n ⇐ size(A,1); *% The number of document entities is assigned to the variable n*
L ⇐ identity-matrix(n) - $D^(- 1/2) * A * D^(- 1/2)$; *% The normalized Laplacian is constructed based on*
% the degree, adjacency and identity matrix.

Algorithm 11: Calculation of the normalized Laplacian

indices = **normJordan**(E, number of clusters)

L ⇐ normLap(E); *% Calculation of the normalized laplacian based on the similarity matrix 'E'*
[V,D] ⇐ eig(L); *% Calculation of the eigenvalues of the Laplacian 'L'*
%Two possible configurations:
% - the number of clusters is given beforehand, and the procedure uses this value. % - the SSI algorithm is used to obtain the number of clusters value.
if number of clusters ¿ 0 **then**
  k ⇐ number of clusters;
else
  B ⇐ L(w,w); *%Laplacian reordered according to the eigenvector belong the the second largest eigenvalue. This reordering is used to identify the square segments.*
  [k,a] ⇐ SSI(B); *% The square segments identification procedure*
end if
U ⇐ V(:,1:k); *%The first 'k' eigenvectors are selected, which creates a new subspace.*
for i=1:size(U,1) **do**
  U(i,:) ⇐ U(i,:)/norm(U(i,:)); *%The matrix is normalized by using the 2-norm.*
end for
indices ⇐ clustering-step(U,k); *% The second clustering step is performed, e.g. hierarchical agglomerative clustering. This algorithm results an array size (1 x number of entities) wherein every index indicates the coherent component where this entity belongs to.*

Algorithm 12: Normalized Jordan spectral clustering

L = **extract-vectors**(indices, document)

de ⇐ read(document); *% The document entities from the document are read into the array 'de'. This process is based on the*
% identification of the delimiters. The string between a pair of delimiters is a document entity.
[order,idx] ⇐ sort(indices); *% The 'indices' are sorted which results in the sorted array 'order', and the array 'idx' which contains the original positions of the document entities;*
de ⇐ de(idx); *% The document entities 'de' are sorted in the order as imposed by the array 'order'.*
directory-location ⇐ write(de); *% The recombined document entities are written in separate files. Each file contains document entities with the same label.*
A ⇐ indexing(directory-location); *% All newly created documents are indexed, preprocessed and stored in the matrix 'A' wherein every column contains a document vector.*

Algorithm 13: Construction of the vectors of the topic related components

Appendix E

Near-duplicate Identification

The procedure of near-duplicate identification between a pair of documents is described in this section. As explained in Section 7.2.3, this procedure is composed of three phases. The first phase is the construction of the multivector representation of the two documents. The second phase is the pairwise comparison of all coherent components of both documents. If such a pairwise comparison results in a similarity value which is higher than a predefined threshold value, the third phase is applied on these components: a comparison based on their lexical chains. In the next algorithms, this procedure is explained.

The main procedure for this near-duplicate identification process is shown in Algorithm 14, which calls Algorithms 15 and 16 for the second and third phase of the process. Algorithm 15 is the cosine distance calculation, while the Algorithm 16 is the comparison based on the lexical chains in both components.

In the described algorithms, the comments in the code are in italic font and proceeded with the '%'sign. The variables in these comments are marked with '' (e.g. 'variable').

score = **ndd**(document1, document2)

```
[A1,chains1] ⇐ multiple-vectors-of(document1); % The multivector
representation and chains of document1 are identified.
[A2,chains2] ⇐ multiple-vectors-of(document2); % The multivector
representation and chains of document2 are identified.
score ⇐ []; % The array 'score' stores the pairwise comparisons of the
components of document1 and document2.
for (i=1:size(A1,2)) do
  vector1 = A1(i,:); % 'vector1' is the vector representation of component 'i'
  of document1.
  for (i=j:size(A2,2)) do
    vector2 = A2(j,:); % 'vector2' is the vector representation of component 'j'
    of document2.
    s = ndd-1p(vector1,vector2);
    if (s ¡ threshold) then
      score ⇐ [score 0]; % If the similarity 's' is below the predefined
      threshold, a zero value is added to the score array.
    else
      ch1 ⇐ retrieve-chains-of(chains1,i); % If the similarity is above the
      predefined threshold value,
      ch2 ⇐ retrieve-chains-of(chains2,j); % the chains in the related
      document entities are retrieved.
      s = ndd-2p(ch1, ch2);
      score ⇐ [score s]; % The similarity 's' is added to the score array.
    end if
  end for
end for
```

Algorithm 14: Near-duplicate identification algorithm

score = **ndd-1p**(vector1,vector2)

```
score ⇐ 1-'cosine-distance' (vector1, vector2); % The cosine distance is
calculated between the vectors 'vector1' and 'vector2'.
```

Algorithm 15: First phase in the near-duplicate identification process

dissimilarity = **ndd-2p**(chains1, chains2)

```
nbchains ⇐ min(chains1,chains2); % The variable 'nbchains' is the minimum
of the number of chains1 and the number of chains2.
score ⇐ 0; % The score array will contain the pairwise score of the two
components based on chain comparisons.
for (i=1:size(chains1)) do
  for (j=1:size(chains2)) do
    if (common-length(chains1(i),chains2(j)) then
      score ⇐ score + min(chain1(i),chain2(j))); %If they have a chain in
      common, a score equal to the minimal number of occurrences is added
      to the score.
    else
      score ⇐ score - max(chains1(i),chains2(j)); % If not, the non-zero value
      is subtracted from the score.
    end if
  end for
end for
score ⇐ score/nbchains; % The score is divided by the variable 'nbchains' to
obtain a score based on the lowest number of chains.
```

Algorithm 16: Second phase in the near-duplicate identification process

Appendix E

Appendix F

Incremental clustering algorithm

F.1 Operations for Incremental Clustering

The incremental clustering algorithm described in Section 3.1.2 is

F.1.1 Insertion operation for non-leaf node

F.1.2 Insertion operation for leaf node

F.1.3 Splitting operation for leaf node

$Node_i$ = **InsertionNonLeaf**$(Node_i, doc)$

$S = pairwiseSimmilarity(doc, Child_i)$ *%S array with similarities*
$(maxSim, j) \Leftarrow max(S)$ *%j-th position in the S array*
if maxSim > s2 **then**
$Child_{i,j} \Leftarrow InsertionLeaf(Child_{i,j}, doc)$ *%leafs only*
else
if maxSim > s3 **then**
$Child_{i,j} \Leftarrow InsertionNonLeaf(Child_{i,j}, doc)$ *%nonleafs only*
else
$Child_{i,j+1} \Leftarrow doc$
end if
if $Child_i$ > B **then**
$Node_i \Leftarrow split(Child_i)$
end if
end if

Algorithm 17: Insertion procedure for a non-leaf node in the incremental clustering algorithm

$Node_i$ = **InsertionLeaf**$(Node_i, doc)$

$S = pairwiseSimmilarity(doc, Child_i)$ *%S array with similarities*
$(maxSim, j) \Leftarrow max(S)$ *%j-th position in the S array*
if maxSim > s1 **then**
$Child_{i,j}.docs \Leftarrow doc$
else
$Child_{i,j+1} \Leftarrow doc$
end if
if $Child_i$ > B **then**
$Node_i \Leftarrow split(Child_i)$
end if

Algorithm 18: Insertion procedure for a leaf node in the incremental clustering algorithm

if max(S1,S2)>threshold **then**
  if S1>S2 **then**
    $Seed1.add \leftarrow Child_{n,j}$
  else
    $Seed2.add \leftarrow Child_{n,j}$
  end if
else
  $Child_{n,j}.it = Child_{n,j}.it + 1$
  if $Child_{n,j} > maxit$ **then**
    $delete(Child_{n,j})$
  else
    add $Child_{n,j}$to queue
  end if
end if

Algorithm 19: Splitting procedure for a leaf node in the incremental clustering algorithm

Appendix F

Appendix G

SAME

G.1 Case Results

G.1.1 20Newsgroup

The first case subjected to the SAME technique was the 20Newsgroup, of which four topics were selected. The selected topics were

- Cryptography
- Car
- Atheism
- Baseball

The results of the SAME technique on this case is shown in the following sections. In Section G.1.1.1, the most important terms for the first layer of the four most important MLCs are given. Section G.1.1.2 shows the term networks of these four largest MLCs.

G.1.1.1 Terms related to multilayer clusters

In Table G.1, a limited selection of important terms of the first layer of each of the four MLC is shown, obtained from the SAME technique. Each column displays the terms of a MLC. The listed terms already give an indication of the document type they represent.

Table G.1: Selection of terms of the four most important MLC

Crypto	**Atheism**	**Baseball**	**Car**
'key'	'god'	'team'	'car'
'clipper'	'keith'	'game'	'cars'
'chip'	'sgi'	'baseball'	'oil'
'encryption'	'livesey'	'year'	'engine'
'government'	'atheists'	'games'	'clutch'
'keys'	'caltech'	'players'	'dealer'
'escrow'	'morality'	'braves'	'insurance'
'netcom'	'religion'	'runs'	'ohio'
'access'	'atheism'	'hit'	'uiuc'
'nsa'	'moral'	'morris'	'miles'
'algorithm'	'islam'	'win'	'state'
'phone'	'mathew'	'season'	'ford'
'public'	'solntze'	'sox'	'price'
'crypto'	'people'	'player'	'callison'
'security'	'jon'	'ball'	'usa'
'sternlight'	'wpd'	'league'	'university'
'david'	'sandvik'	'pitcher'	'speed'
'system'	'islamic'	'phillies'	'bmw'
'pgp'	'okcforum'	'hitter'	'acs'
'intercon'	'tek'	'colorado'	'purdue'

G.1.1.2 Term networks

In this section, the term networks of the four largest MLCs are shown as a result of the SAME technique discussed in Chapter 4. Based on these networks, the experts provided their answers during the validation process. On the next pages and in the following order, these figures can be found:

- Figure G.1 represents the term network that, after the validation process, is related to the document type of 'Crypto'.
- Figure G.2 describes the fuzzy topic of atheism, which did result in a general document model.
- Figure G.3 shows the term network related to the topic of 'Baseball'.
- The topic of 'cars' is finally described in the term network shown in Figure G.4

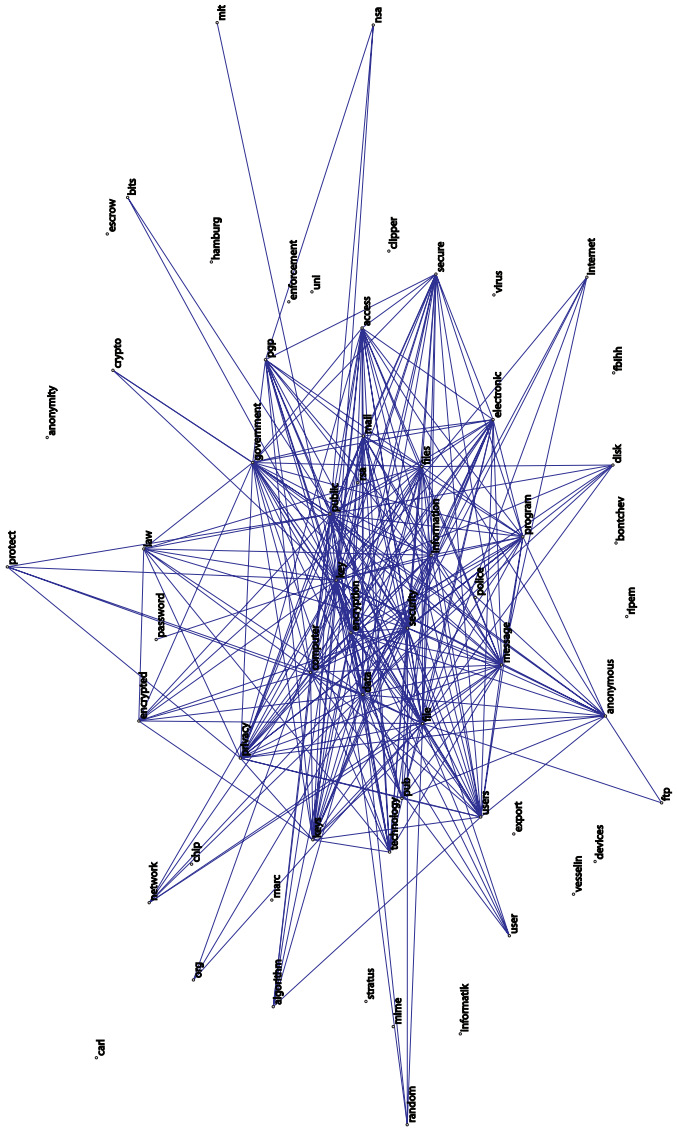


Figure G.1: Term network of the multilayer cluster related to 'Cryptography'

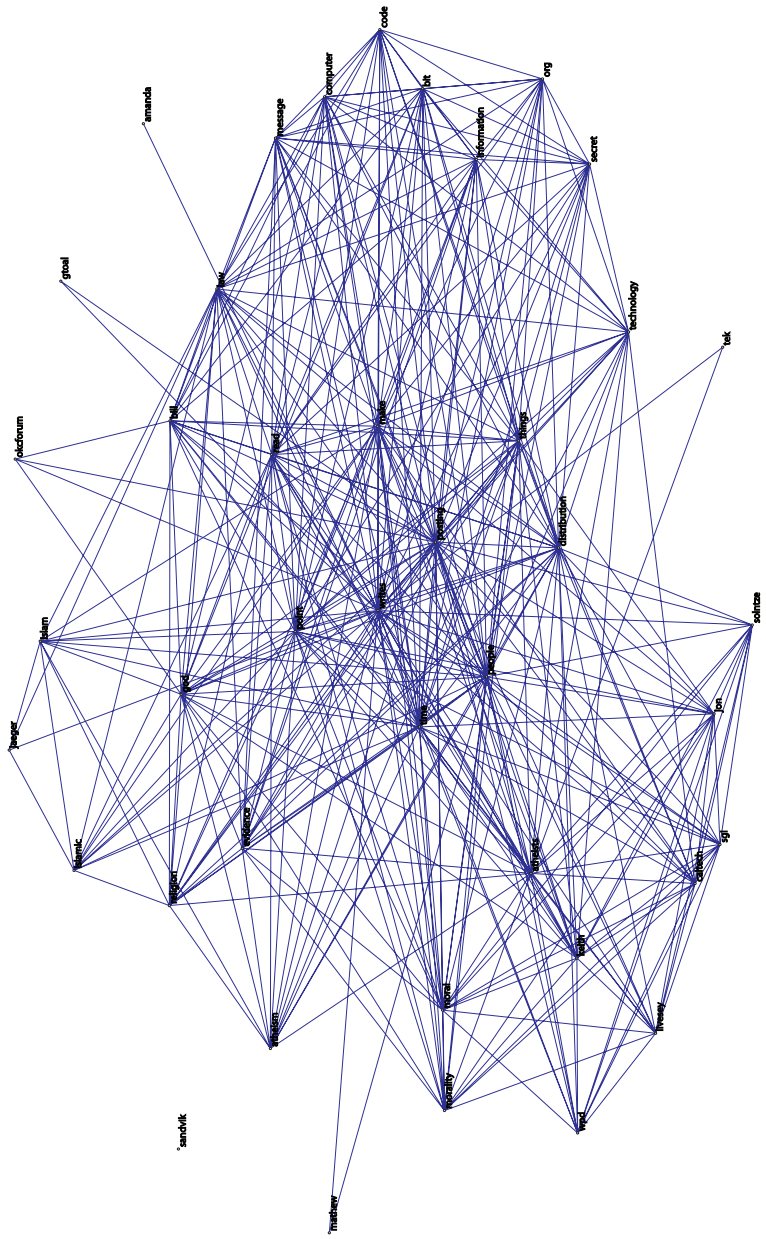


Figure G.2: Term network of the multilayer cluster related to 'Atheism'

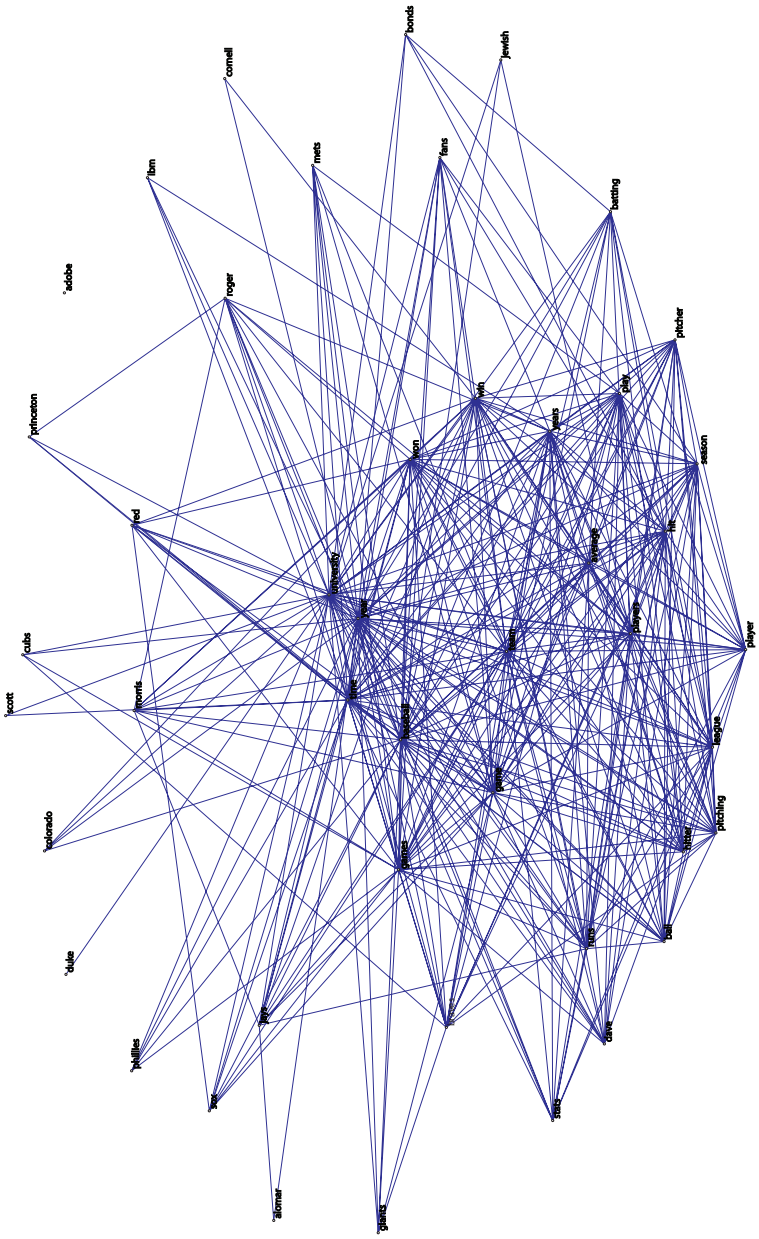


Figure G.3: Term network of the multilayer cluster related to 'Baseball'

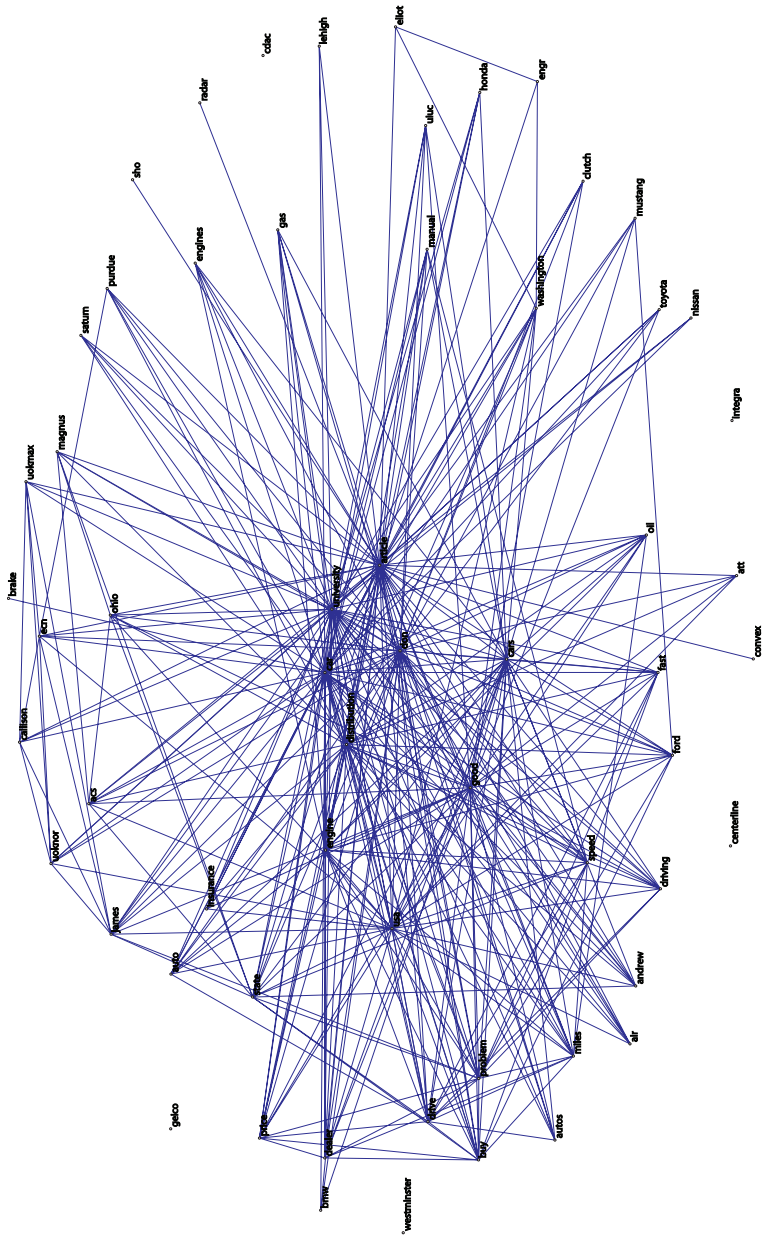


Figure G.4: Term network of the multilayer cluster related to 'Cars'

G.1.2 Research group

The results of the SAME technique on the Research case are shown in the following sections. In the first section, the terms used for the template identification in Section 4.5.2 are shown in Table G.1.2.1. In Section G.1.2.2, the term networks for the five most important MLCs are given.

G.1.2.1 Template terms

To calculate the template score for the five most important MLCs, the terms in Table G.1.2.1 were used.

Admittance letters	**Invoice**	**Thesis proposals**	**POC reports**	**Thesis contracts**
reference	invoice	thesis	poc	agreement
kuleuven	afdelings-	students	courses	firm
kenmerk	verantw.	company	timetable	title
admittance	kuleuven	work	meeting	students
cib	kenmerk	proposal	mim	name
academic	cib	fields		
programme		week		

G.1.2.2 Term networks

The term networks that were used in the validation procedure are shown in this section. The five identified document types are, in the same order as shown on the next pages:

- Admittance letter
- Thesis proposal
- Thesis contract
- Invoice
- Timetable

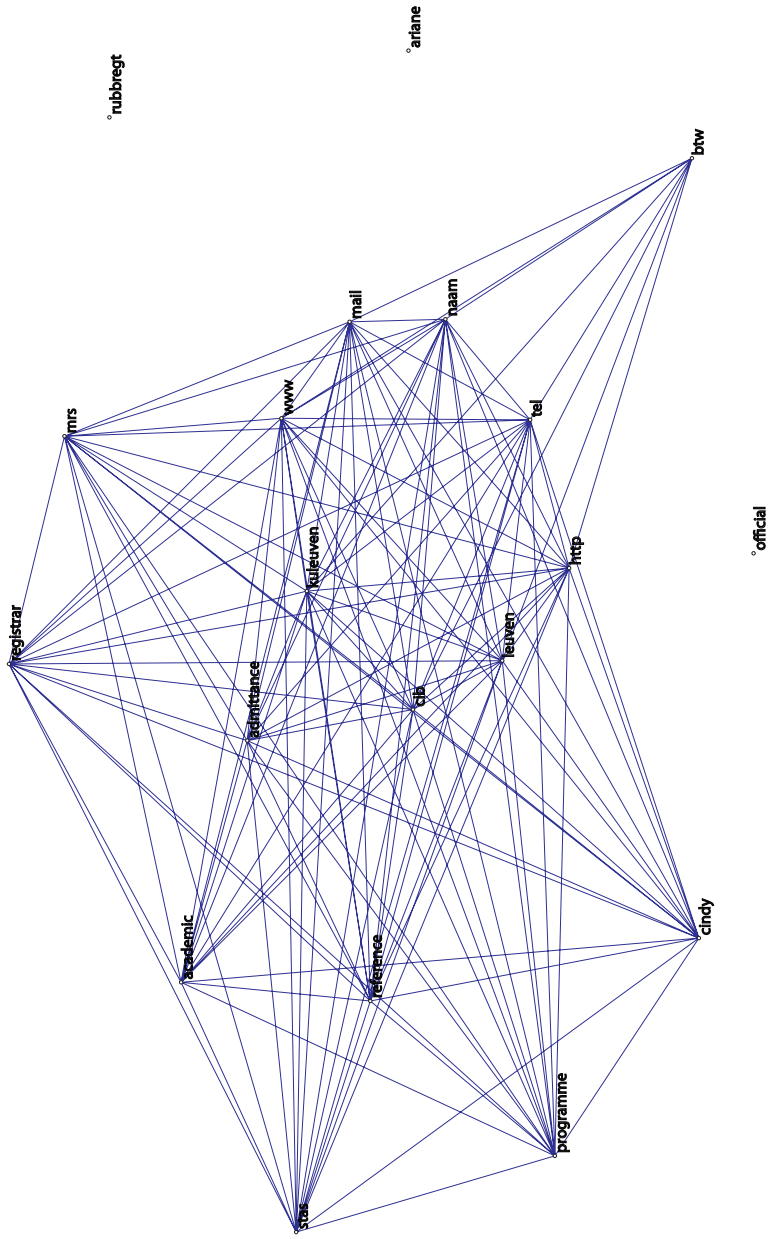


Figure G.5: Term network of the multilayer cluster related to 'Admittance letters'

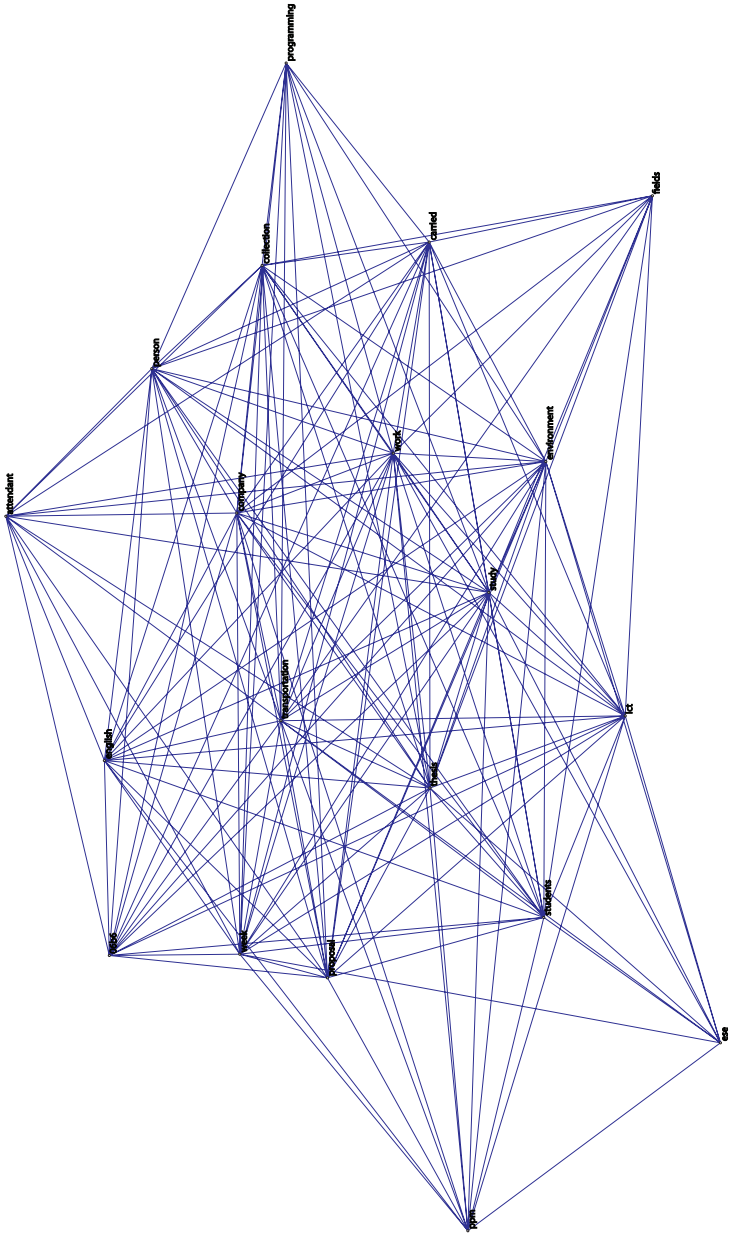


Figure G.6: Term network of the multilayer cluster related to 'Thesis proposals'

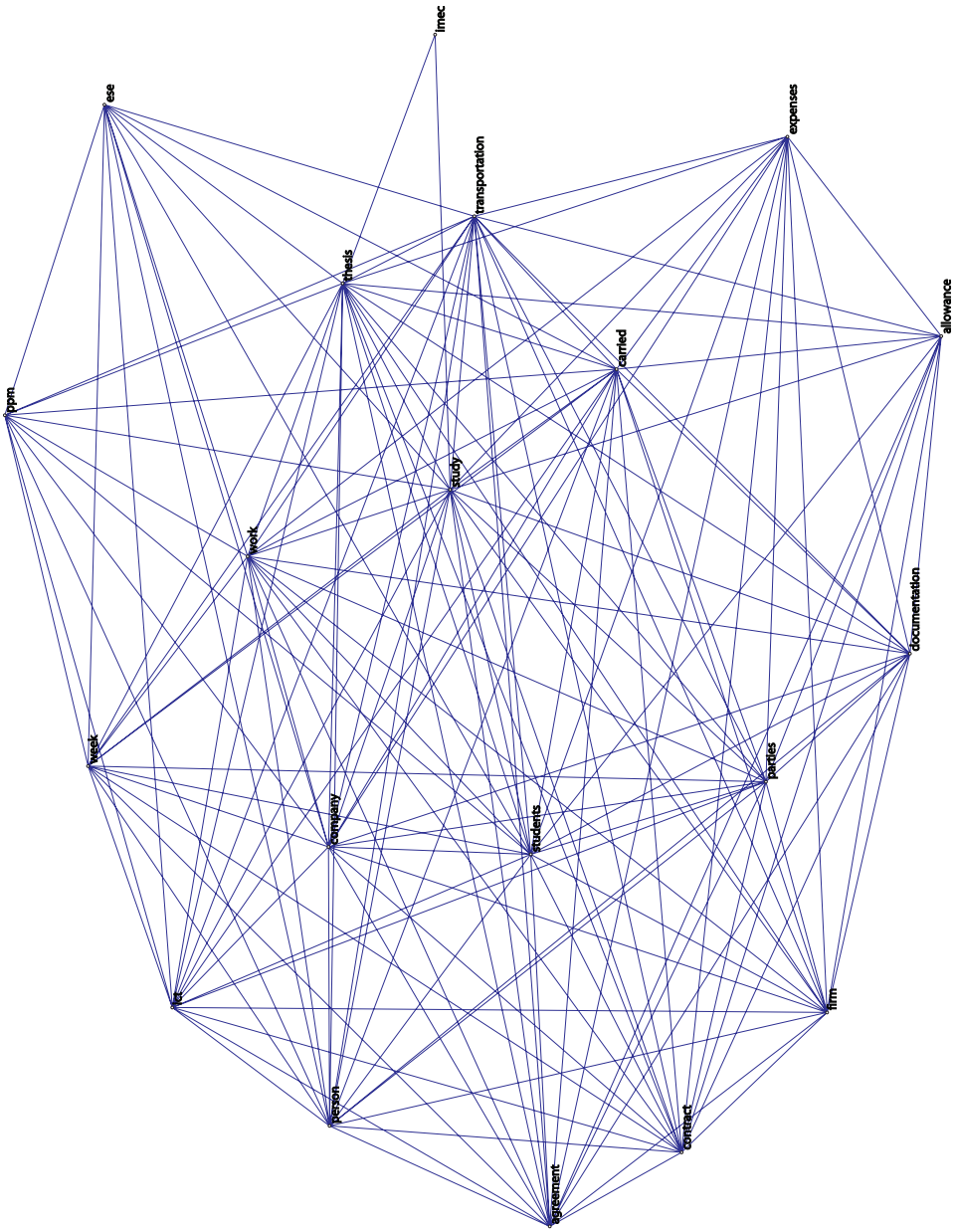


Figure G.7: Term network of the multilayer cluster related to 'Thesis contracts'

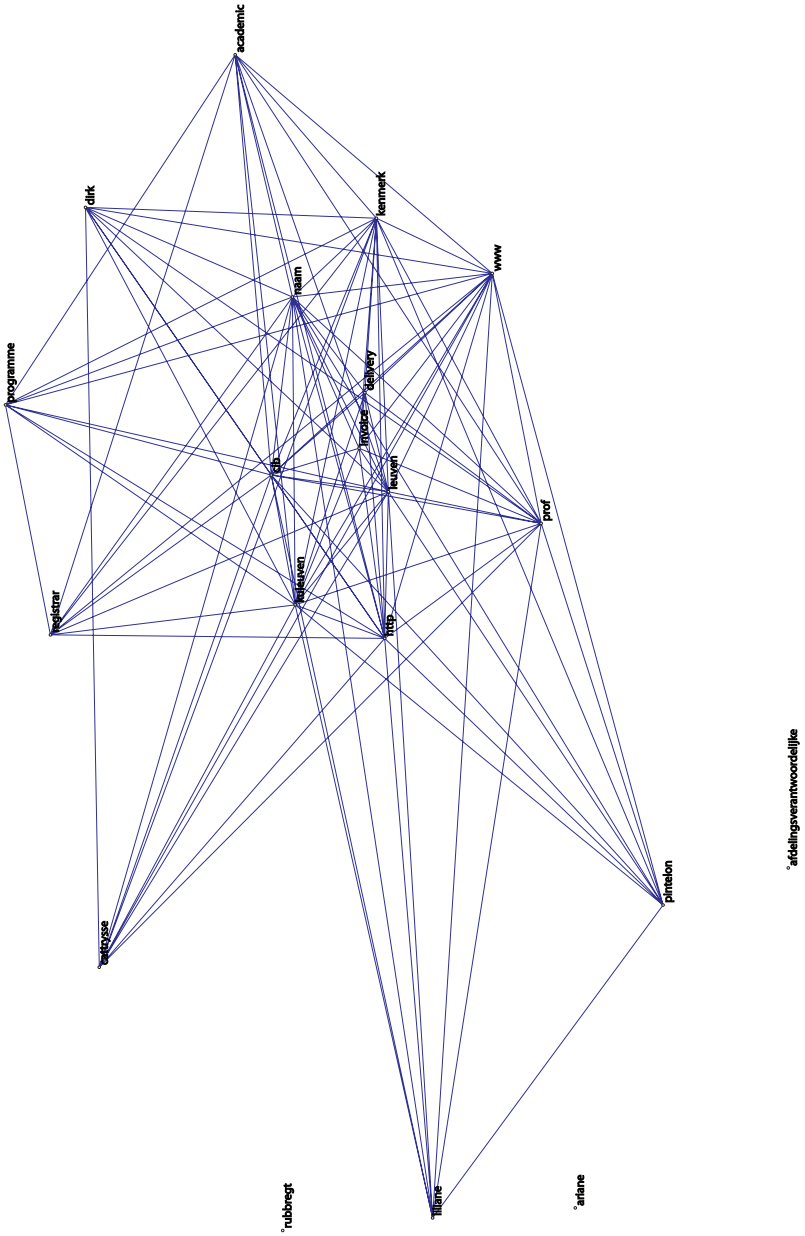


Figure G.8: Term network of the multilayer cluster related to 'Invoices'

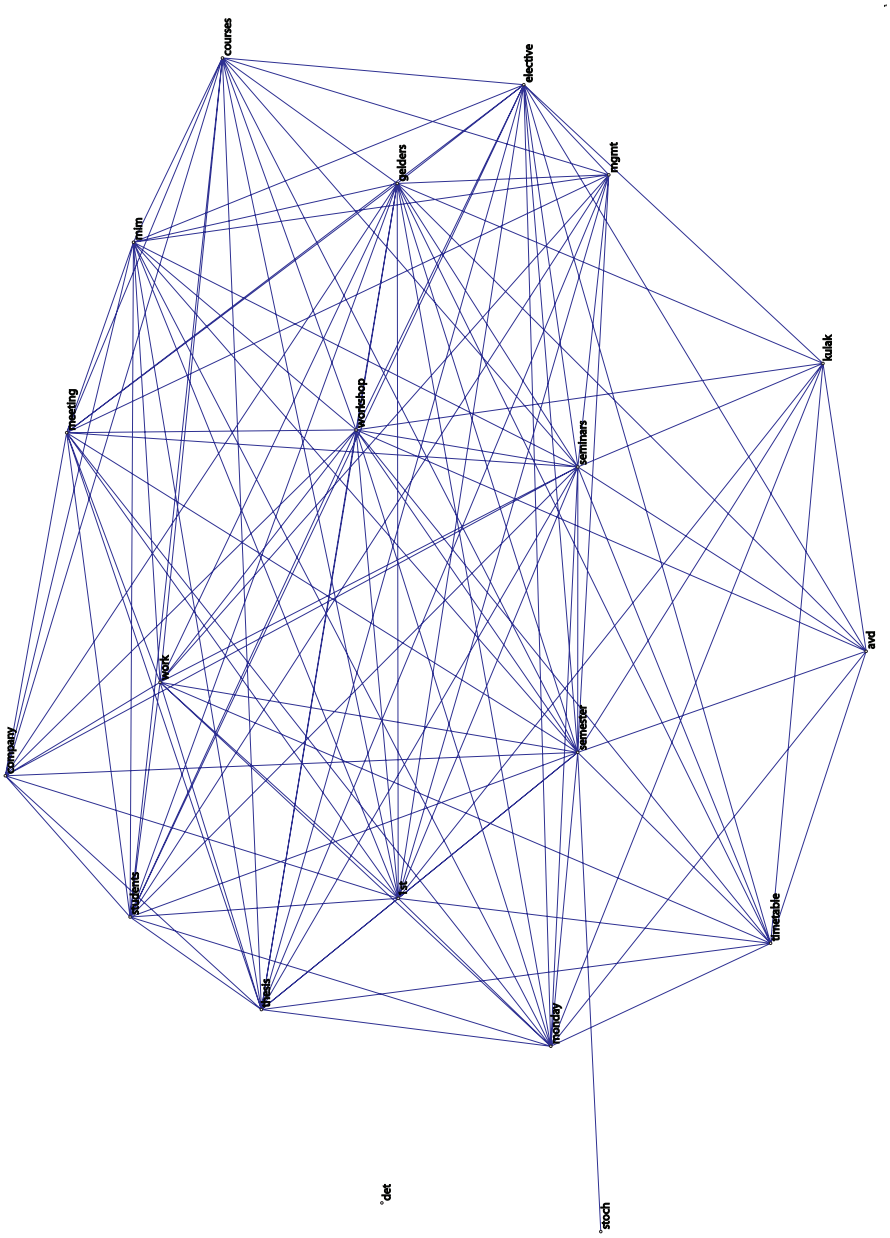


Figure G.9: Term network of the multilayer cluster related to 'Timetables'

G.1.3 Quotes

G.1.3.1 Terms related to multilayer clusters

In this section, the terms defining several MLCs are presented. In the following order, the terms for MLC number 2, 3 and 4 are presented in Tables G.2, G.3 and G.4.

Table G.2: Selection of the most important terms for MLC number 2

quinn	base	sep
peter	welding	acknowledgement
dean	yioula	<company name>
group	olivier	pdf
	rapoye	smtp
	glassworks	quote
	nicola	sales
	mazza	public
	material	received

Table G.3: Selection of most important terms for MLC number 3

mortier	deutschland	werk
liebich	panzerung	kontakt
manfred	anfrage	tevfik
yves	demirtas	telefon
ardaghglass	nienburg	lackmajer
ardagh	angebot	sylvia
		austria
		hals

Table G.4: Selection of most important terms for MLC number 4

dominique	matiere	demande
lopez	metallisation	franky
gerresheimer	momignies	rue
france	dobbelstein	philippe
	prix	cartel
	oorreel	usine
		achats

G.1.3.2 Term networks

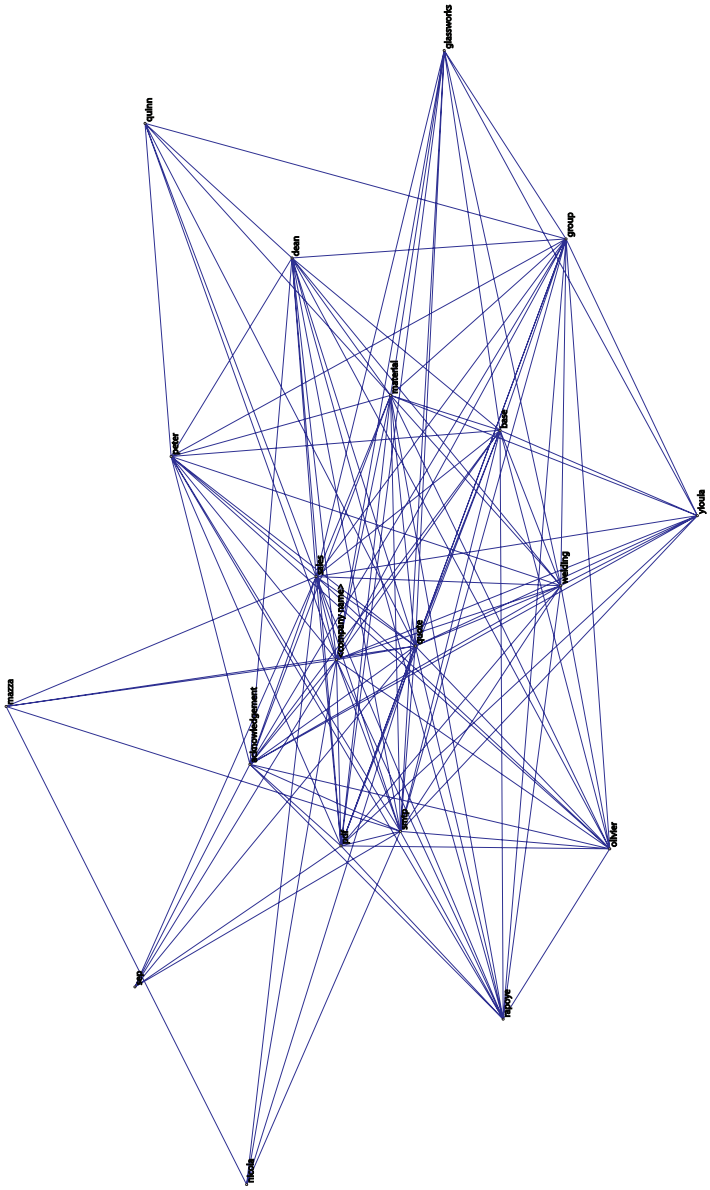


Figure G.10: Term network of the first multilayer cluster related to 'English sales quotes'

G.2 Algorithmic descriptions

In this section, the algorithmic descriptions of the SAME technique are provided. In these descriptions, the following notation conventions are used:

- The comments in the code are in italic font, and proceeded with the '%'sign.
- The variables in these comments are marked with '' (e.g. 'variable').
- The use of a function 'function-name' with parameter 'parm' is denoted with function-name(parm).
- The index 'i' in a structure 'array' is indicated as array(i).
- The concatenation of items 'items' to a structure 'array' is denoted as array = [array items]

As indicated in Figure 4.5, the SAME technique starts with the necessary preprocessing steps in order to convert each document to a workable format. The complete document collection is converted to a DTM using the approach described in Section 2.2.2.1. These preprocessing steps are the necessary prerequisites for the SAME technique, which is described in Algorithm 20. This algorithm has as input a DTM and the dictionary (or domain vocabulary) of the document collection. A number of iterations (i.e. a parameter) of this algorithm is performed, wherein the following steps are carried out:

- Hierarchical agglomerative clustering algorithm is applied on the DTM.
- The important terms defining this cluster structure are identified. Algorithm 21 describes the selection procedure of the deciding words of each cluster obtained in the first phase. The inputs of this algorithm are the DTM of the cluster from which the deciding words will be determined, and the list of terms or 'dictionary' as the deciding words are indicated by indices from this list. The 'threshold' parameter is used to decide the important terms. The algorithm proceeds by calculating the average term frequencies of the documents in the cluster. The list of these average frequencies is then sorted in descending order. In this way, the term with the highest average frequency is considered to be the most important term. The number of terms to be selected is governed by the accumulated average frequency. Every selected term adds his average frequency to this accumulated number. If the ratio of this figure and the maximum available frequency ('max-cumulative-frequencies') exceeds the 'threshold' value, the algorithm halts. The obtained list of terms contains the most important terms for this cluster.

- A filtering step is applied on the DTM of the document collection. The important terms and possible empty documents are removed (columns and, if applicable, rows are deleted).
- The fourth step is the identification of the MLCs. In Algorithm 22, the identification algorithm is shown. For each document, its cluster chain is compared to other document's cluster chain in order to identify the new MLCs.
- For each MLC, the co-occurrence matrix is constructed from which the term network can be obtained. This process is described in Algorithm 23.

A = **SAME**(DTM,dictionary)

```
MLC ⇐ []; A structure to store the MLCs, contains cluster-id chains
for (a ≤ number of cluster layers) do
  C ⇐ clustering-step(DTM,k); % C is the set of clusters, k is the required
  number of clusters.
  terms ⇐ []; % A structure to store the important terms.
  i ⇐ 1;
  for (i≤ number of clusters) do
    newterms ⇐ deciding-words(C,dictionary,threshold) Identification of
    deciding words of each cluster.
    terms ⇐ [terms newterms]; Add new terms with the existing list of terms
  end for
  indices = get-indices(terms) %Retrieve the indices in the DTM of the
  important terms.
  DTM(indices)=[]; % Remove the term columns, related to the important
  terms, from the DTM.
  d ⇐ 1;
  for (d ≤ number of documents) do
    if (sum-terms(DTM(d)) = 0 then
      DTM(d) ⇐ []; % Ff the document is empty, remove it from the DTM
    end if
  end for
  if (a > 1) then
    MLC ⇐ MLC-identification([MLC,C]); % Identification only useful of
    more than one iteration has passed
    for (b ≤ number of MLCs) do
      DTMcluster = DTM(documents,MLC,terms); a DTM is constructed of
      the documents belonging to a MLC;
      A = co-occurrence(DTMcluster); Co-occurrence matrix is constructed,
      and networks can be drawn
    end for
  end if
end for
```

Algorithm 20: Algorithmic description of the SAME technique

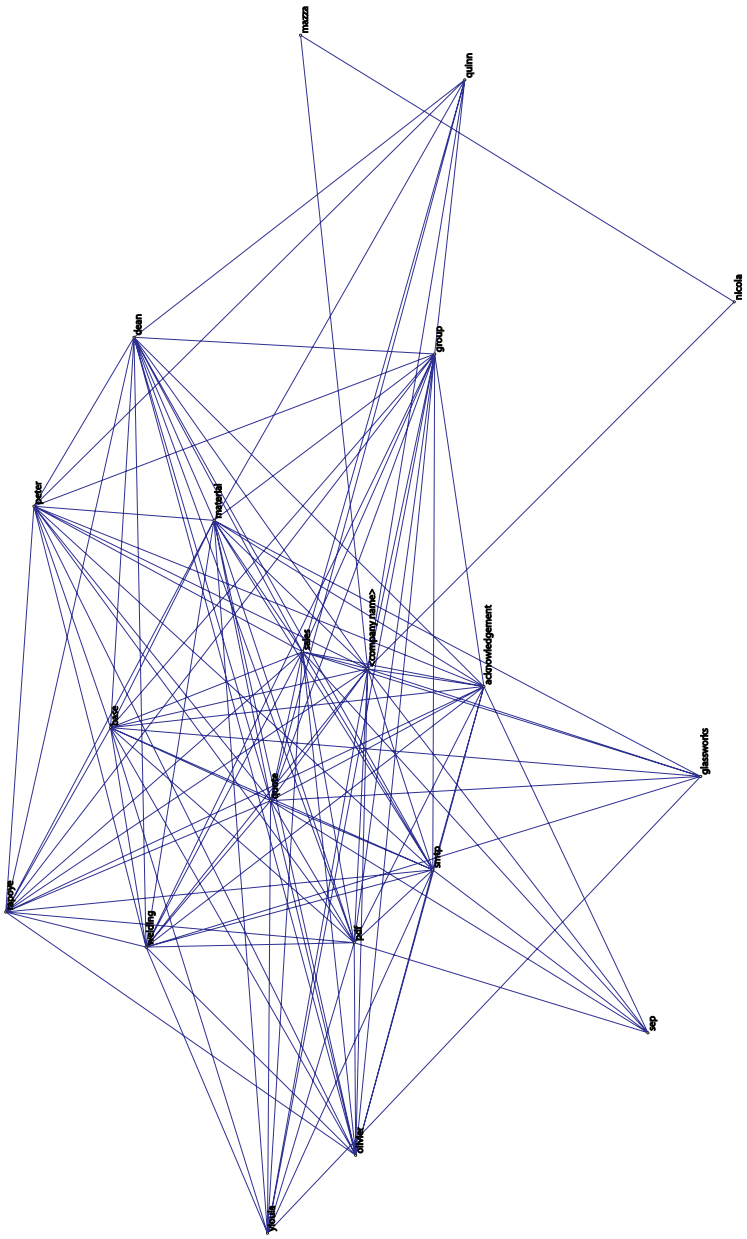


Figure G.11: Term network of the second multilayer cluster related to 'English sales quotes'. The difference between the first and second MLC is situated in the names of the company salesmen.

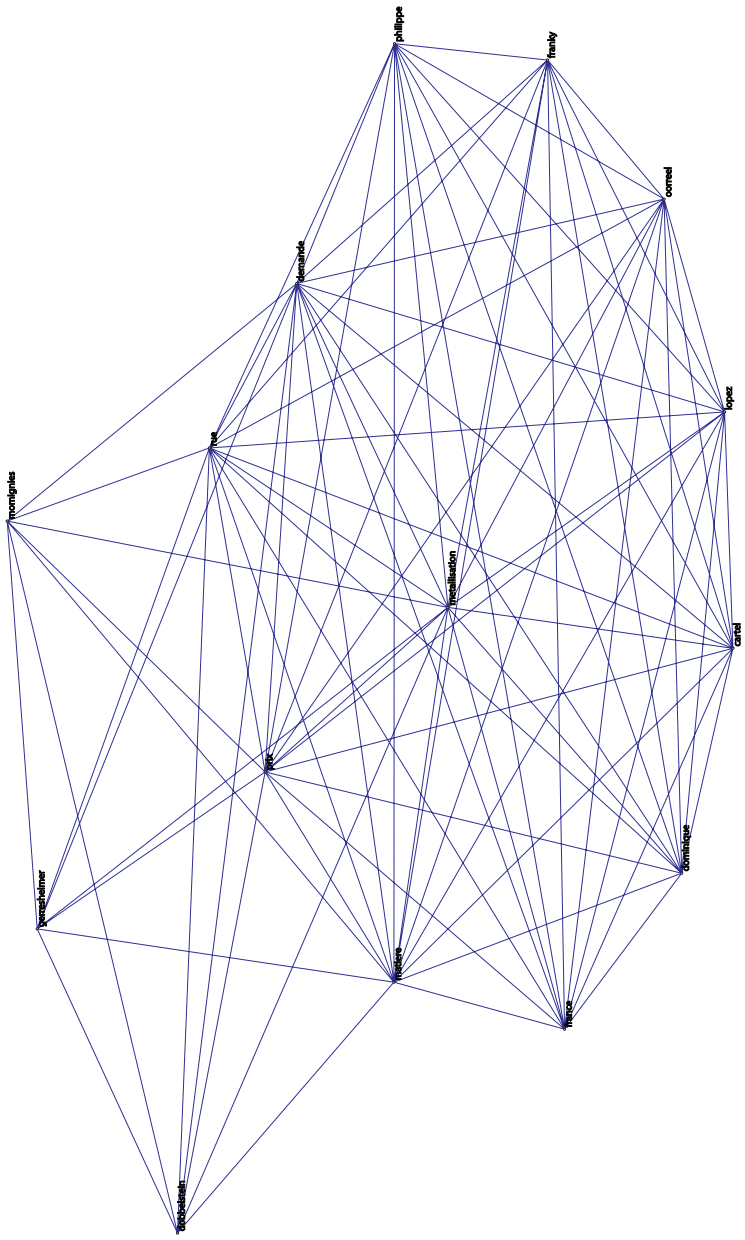


Figure G.12: Term network of the multilayer cluster related to 'French sales quotes'

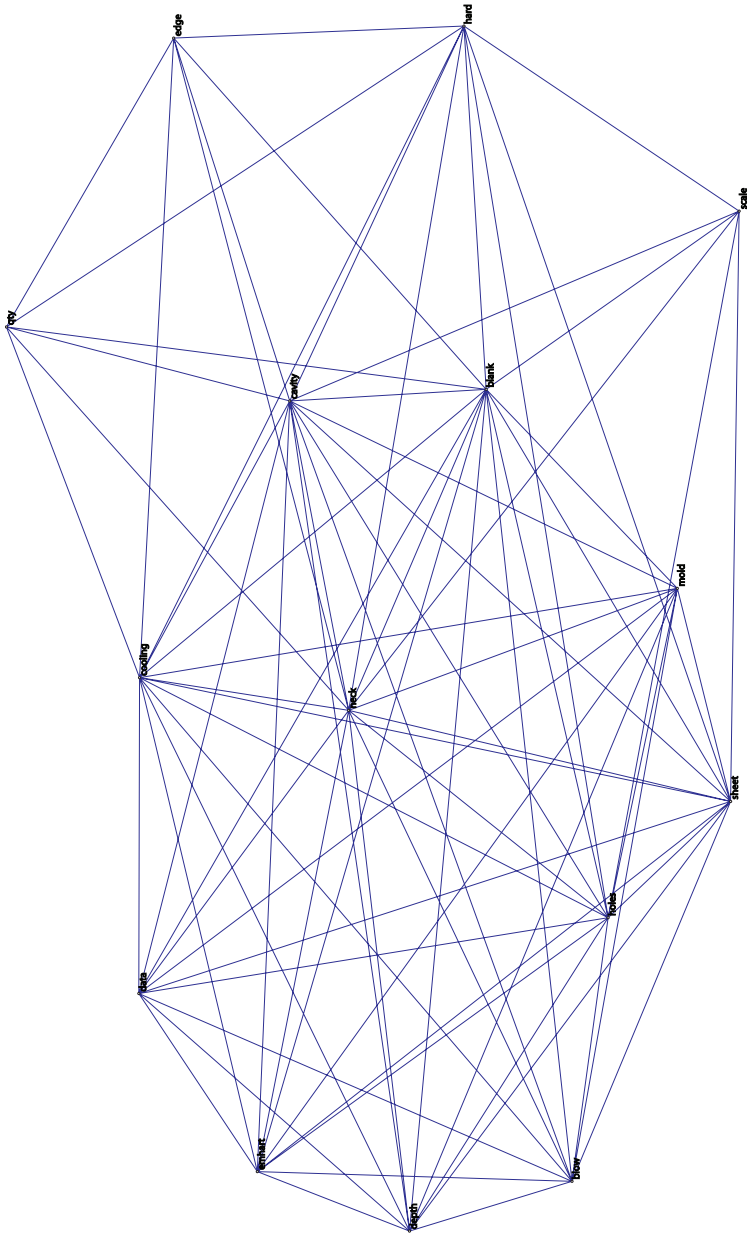


Figure G.13: Term network of the multilayer cluster related to 'Drawings'

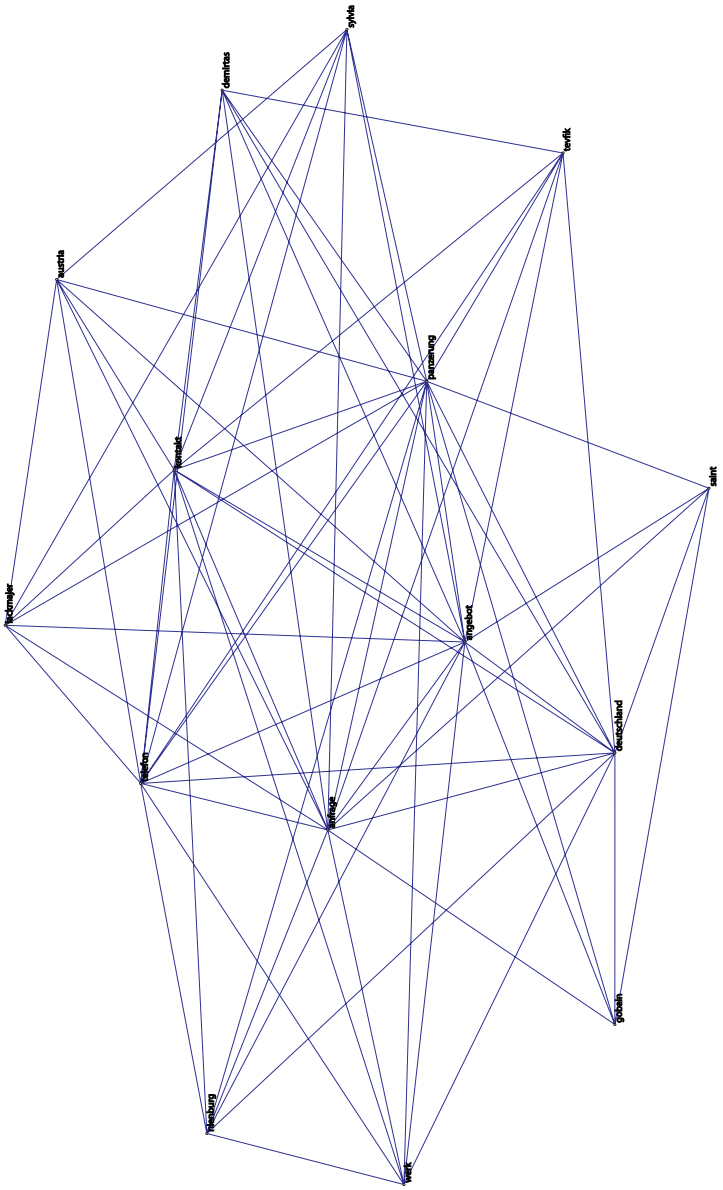


Figure G.14: Term network of the multilayer cluster related to 'German sales quotes'

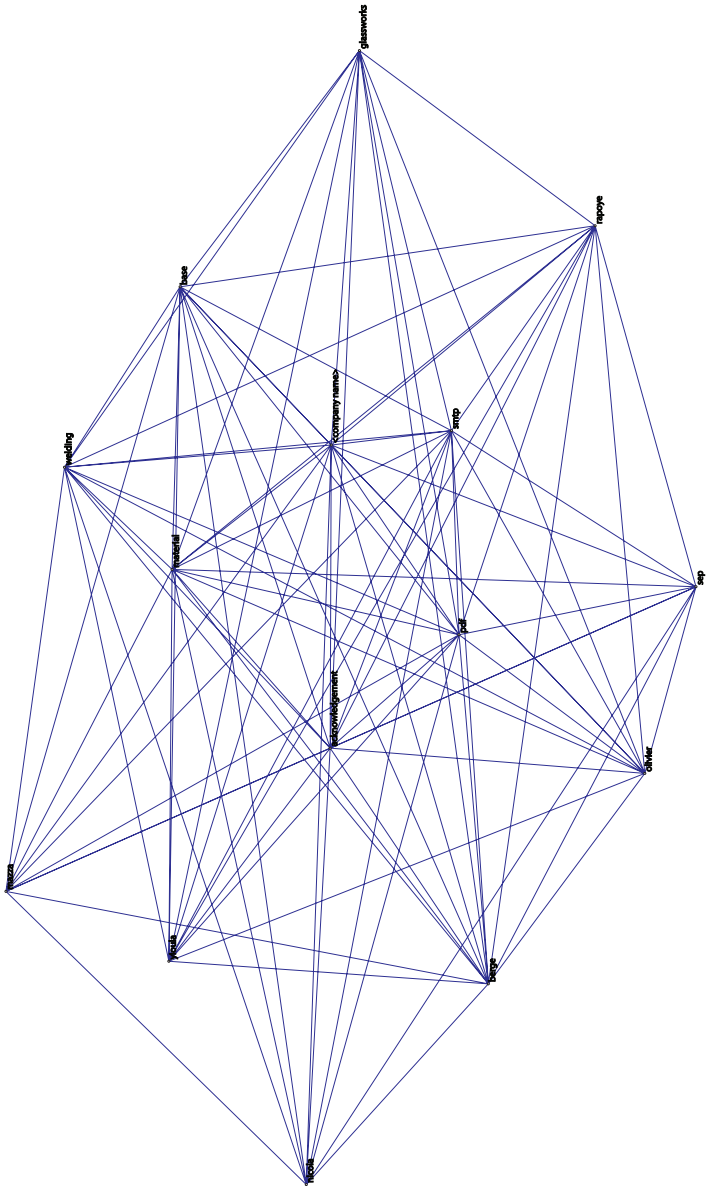


Figure G.15: Term network of the multilayer cluster related to 'English sales quotes'

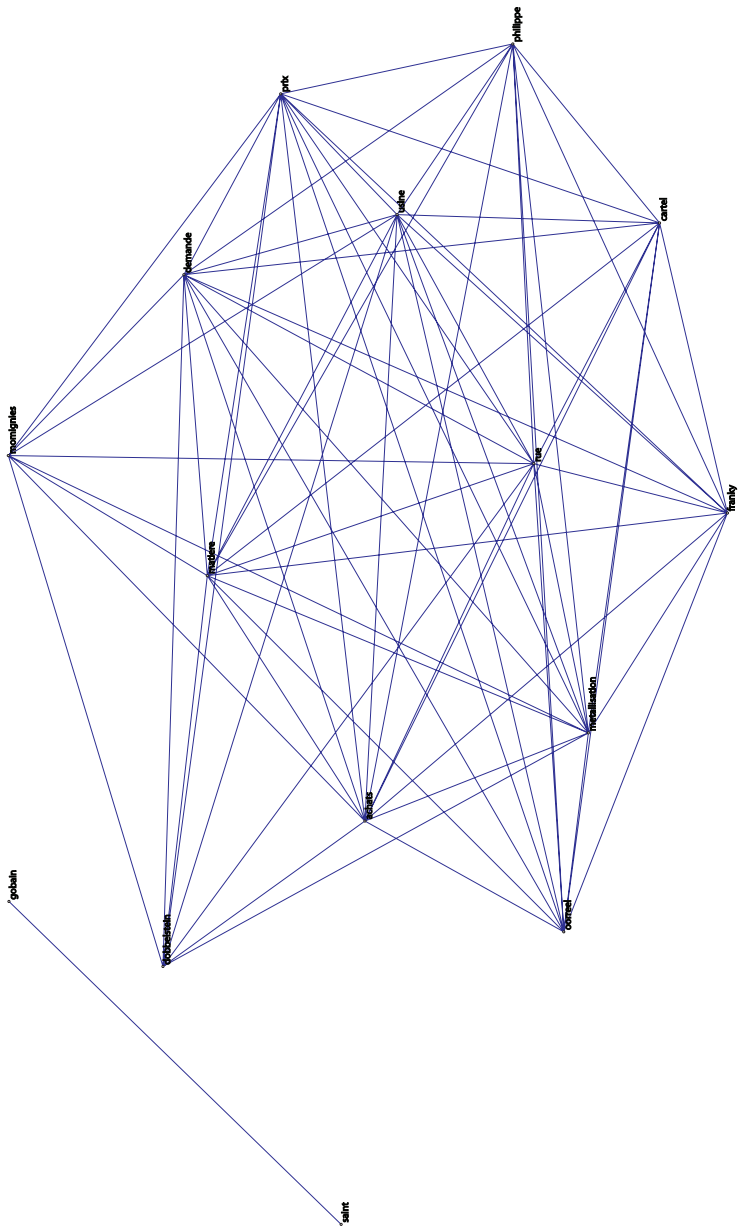


Figure G.16: Term network of the multilayer cluster related to 'French sales quotes'

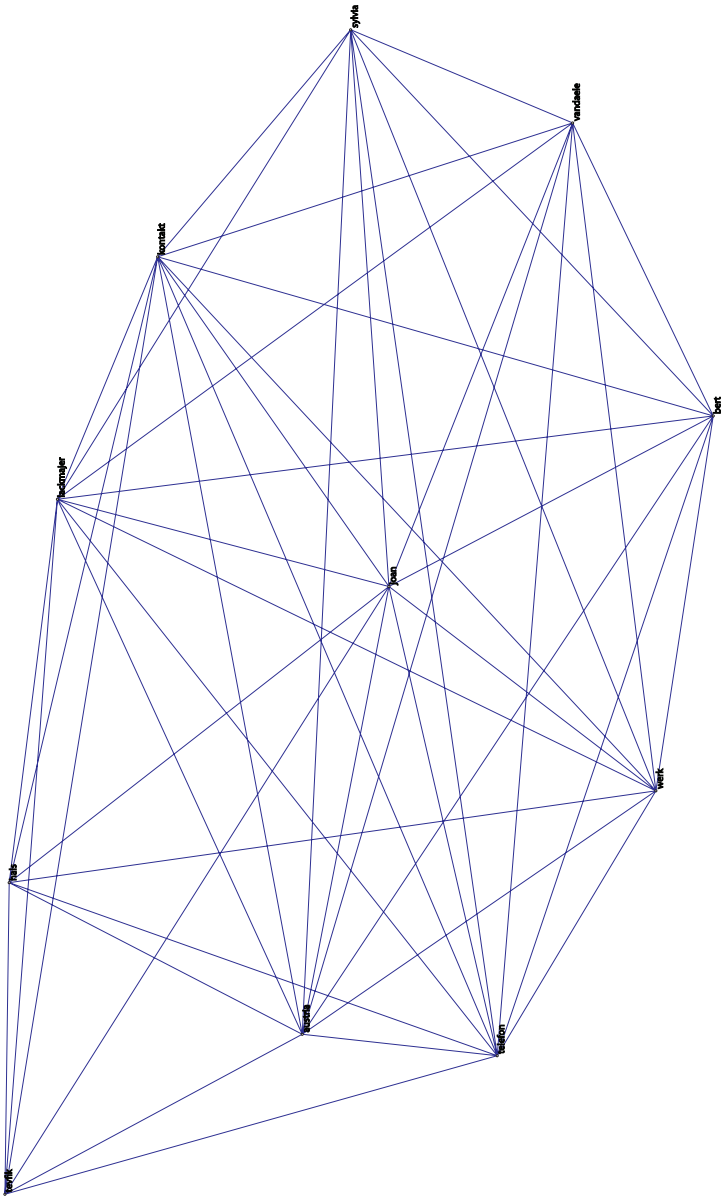


Figure G.17: Term network of the multilayer cluster related to 'German sales quotes'

```
deciding-terms = deciding-words(DTM,dictionary,threshold)
  deciding-terms ⇐ []; % Empty array for the deciding terms
  average ⇐ []
  for (i ≤ number of terms) do
    average(i) ⇐ sum(term(i))/(number of terms);
  end for
  [sort-average,indices] ⇐ sort(average);
  max-cumulative-frequencies = sum(sort-average);
  max-available-frequencies ⇐ sum(sort-average);
  k ⇐ 0;
  cumulative-used-frequencies ⇐ 0; % the already selected accumulated frequency
  while (cumulative-used-frequencies/max-available-frequencies) ¡ threshold do
    deciding-terms = [deciding-terms dictionary(indices(k)]; % add this
    important word to the list.
    cumulative-used-frequencies ⇐ cumulative-used-frequencies +
    sort-average(k);
  end while
```

Algorithm 21: Identification of the deciding words

```
MLC = MLC-identification(MLC)
  % MLC has a size of (number of documents) x (number of cluster layers+1).
  The first column of this matrix contains the different MLC-numbers of the
  documents
  d ⇐ 1; % document index
  for (d ≤ number of documents) do
    if (exists(cluster chain of document d) then
      MLCnb ⇐ getMLC(cluster chain of document d)); %compare the
      cluster-ids of d with other cluster-ids of other documents.
      %MLCnb is the MLC number of the first occurrence of an equal chain
      MLC(d(1)) ⇐ MLCnb; % MLC number of document d is set to MCLnb
    else
      MLC(d(1)) ⇐ highest MLC number +1; If no equal cluster chains exists,
      a new MLC is created
    end if
  end for
```

Algorithm 22: Identification of the multilayer clusters

```
A = co-occurrence(DTM)
  for (i≤ number of terms) do
    D_i ⇐ vector of document indices indicating the frequency of term i;
    j⇐ i+1;
    for (j≤ number of terms) do
      D_j ⇐ vector of document indices indicating the frequency of term j;
      for k≤ number of terms do
        if (D_i(k) == D_j(k) > 0) then
          A(i,j) = A(i,j)+1;
        end if
      end for
    end for
  end for
```

Algorithm 23: Construction of a co-occurrence matrix based on a DTM

List of Publications

Articles in internationally reviewed scientific journals

- Verhaegen P.A., D'hondt J., Vertommen J., Cattrysse D. and Duflou, J. R. Relating Properties and Functions form Patents to TRIZ Trends. *CIRP Journal of Manufacturing Science and Technology*, 24(5):513–523, 2009.

- D'hondt J., Vertommen J., Verhaegen P.A., Cattrysse D. and Duflou, J. R. Pairwise-adaptive dissimilarity measure for document clustering. *Information Sciences*, 180(12):2341-2358, 2010.

- Verhaegen P.A., D'hondt J., Vandevenne D., Dewulf S. and Duflou, J. R. Identifying Candidates for Design-by-Analogy. *Computers in Industry*, Accepted.

- D'hondt J., Verhaegen P.A., Vertommen J., Cattrysse D. and Duflou, J. R. Topic identification based on document coherence and spectral analysis. *Information Sciences*, Under Review.

Papers at international conferences and symposia, published in full in proceedings

- D'hondt J., Vertommen J., Cattrysse D. and Duflou, J. R. Development of a Mid-frequency Favoring Weighting Scheme for Document Clustering. *Dutch-Belgian Information Retrieval Workshop*. Leuven, 2007.

- Vertommen J., D'hondt J. and Duflou, J. R. Facilitating Product Development with the Help of Knowledge Management. *CIRP STC Design Seminar*. Berlin, 2007.

- Duflou, J. R. and D'hondt J. Applying TRIZ for Systematic Manufacturing Process Innovation: The Single Point Incremental Forming Case. *ETRIA TRIZ Future Conference*. Frankfurt, 2007.
- Verhaegen P.A., D'hondt J., Vertommen J., Dewulf S. and Duflou, J. R. Relating Properties and Functions from Patents to TRIZ Trends. *CIRP Design Conference*. Twente, 2008.
- Vertommen J., D'hondt J., Verhaegen P.A., Cattrysse D. and Duflou, J. R. An Expert Search Functionality based on Multiple-Vector User Profiles. *International Conference on Digital Enterprise Technology*. Nantes, 2008.
- Verhaegen P.A., D'hondt J., Vertommen J., Dewulf S. and Duflou, J. R. Searching for Similar Products through Patent Analysis. *ETRIA TRIZ Future Conference*. Twente, 2008.
- Verhaegen P.A., D'hondt J., Vertommen J., Dewulf S. and Duflou, J. R. Interrelating Products through Properties via Patent Analysis. *CIRP Design Conference*. Cranfield, 2009.
- Verhaegen P.A., D'hondt J., Vertommen J., Dewulf S. and Duflou, J. R. Quantifying and Formalizing Product Aspects through Patent Mining. *ETRIA TRIZ Future Conference*. Timisoara, 2009.
- D'hondt J., Verhaegen P.A., Vertommen J., Cattrysse D. and Duflou, J. R. Near-Duplicate Detection Based on Text Coherence Quantification. *European Conference on Knowledge Management*. Vicenza, 2009.
- D'hondt J., Verhaegen P.A., Vertommen J., Cattrysse D. and Duflou, J. R. Identifying Document Metadata based on multilayer clustering. *Advances in Soft Computing, Springer Berlin*, 661527–1538, 2009. Hong Kong, 2009.
- Verhaegen P.A., D'hondt J., Vertommen J., Dewulf S. and Duflou, J. R. Automatically Characterizing Products through Product Aspects. *CIRP Design Conference*. Nantes, 2010.