
WEB IMAGE CONTEXT EXTRACTION: METHODS AND EVALUATION

Inaugural-Dissertation

zur

Erlangung des Doktorgrades der
Mathematisch-Naturwissenschaftlichen Fakultät
der Heinrich-Heine-Universität Düsseldorf

vorgelegt von

Sadet Alcic
aus Leskovac

Oktober 2011

Aus dem Institut für Informatik
der Heinrich-Heine Universität Düsseldorf

Gedruckt mit der Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Heinrich-Heine-Universität Düsseldorf

Referent: Prof. Dr. Stefan Conrad
Koreferent: Prof. Dr. Michael Schöttner
Tag der mündlichen Prüfung: 24.11.2011

Dedicated to

Imène

ACKNOWLEDGEMENTS

This thesis is the outcome of my four years Ph.D. research at the Databases and Information Systems Institute of the Department of Computer Science at the Heinrich-Heine-University of Duesseldorf.

First, I would like to thank my supervisor *Prof. Dr. Stefan Conrad* for all his time, support and feedback. It was a great pleasure for me to work under his supervision. Stefan has the ability to transform a vague formulated problem into a clear and solvable question, and I always left our meetings feeling more optimistic than when I entered them. I also want to thank the second reviewer of this thesis, *Prof. Dr. Michael Schöttner*, for his interest in my work and willingness to be the second referee.

My special compliments go to my former colleagues at the database group *Johanna Vompras* and *Katrin Zaiß*. I thank Johanna for her guidance especially at the beginning of my research, and Katrin for her patient listening, the many discussions and the gratifying atmosphere she created during the time sharing her office with me.

I extend my compliments to my colleagues *Ludmila Himmelspach*, *Juwu Zhao* and *Tim Schlüter*. I acknowledge Ludmila and Jiwu for the enlightening discussions related to clustering, and *Tim* for several support and feedback and his gift to cheer me, especially in the cloudy days.

I am also grateful to my new colleagues *Magdalena Rischka*, *Thomas Scholz* and *Thi Thuy Anh Nguyen* for their help and feedback during the last writing phase.

Furthermore, I want to acknowledge *Guido Königstein*, *Sabine Freese* and *Marga Potthoff* for all their administrative assistance. I especially thank Guido for all the time he spend with me trying to fix any technical problems, which occurred during the time at the databases institute.

I am deeply grateful to my parents for teaching me the basic principles of life. I thank them for their continuous support and for giving me encouragement to tackle any problem I am faced with.

Above all, I want to thank my wife Imène for creating the best circumstances for a confident work and for enriching my life with love, security, understanding and warmth.

ABSTRACT

Images on the Web come in hand with valuable textual content on hosting web pages that can be exploited to generate image annotations. However, web documents are usually composed of contents to multiple topics and the context of an image makes only a small portion of the full text of the web page. In order to get qualitative descriptions, methods that are able to extract the image context become essential.

Existing solutions in the literature reach from simple full text extractors to intelligent approaches that perform a page segmentation as a preprocessing step. To be able to evaluate and compare the different methods, we introduce an evaluation framework that includes a ground truth dataset consisting of twelve different testing collections. The accordance between extraction output and ground truth is estimated using newly adapted evaluation measures that are a part of the framework.

Most of the existing methods are based on simple heuristics and hence in general can not deal with the variety of different web page designs. Our first approach is therefore more adaptive: it arranges first the smallest content units of a web page to possible context candidates (articles) and assigns then to each image of the web page the most suitable candidate. This approach is extended by concepts that are able to handle the two-dimensional HTML-tables that are frequently used as layout elements.

Another contribution is an image context extraction method that is based on page segmentation as a preprocessing step. By separating a web page into blocks of coherent topics, the images just can be associated with the complete text of the common block. In an extended analysis, we investigate different approaches to solve the page segmentation task by web content clustering. Different representations for web contents are combined with various clustering approaches and evaluated. The gained experience is used to build a novel clustering-based context extraction method.

Both methods achieve very good results on almost all test collections and can thus be applied as a preprocessing step in applications that can benefit from images with descriptions.

ZUSAMMENFASSUNG

Digitale Bilder im Web treten in Webseiten gemeinsam mit wertvollen Texten auf, die zur Generierung von Bildbeschreibungen genutzt werden können. Leider besteht eine Webseite in der Regel aus mehreren Inhalten zu unterschiedlichen Themen, und der Kontext eines einzigen Bildes stellt nur einen Bruchteil des Gesamtinhalts der Webseite dar. Um dennoch qualitative Beschreibungen zu erhalten, ist es notwendig Methoden zu entwickeln, mit denen der Kontext eines Bildes aus einer Webseite extrahiert werden kann.

Bestehende Lösungen aus der Literatur reichen von einfachen Extraktoren, die den gesamten Text einer Webseite übernehmen, bis hin zu intelligenteren Methoden, die als Vorverarbeitungsschritt eine Einteilung der Webseite in einzelne Bereiche vornehmen. Um die Qualität der einzelnen Verfahren ermitteln und vergleichen zu können, wurde im Rahmen dieser Arbeit ein Evaluationsframework entwickelt, das eine eigens erzeugte Testdatenmenge (Gold Standard) bestehend aus zwölf Kollektionen umfasst. Zur Bestimmung der Übereinstimmung zwischen der Ausgabe der Extraktionsverfahren und dem Gold Standard wurden geeignete Evaluationsmaße entwickelt und in das Framework integriert.

Die meisten existierenden Extraktionsverfahren basieren auf einfache Heuristiken und können daher im Allgemeinen nicht mit der Vielfalt an unterschiedlichen Webseitendesigns umgehen. Unser erster Ansatz fasst deshalb zunächst unabhängig vom Design der Webseite die einzelnen Textinhalte zu möglichen Kontextkandidaten (Artikeln) zusammen und weist dann einem Bild den geeignetsten Kandidaten zu. Dieser Ansatz hat Schwierigkeiten, wenn Tabellen als Layoutelemente verwendet werden und deshalb wird um Konzepte erweitert, welche auch mit HTML-Tabellen umgehen können.

Ein weiterer Ansatz basiert auf der Webseitenpartitionierung als Vorverarbeitungsschritt. Ist eine Webseite erstmal in ihre Teilbereiche unterteilt, kann ein Bild mit dem in seinem Bereich enthaltenen Text assoziiert werden. Wir untersuchen ausführlich mehrere Möglichkeiten, die Webseitenpartitionierung durch ein Clustering der kleinsten Inhalte einer Webseite durchzuführen. Dabei werden verschiedene Darstellungsformen für Webinhalte mit unterschiedlichen Clusteringverfahren kombiniert und evaluiert. Mit den aus dieser Analyse gewonnenen Erkenntnissen wird ein neues auf Clustering basierendes Extraktionsverfahren entwickelt.

Beide vorgestellten Ansätze liefern auf fast allen Kollektionen sehr gute Ergebnisse und können somit in vielen Applikationen, die Beschreibungen zu Webbildern benötigen, als Vorverarbeitungsschritt eingesetzt werden.

CONTENTS

Contents	i
1 Introduction	1
1.1 Image Context Sources in Web Documents	2
1.2 Challenges of Web Image Context Extraction	4
1.3 Web Image Context Applications	5
1.4 Contributions	6
1.5 Outline of this Work	7
2 Basic Terms and Concepts	9
2.1 The World Wide Web	9
2.1.1 Web Browser	10
2.1.2 Web Documents	11
2.2 Information Retrieval	16
2.2.1 Information Representation	16
2.2.2 Relevance Computation	18
2.2.3 Cluster Analysis	20
2.2.4 Evaluation of IR Methods	23
2.3 Basics of Text Analysis	26
2.3.1 Stopword Filtering	26
2.3.2 Stemming	26
2.3.3 Other Pre-Processing Methods for Text	26
2.3.4 Word Similarity	27
2.4 Summary	29
3 Related Work	31
3.1 Image Context Extraction Methods	31
3.1.1 Window-based Context Extraction	32
3.1.2 Structure-based Wrappers	36
3.1.3 Context Extraction By Page Segmentation	39
3.1.4 Summary	42
3.2 Evaluation of WICE Methods	43

3.2.1	Direct Evaluation of WICE	43
3.2.2	Implicit Evaluation of WICE	44
3.2.3	Summary	45
3.3	Applications of Web Image Context	45
3.3.1	Web Image Retrieval	45
3.3.2	Web Image Annotation	47
4	Extraction of Web Image Context	49
4.1	The Attributes of Web Image Context	49
4.2	A Formalization of the WICE Task	53
4.3	WICE as an Information Retrieval Task	54
4.4	Summary	56
5	Evaluation Framework for WICE	57
5.1	Why a new Evaluation Framework?	57
5.2	Evaluation Framework Design	59
5.2.1	Web Documents	60
5.2.2	Gold Standard	62
5.2.3	Extraction Methods	64
5.2.4	Performance Measures	65
5.2.5	Framework Output	67
5.3	The Evaluation Framework in Practice	67
5.3.1	Time Analysis	68
5.3.2	Precision, Recall, F -score	69
5.4	Summary	73
6	Web Image Context Extraction by nearest Article Detection	75
6.1	The Idea behind WICE by Article Detection	76
6.2	Article Detection	77
6.2.1	Content extraction	77
6.2.2	Content Classification	78
6.2.3	Grouping Contents to Articles	79
6.3	Image-to-Article Mapping	81
6.3.1	The TABLE Problem	82
6.3.2	Measuring the Relationship of Contents in 2 Dimensions	83
6.4	Evaluation Studies	84
6.4.1	Practical Issues of Article-based WICE	84
6.4.2	Results	85
6.5	Summary	89

7	Web Image Context Extraction by Page Segmentation	91
7.1	Page Segmentation by Clustering	92
7.1.1	Problem Formulation	92
7.1.2	Web Content Representation	93
7.1.3	Distance between Web Contents	93
7.1.4	Clustering Methods	98
7.1.5	Experimental Studies	100
7.2	Image Context Extraction by Web Content Clustering	106
7.2.1	Properties of the DOM-based Distance	106
7.2.2	A new Formulation of the WICE Problem	107
7.2.3	1D-Clustering based on Distance Threshold	108
7.2.4	Threshold estimation	108
7.2.5	Visual Representation of Clustering	109
7.2.6	Evaluation	110
7.3	Summary	114
8	Conclusion and Future Work	115
8.1	Summary and Conclusions	115
8.2	Future Work	117
	References	119
	List of Figures	129
	List of Tables	133
	Appendix	135
A	Evaluation Results for Simple-Structure-Document Collections	135
B	Evaluation Results for Complex-Structure-Document Collections	138

1

INTRODUCTION

In the recent years, the World Wide Web (or the *Web* for short) has become an integral part in our lives comprising billions of web documents with information to almost every conceivable topic. The information itself is presented by different media types, such as *audio*, *image*, *text* and *video*.

After text, image is the most prevalent media type on the Web. For this reason, tools and applications that allow to manage this large data repository became inevitable. Commercial web search engines have a great success in indexing textual web contents, since they can build on a solid work of research in the fields of Information Retrieval and Text Mining done over the last decades. However, compared to text, the organization of images by means of the semantics they depict is much more complicated [DJLW08].

While humans recognize objects depicted in images almost unconsciously, the automatic understanding of images is still one of the most challenging tasks in computer vision. The main difficulty is known as the *Semantic Gap* [GR95] which exists between the internal *low-level* representation of an image and its *high-level* semantic concepts. Until now, there is no general approach to automatically compute the semantics of pictures from underlying pixel information on a satisfying level of quality. Figure 1 contains a more vivid representation of the Semantic Gap.

In contrast to isolated images, images on the Web often come in hand with valuable textual descriptions on hosting web pages that share common semantics with the images. These textual descriptions can be exploited to generate suitable annotations with common information retrieval methods for text contents [FSC04, ZXJQH05]. In this way, the direct confrontation with the Semantic Gap is avoided, since the image understanding problem can be reduced to the well known text indexing. Furthermore, indexes based on textual description instead of pixel statistics allow keyword-based

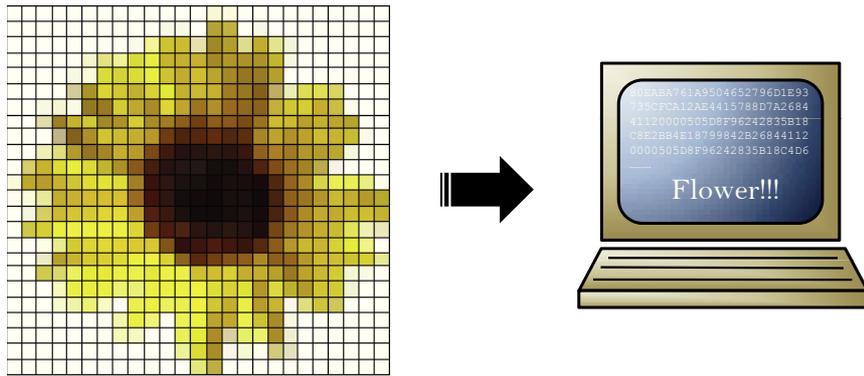


Figure 1.1: From low-level pixels to high-level semantics: the visual Semantic Gap.

query formulation, which offer a more natural way for a user to specify his/her needs.

However, web pages are only partially structured and the context information of images is not explicitly given. Since contextual descriptions make only a small portion of web page content, reliable methods to detect and extract the image context are emergent and will be the main target of this work.

In this introducing chapter, we will first present and discuss the different image context sources included in web documents in Section 1.1. Section 1.2 introduces the challenges to web context extraction methods that are postulated by the Web. Section 1.3 contains a brief overview of applications that can profit from image context, and finally, this chapter is concluded by a summary of the main contributions in Section 1.4 and an outline of the thesis in Section 1.5.

1.1 Image Context Sources in Web Documents

Images embedded in web documents can profit from valuable textual descriptions in their hosting documents. However, in contrast to web document retrieval, where the complete content of a web page is relevant, web image context mostly makes only a small part of the web page. Different elements of a web page come into consideration to share common semantics with an embedded image of this web page; namely, *image URL*, *page title*, *alternative text* (image ALT-attribute), *full text*, or the *associated text* passages (or image paragraph). In the following, we want to introduce and discuss the benefits and the shortcomings of possible image context sources.

The **image URL** contains the filename of the image and its path on the web server. If properly chosen, the filename can include keywords that describe the image content. The path can also be useful: some web designers organize web images in hierarchical folders, e.g., “http://www.example.com/people/german/poet/Heinrich_Heine.jpg”

allowing to build taxonomy-based annotations derived from image URLs [SC97]. However, well organized URLs are rare on the Web. Moreover, there are hardly web images with manually assigned filenames. Many web images possess generated filenames (e.g., “img0001.jpeg”) that are not suitable for our needs.

The **page title** is usually a short description of the web page content and therefore can be an important hint to the image semantics. For example, imagine a web page with a short biography and a portrait of Heinrich Heine titled by “The life of Heinrich Heine”. In this example, the page title is a good context provider for the portrait image. Unfortunately, many web pages do not comprise only one topic, but many diverse topics (good examples are the start pages of most news domains). In such cases, the page title is hardly related to the images of that page, because it is either too general (e.g., “Breaking News”) or just contains information to the global web site (e.g., “Welcome to our Site”), which in both cases would generate a misleading image description.

The **alternative text** or the value of the *ALT*-attribute corresponding to the *IMG*-tag was originally intended as a short image description to be displayed in environments (e.g., command line web browsers) where the image can not be rendered. If properly used, the alternative text might be the most appropriate image context source, since it is directly related to the image and can be easily extracted. However, web authors usually do not create eligible descriptions, and texts like “img1” generated by web designing tools are met very often. Rather, although the *ALT*-attribute is obligatory [Spe99], many images do not have an alternative text at all. Feng et al. [FSC04] observed that only 21% of images they have analysed in their empirical studies possess an *ALT*-attribute. Hence the benefits of alternative text as image context provider are limited.

As **full text**, we consider the complete displayable text of a web document. Usually, text and image contents can both be used to describe similar information in different ways and therefore web articles and images on web pages may be semantically cohesive. However, web documents are usually cluttered with multi-topic contents and different images on this page can belong to different particular contents, but in this approach each image would be associated with all contents. This results in an “excellent” recall, but the precision of the assigned information is very low and thus misleading. Furthermore, web pages contain several noisy contents, like advertisements, layout and navigational elements which do not contribute to the information content at all and further decrease the precision.

Recognizing the shortcomings of full text context, the idea of **associated text** is to select only a small text portion of full text, which is semantically related to the image content. For example, some web images possess an image caption with a small description of the image context, which is similar to the alternative text and hence

may own the same quality properties. Even if an explicit image caption is not present, the text surrounding an image is considered to share semantics with the image to a high degree and thus can be valuable. Several studies [CCS⁺04, FSC04, Alc07] on the quality and reliability of different context sources in matters of ability to describe web images show clear advantage for the associated text paragraphs. However, the associated text is not explicitly given and proper extraction methods have to be found. In the next section, we will summarize the problems that context extraction methods are confronted with.

In the remainder of this thesis, we will equally use the terms *associated text* and *web image context* to refer to the same subject.

1.2 Challenges of Web Image Context Extraction

Web Image Context Extraction (*WICE*) has many benefits since it delivers valuable image descriptions without any human effort. However, if not properly extracted, the image context can lower the quality of involved applications. The detection of images and associated context on hosting web pages is confronted with several challenges that are mainly set by the Web:

- **Large volume.** The amount of data available on the Web is tremendous. One can find information to almost every imaginable topic on the Web. But the exponential growth of the Web poses scaling issues that are difficult to cope with. Therefore *WICE* methods have to be simple and in the first instance fast.
- **Cluttered web pages.** Web documents are cluttered with multiple contents to different topics. The identification of the particular content blocks is inevitable in order to guarantee precise image descriptions. Moreover, beside the main content, web pages include noisy contents such as advertisements that make *WICE* even more complicated.
- **Dynamic contents.** Information and representation structure changes very quickly. It is estimated that 40% of the data on the Web changes every month [Kah97]. Because layout structure is also volatile, *WICE* can not rely on strict extraction rules since they have to be adapted whenever the layout changes.
- **Heterogeneous data.** Information on the Web is heterogeneous. Among different web pages, contents are structured using different (designing) patterns. Hence, we also characterize web pages as semi-structured documents. Even for a single web page, the context of an image can vary from small caption to a long article. *WICE* methods have to be adapted to handle such heterogeneous structures.

- **Data quality.** The Web allows everyone to publish information while there is no editorial institution, which proves the correctness of the shared contents. As a consequence, there is no guarantee on the quality of the contents. In other words, we could find a web image with a cat while the surrounding context tells something about elephants. This problem can not be solved by an extraction method and shows the limits of the web image context approach.

It is important to keep these problems in mind on the way to new solutions for web image context detection, as well as to realize the quality limits of web image context in applications that could profit from image context. Some of these applications will be presented in the next section.

1.3 Web Image Context Applications

A variety of applications could benefit from image context extracted from web documents. Some application areas found in the literature are (without any claim to completeness):

- Web image retrieval,
- Web image annotation,
- Visual concept learning,
- Web page redesign for embedded devices, and
- Accessibility improvement.

The most obvious application of web image context is **web image retrieval**. In this application the textual description associated with an image is used to generate an index for the image. The indexing process can be accomplished using standard methods from text retrieval. Usually, the text is first split into words, then the words are stemmed to basic terms, and finally stop-words are eliminated. User queries are matched against these index terms and the corresponding images are finally presented as results.

Web image annotation is another important application. The aim of this application is to find keywords that describe the depicted objects best. Image annotation itself can be used as input to many other applications. For example, annotations can be used for image retrieval as described in the first application. But also other applications like image organization or image browsing are possible by clustering images with similar annotations. Furthermore, image annotations serve as input for the next application, the learning of visual objects and concepts.

Learning of visual concepts describes the process of finding correlations between annotations and visual descriptions of images. If a correlation occurs more often, one can assume that the image depicts the object described by the keywords. In this way translations of visual to textual concepts are learned and can be stored in a visual dictionary. Later, this dictionary can be applied to generate annotations for new unlabeled images.

Embedded devices as mobile phones or PDAs are restricted in hardware resources. Especially, they have small screens and can only display a small excerpt of a web page. By **web page redesign**, we denote the process of restructuring a web page to fit into smaller displays. The usual preprocessing algorithms for this purpose try to filter noisy contents and to display that main content in a condensed structure that fits on the screen restrictions. Using WICE as preprocessing delivers new possibilities: a web page can be displayed as a photo album first with all content images, and then the user can choose an image from this list to read the corresponding article.

WICE can **improve the quality of accessibility applications**. Visually impaired people can profit from the descriptions belonging to images. Especially, when images do not have an alternative text (Feng et al. [FSC04] estimated that only 21% of images they have downloaded possess ALT-text), it can be of great benefit to have contextual captions. These captions can be presented to impaired people instead of the images and they can better follow the web contents.

1.4 Contributions

This thesis deals with several topics related to the WICE problem. Additionally to a quite complete overview of methods and techniques proposed so far in the context of WICE, the thesis makes four major contributions that are new to the field.

Our first contribution addresses the **evaluation of WICE**. While there is only a few research work published in the context of WICE, there is even less work done in the context of WICE evaluation. Most of the conducted evaluation studies treat WICE as a part of a main application and in this way measure its quality only implicitly. However, in order to be able to justify objectively which WICE method is best suited for a certain application, an appropriate evaluation environment for WICE becomes essential. We fill this gap by presenting a modular evaluation framework, consisting of twelve test collections with many annotated web documents and appropriate similarity measures to compute the accordance of ground truth and extracted context. Furthermore, we integrate many existing WICE methods from the literature in this framework in order to test its applicability and also for the later comparison with newly introduced extraction methods.

During the investigation of the methods proposed in other research work, we noticed that these methods are usually based on simple heuristic rules. This approach is not necessarily bad, but in general simple heuristics are not able to handle the high variety of web pages of the Web. Therefore, methods that can adapt to various web page designs without relying on specific web content patterns are required. Motivated by this demand, our second contribution is **a new adaptive article-based WICE method**, which divides the WICE task into two main steps. In the first step, the images of the web page are ignored and the textual contents are grouped into candidate articles. The grouping is done only based on the logical structure of the textual contents. In the second step, the computed articles are mapped to the web images as image context using a distance measure. The mapping is difficult when HTML table constructs are used for layout definition: for this purpose, an extended version of the distance measure based on a two-dimensional model is proposed.

Another adaptive approach to WICE is to apply a Web Page Segmentation (WPS) algorithm to partition a web page into semantic and structural cohesive blocks, and then to associate the images with text contained in the common block. Existing WPS approaches are usually targeting the partitioning of web documents into basic parts such as header, footer, sidebar and main content. However, in order to suit our purpose, they would require further modifications, which enable them to detect image blocks of different granularities. Our third contribution is therefore **a new approach to WPS by web content clustering**. In particular, we combine different distance measures based on different web content representations with several clustering methods and evaluate the WPS output. As result, our newly defined DOM-based distance and an extended density-based clustering algorithm achieve the highest quality.

Finally, the insights gained from the third contribution are pursued to develop **a new WPS-based WICE method**, which is our fourth contribution. To be precise, we harness the property of the DOM-based distance, which maps the web contents to a one-dimensional space, in order to develop a new dynamic and adaptable clustering method for web contents.

1.5 Outline of this Work

This thesis is organized as follows: **Chapter 2** provides the basic background knowledge, needed to follow the approaches and solutions presented in this thesis. In particular, we introduce some basic terms and concepts in the fields of the Web, Information Retrieval and Text Mining.

Chapter 3 introduces the related work in the fields of WICE and Page Segmentation, and additionally presents different applications and systems that take the advan-

tages of web image context.

In **Chapter 4**, the WICE problem is formalized and two practical interpretations of the formal definition are given, which consider the extracted context either as a sequence of characters or as a sequence of text node contents. After that, the point of view is changed and the WICE task is regarded from the IR perspective while several concepts are reinterpreted.

Next, in **Chapter 5** the evaluation framework for WICE methods is presented. We first illustrate the necessity for an evaluation environment by describing the weaknesses of the existing evaluation attempts. Then, each component of the proposed solution is detailed. And finally, the practical feasibility of the framework is tested with several existing WICE algorithms, which have been presented in related work.

Chapter 6 introduces our approach to WICE by article detection. The algorithm consists of two main steps, namely the article detection and the image-to-article mapping, which are successively introduced. At the end of the chapter, the WICE method is evaluated using the proposed framework and compared to the other methods from the literature.

Chapter 7 approaches the WICE problem by clustering-based page segmentation. Initially, we introduce different representations and corresponding distance measures for web contents. These are combined with common clustering approaches and tested for their suitability for detecting image context blocks best. Based on the results, a new WICE method is proposed, which uses a newly introduced, adaptive clustering method to group web contents.

Finally, **Chapter 5.4** concludes this thesis with a summary of the main findings and a short overview of future works.

2

BASIC TERMS AND CONCEPTS

This study of *Web Image Context Extraction* (WICE) deals with several concepts from different research fields that are applied under different circumstances. The most prevalent topics in this thesis are the *World Wide Web* and *Information Retrieval*. The World Wide Web (or the Web for short) comprises the raw web pages to be analyzed and thus determines the structure of the input data and the methods to access the information in this data. On the other side, since the WICE task can be considered as a special Information Retrieval (IR) task, it provides many useful concepts and methods that can be transferred to WICE (as will be shown in Chapter 4). Since the web image context is represented as text, there are some useful methods related to text representation and analysis that will be presented in Section 2.3. The main scope of this chapter is to familiarize the reader with the basic terms and concepts from the mentioned research fields and in this way to supply him with enough background information required to follow the later chapters of this thesis.

2.1 The World Wide Web

The World Wide Web – started as an obstinate vision of Tim Berners Lee in CERN in 1989 – has become the biggest and the most widely used information source today. The official definition describes the Web as “a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents” [Hug94]. In simpler words, the Web consists of billions of interconnected documents (called web pages) that were created by millions of people. The web pages are hosted on millions of *web servers* over the world and the access is provided through the Internet. In this way, the Web follows a typical Client-Server paradigm: usually a *client* initiates

a query by specifying the *Uniform Resource Locator* (URL) (an unique address that determines the exact location of a document on the Web) of the web document and sending a request to the corresponding web server. The server then processes the query and sends the requested document back to the client.

Since its inception, the Web has dramatically changed the way how people communicate with each other and how information is published and retrieved. Before the Web, getting information means asking someone or reading a lend or purchased book. Today, every information is just a few clicks away for free and from any place [Liu07].

The information itself is enclosed within the web pages and can be accessed by using a browser application, which also allows non-expert users to request and display the desired information through a graphical user interface.

2.1.1 Web Browser

The Web offers a very large pool of information that can be accessed using a simple and interactive application – the *web browser*. A browser usually provides a *Graphical User Interface* (GUI) that allow users to view the documents and to execute tasks interactively by simple point-and-click-actions.

The information itself can be represented by different media types – such as text, image, video and audio – and a web browser usually provides appropriate extensions to display common media file formats. Information in books or traditional documents is sequentially presented and can be hierarchically structured by sections, subsections or paragraphs. Nonetheless, humans usually process such information linearly. In opposite to traditional documents, the Web is a *Hypermedia-System* consisting of *hyperdocuments* that are interlinked with each other. Following these *links*, users can jump to different parts of the document or even to another document that can be anywhere else on the Web. The browser offers all methods needed to navigate to other locations by simply clicking on the text with a corresponding link. Links are not only available for text but also for the other media types as images, video clips and audio files.

Another important function of a web browser is the information visualization. A web browser usually interprets the source code of a web document and automatically reloads all needed resources to render the document and present it to the user. During this rendering process each content part is assigned some visual properties, e.g. an exact position in the browser panel and the dimensions of the content elements. Figure 2.1 depicts an excerpt of the CNN home page with some visual features (absolute position of the upper-left corner and width/height dimensions) of selected content elements. Such information can be very useful for web content mining applications since the visual properties include some clues about the relations between distinct objects.

Getting the visual properties of web contents needs the source code to be processed



Figure 2.1: Visual features of selected Web Contents

by a layout engine. A free and open source layout engine is *Gecko* [Gec11] that is used by many applications provided by the Mozilla Foundation (e.g., the Firefox Browser) and others. Gecko supports many open Internet standards and is able to display web documents in arbitrary web applications [Gec11]. Since Gecko is written in C++ and relies on XPCOM [XPC11] objects, the embedding in Java applications as the ones developed in this thesis can be accomplished using the JavaXPCOM API [Jav11].

Visual properties can be useful in web content mining applications. However, web documents need to be completely rendered, which can be very time consuming since not only the documents source but additionally all contained media objects have to be loaded in advance. In a testing scenario with documents of different length and structure the processing time for rendering the documents has increased from 6 to 114 times compared to only processing the raw source of the document.

2.1.2 Web Documents

In the context of WICE, web documents provide the input data and are thus the central objects of interest in this thesis. In opposite to human users, WICE approaches the documents from another perspective. Additionally to the rendered representation of the documents which is presented to humans in browsers, document processing applications usually analyze the source code or some derived structure. For this reason, the different

formats, technologies and methods, on which web documents are based, are briefly presented in the given context.

Document Representation and (X)HTML

The source code of a document on the Web is represented in an HTML (Hypertext Markup Language) format. The commonly used formats are the SGML oriented HTML 4.01 [RHJ09] and its XML based successor XHTML 1.1 [AM10], which were both recommended by the *World Wide Web Consortium* (W3C). HTML is a markup language that outlines parts of a document with special properties. For marking, HTML defines special elements – called *tags* – that surround certain parts of the plain text. Tags can be hierarchically nested, and we are talking about parent and child element, referring to the outer and respectively the inner element.

HTML documents are usually composed of two parts: a *header* part containing information to the document itself, such as a document description that can be processed by search engines, and a *body* part including the majority of information. The latter comprises the real content that is created and structured by an author using different markup tags and later displayed in a browser.

The major difference of HTML and XHTML is the stricter syntax of the latter. For example, XHTML is case-sensitive for element and attribute names, while HTML is not. Furthermore, since XHTML relies on XML, all elements are required to be closed, either by a separate closing tag or using the self closing syntax (e.g., `<br / >`). On the other hand, the HTML syntax permits some elements to be unclosed because either they are always empty (e.g., `<input>`) or their end can be determined implicitly (e.g., `<p>`). Besides the syntactic differences, there are additionally some behavioral differences. A typical example is: a fatal parse error in XML (such as an incorrect tag structure) aborts the document processing immediately, while HTML does not expect structural valid code. Moreover, HTML interpreters in browsers usually try to adjust structural errors in HTML.

Markup languages were originally developed to distinguish between the *document structure* (e.g., header or paragraph) and document presentation (e.g., underlined or bold) and in this way to alleviate document processing by external applications. Knowing the document structure can be very worthy for many information extraction applications. However, in HTML both principles (document structure and document presentation) are supported and can be mixed at the same time, which causes that extraction of structural information becomes more complex. Moreover, authors of web documents often misuse structural elements for presentation and vice versa (e.g., a bold tag is used to mark a header).

A more powerful tool to describe the presentation of (X)HTML documents are

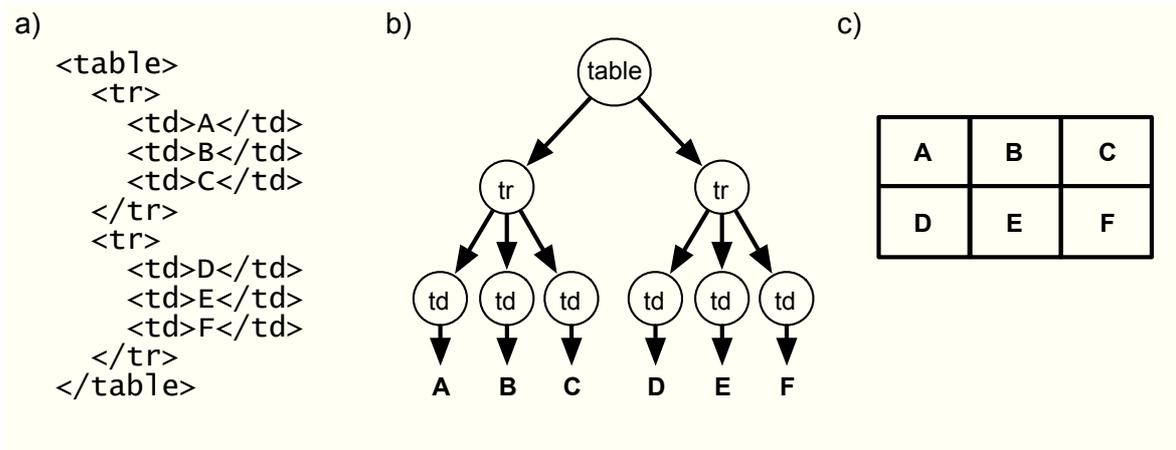


Figure 2.2: An Example HTML snippet a) HTML, b) browser and c) DOM representation.

Cascading Style Sheets (CSS) [LBL98]. The W3C has developed CSS primarily to enable the separation of document content from document presentation and thus improve content accessibility. CSS allows to define styling rules (e.g., layout, fonts or colors) for each element within the document.

Today, CSS belongs to open Internet standards and thus is supported by common web browsers.

DOM

The *Document Object Model* (DOM) is a standard API (Application Programming Interface) described and extended in the W3C specifications [ABC⁺98], [HHW⁺00] and [HHW⁺04]. The primary goal of the W3C group was to define an object model for accessing and modifying the content, the structure and the presentation of (X)HTML documents. It is important to notice that the specification of DOM does not include any concrete implementation in a programming language but just provides a description of the interfaces.

DOM models a web document corresponding to its strictly nested tag structure, i.e. it exploits the *structure tree* of a document and maps it to its own internal data structure. Especially, the parent-child relation of HTML elements remain equal and thus the result of the mapping process is again a tree, called *DOM tree*. Particular markup tags of a (X)HTML document can be addressed by their position in the DOM tree or by an especially selected identifier (e.g., in HTML `id="example"`). Figure 2.2 shows an example HTML snippet with the corresponding DOM tree.

The `node` interface plays a central role in DOM since it defines attributes and methods for navigating and modifying elements in the DOM tree. Simple traversing through the structure tree is possible with the following node attributes:

- `firstChild`,
- `lastChild`,
- `nextSibling`,
- `previousSibling` and
- `parentNode`.

The first two attributes link to the first (or respectively the last) child node of the actual node. They make it possible to navigate on a higher tree level. The sibling attributes might be used to navigate on the same tree level and deliver the next/previous sibling of the actual node. Finally, the `parentNode` attribute links to the parent node that is on a lower tree level.

The methods for manipulating the structure tree are:

- `appendChild()`,
- `removeChild()`,
- `insertBefore()`,
- `replaceChild()` and
- `cloneNode()`.

The first method appends the input child node at the end of the child node list. The `removeChild()` method removes the passed node from the child node list. The other methods are self explanatory.

The different elements of the structure tree of a document are represented by one specific sub-interface of the `node` interface. Figure 2.3 shows a part of the interfaces hierarchy specified by DOM. While, as we already mentioned before, the `Node` interface is in the central position – it defines the general attributes and methods that are inherited to all other sub-interfaces – the graph shows also that all common HTML objects correspond to a specialization of the `HTMLElement`.

The DOM specification has been implemented in many programming languages. The most widely used implementation is possibly part of *JavaScript* which is a scripting language that can be embedded directly in HTML documents and thus manipulates contents of the document while they are shown in the browser.

The DOM tree representation is suitable for different tree-based algorithms that can be applied to mine the contexts. Different approaches to WICE and page segmentation that utilize the tree structure of DOM will be presented in the next chapter. Furthermore, one contribution in this thesis is a DOM-based distance function relying on the structural clues provided by the DOM tree (refer to Chapter 7).

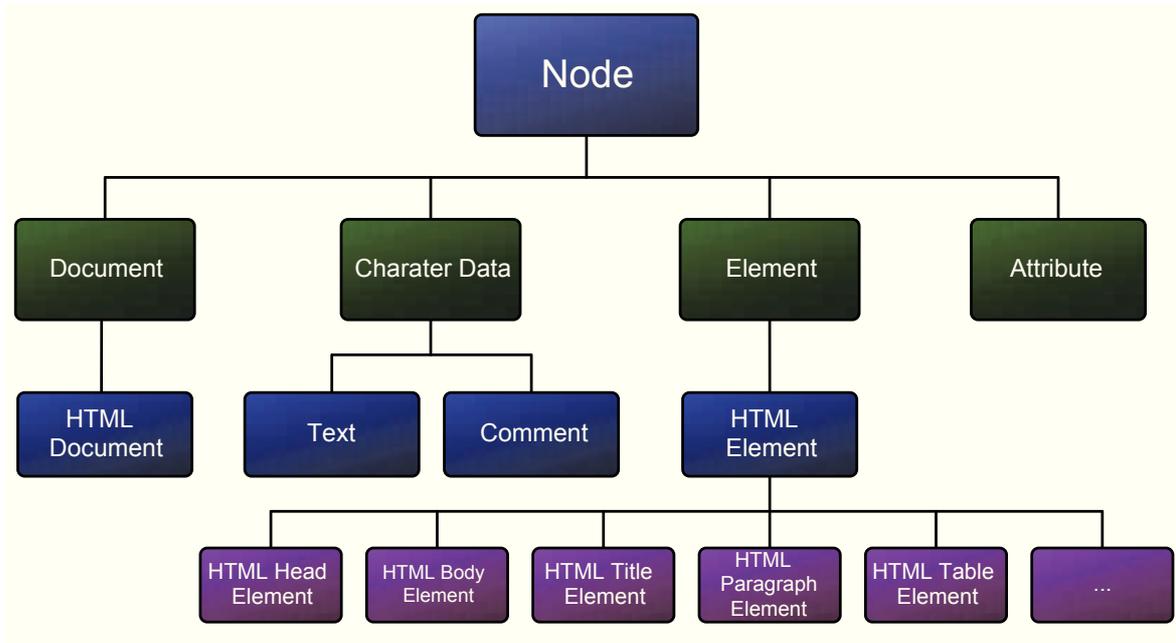


Figure 2.3: Part of the hierarchy of DOM interfaces.

HTML Parser

Parsers are programs that analyze (scan) a text and interpret its particular tokens based on predefined syntactical rules (*formal grammar*). In the context of web documents, a parser translates the HTML based string representation of a document to a data structure relying on the DOM specification. A parser is thus an inherent component in applications, which process HTML or XML documents.

Besides the implementation of the standard interfaces defined in DOM specifications, HTML parsers provide useful methods for *extraction* and *transformation* of web contents [Osw06].

Extraction includes all the information retrieval tasks that are meant to process only a part of the original HTML document. The prevalent applications used by web search engines are *text extraction* and *link extraction*. Link extraction can also be used for crawling through web pages or for ensuring that links on a web page are valid. Other extractable resources in web pages are images and sound files. Extraction can also be used for *site monitoring*, where the facilities of parsers go beyond simplistic diffs. The Java HTML Parser [Osw06], which was used in the implementations of this thesis provides some additional helpers like filters.

On the other hand, transformation encompasses all document handling where the output still remains a web page. For example, a *site capturing* application which stores remote web pages to a local disk requires some links to be modified in order to be still valid. Other applications, such as *ad removal* or *site censorship* mainly filter

the detected target contents while the rest of the web pages remains the same.

A big challenge for web page processing applications is posed by dirty and ill-formed HTML, which is usually found on the Web [Nik11]. Although many documents do not comply with any HTML standard, they are not rejected by web browsers what in turn encourages web page creators not to stick to existing standards. For any serious processing of such documents, it is necessary to first cleanup the clutter and bring the order to tags, attributes and text. There are many HTML parsers (e.g., JTidy [STG⁺11], JSoup [Hed11], HTMLParser [Osw06], HTMLCleaner [Nik11]) that provide suitable methods to turn incomplete and invalid HTML to well defined XHTML.

2.2 Information Retrieval

Information Retrieval (IR) is a broad discipline dealing with the *representation, storage, organization of* and *access to* documents in a repository [BYN99]. A *document* in this context is not restricted to text documents but can be an arbitrary information item carrying some kind of information (e.g., a text, an image, a complete web page, or just an element in a web page).

At the beginning of the IR process there is always a specific *information need* that is specified by a *user query*. The user query is commonly formulated using a list of *keywords*. The task of an IR system is to find the set of documents in the collection that is *relevant* to the query. In contrast to *Data Retrieval* where the sought information is structured and stored explicitly, information in, e.g., images is implicit and other techniques are required to express relevance that go beyond exact matching. For this purpose, both, the query and the retrievable items as well, have to be transformed into a representation which allows their comparison. One of the main characteristics of an IR system is thus the way it represents the information items.

2.2.1 Information Representation

Information in documents managed by IR systems is not explicit and usually can not be ascertained without human interpretation. For example, images are represented by numerical values that express color intensities of each pixel of the image and the pixel groups get a semantic meaning by human interpretation. The lack of a high level semantic description in document representation is known as the Semantic Gap that we already mentioned in the introduction. In order to make documents retrievable, it is thus necessary to (automatically) derive a more semantic description from their raw representation.

A common way to represent user query and documents is the *bag of words* or *terms* concept. That is, a document or a query is described by a set of distinct terms, while

sequence or position of terms in a document are ignored. Following this, a term simply corresponds to a word whose semantics describe a particular topic of the document. It is important to note that a term here may not be a word in natural language but also any kind of statistic value extracted from the document (e.g., the average color of an image). Further, each term is associated with a weight that emphasizes the occurrence of a specific term or property. A more formal description of this representation is as follows [Liu07]:

Definition 1. *Given a collection of documents (or information units) D , let $V = \{t_1, t_2, \dots, t_{|V|}\}$ be a finite set of distinctive terms t_i in the collection. Usually the set V is referred to as vocabulary of the collection, while $|V|$ corresponds to its size. Each term t_i appearing in a document $\vec{d}_j \in D$ is associated with a weight $w_{ij} > 0$. Terms $t_i \in V$ that are not contained in document \vec{d}_j are assigned a weight $w_{ij} = 0$. A document \vec{d}_j can thus be represented by a term vector,*

$$\vec{d}_j = (w_{1j}, w_{2j}, \dots, w_{|V|j}),$$

where each weight w_{ij} corresponds to the term $t_i \in V$, and quantifies the level of importance of t_i in document d_j . In general, the sequence of the terms in the vector is not significant, but once determined should not be changed among the documents.

There are different ways to set the term weights that depend on the retrieval model applied. For example, in the traditional *Boolean Retrieval Model* the term weight $w_{ij} (\in \{0, 1\})$ is 1 if the term t_i is present in document \vec{d}_j , and 0 otherwise. One shortcoming of this model is that it lacks of modeling any significance of specific terms. In contrast, the *Vector Space Model* is more flexible since it does not limit the weights to $\{0, 1\}$ but allows any number. For term weight computation, there are different weighting schemes in the literature.

The most well known weighting scheme for text documents is *TF-IDF*. It comprises two measures, the *Term Frequency (TF)* and the *Inverse Document Frequency (IDF)*, which are defined as follows (based on [Liu07]).

Let f_{ij} be the frequency count of term t_i in document \vec{d}_j . The *normalized term frequency* tf_{ij} of term t_i in document \vec{d}_j is given by:

$$tf_{ij} = \frac{f_{ij}}{\max\{f_{1j}, f_{2j}, \dots, f_{|V|j}\}},$$

where the maximum is computed over all terms that appear in document \vec{d}_j . As mentioned above, $|V|$ corresponds to the size of the vocabulary. If t_i does not appear in \vec{d}_j then $tf_{ij} = 0$.

Let N refer to the total number of documents in the collection and df_i be the number of documents in which t_i appears at least once. The *inverse document frequency* idf_i of term t_i is given by:

$$idf_i = \log \frac{N}{df_i}.$$

The idea here is that terms appearing in many documents are less important and thus should be lower weighted. Finally, the TF-IDF term weight w_{ij} is given by:

$$w_{ij} = tf_{ij} \times idf_i.$$

We have now briefly discussed how information in documents can be represented. On the other hand, it is inevitable to represent the user query in an appropriate format, as well. There are different possibilities to represent a query that depend on the application. However, the most convenient way is to use the same vector representation as used for document representation, since then the comparison of query to documents becomes more practicable. A simple method to build the query vector \vec{q} for a keyword based query is, for example, to set the terms weights of the terms appearing as keywords to 1, and the others respectively to 0. Another querying concept (known as *Query by Example* (QBE)) is to provide an example document and to use its derived representation as query. In both cases, the query representation equals the document representation.

Once the query and the documents are transformed into an appropriate representation, in the next step the relevance of a document with regard to the given query can be computed using a *similarity* (or a *dissimilarity* function). Both of them will be formalized next and some prototype functions will be presented.

2.2.2 Relevance Computation

In contrast to data retrieval, where relevance is *discrete* – meaning that a document is either relevant or not – in the context of IR, the relevance is continuous. As a consequence there are documents that are more relevant to a query than others and the result set becomes more a *relevance ranking* of documents ordered by their degree of similarity.

One advantage of the vector space model is that it maps documents and query into a vector space, which is a well-grounded concept in the field of linear algebra. In particular, it provides a variety of tools and methods allowing to mathematically define the relevance between documents and query. But before presenting concrete similarity (or dissimilarity) measures, we will first provide a general definition to both concepts.

Similarity Measure

Let D be the domain of documents (or document representations). A similarity measure is a function $sim : D \times D \rightarrow \mathbb{R}^+$ that takes two documents as input and associates them with a positive numerical value (usually normalized to $[0, 1]$). Without loss of generality, one of the documents might be the user query. Additionally, a similarity measure has to fulfill the following properties:

1. *Positiveness*: $\forall a, b \in D : sim(a, b) \geq 0$,
2. *Maximality*: $\forall a, b, c \in D : sim(a, a) \geq sim(b, c)$,
3. *Symmetry*: $\forall a, b \in D : sim(a, b) = sim(b, a)$.

The first property of positiveness follows directly from the definition of the value set of a similarity function. The value of 0 stands for the minimal similarity that two arbitrary documents can possess. In contrast, the maximality property expresses that the highest similarity is that between two identical documents. Finally, the symmetry property tells us that it does not mind from which direction we compute similarity, since both deliver the same value.

There is a variety of similarity measures for different document representations and applications. The most widely known similarity measure for documents in the vector space is the *cosine similarity*, which is defined as follows.

Example 1. Let $\vec{d}_j, \vec{d}_k \in D$ be two document vectors in a vector space D . The cosine similarity sim_{cos} which corresponds to the cosine of the angle between the document vectors \vec{d}_j and \vec{d}_k is given by:

$$sim_{cos}(\vec{d}_j, \vec{d}_k) = \frac{\langle \vec{d}_j, \vec{d}_k \rangle}{\|\vec{d}_j\| \cdot \|\vec{d}_k\|} = \frac{\sum_{i=1}^{|V|} w_{ij} \cdot w_{ik}}{\sum_{i=1}^{|V|} w_{ij}^2 \cdot \sum_{i=1}^{|V|} w_{ik}^2}.$$

The cosine similarity does not consider any synonymy or hyponymy relationships between the terms of the documents. In Section 2.3.4, we will introduce the concept of taxonomy and present some word similarity measures from the literature that exploit the synonymy and hyponymy relationships of the taxonomy to compute a more accurate similarity.

Distance Function

A distance function (or *dissimilarity measure*) is quiet opposite to a similarity measure and expresses how unlike two given documents are. The distance values are non-negative numbers, while a distance value of 0 is assumed to be the maximum similarity. Further, the distance value is usually not normalized to a specific interval as similarity measure is.

Given a pair of document representations $a, b \in D$, let $dis(a, b)$ be a binary function that takes the document pair (a, b) as input and associates it with a positive real number:

$$dis : D \times D \rightarrow \mathbb{R}_0^+.$$

This function may possess different properties:

- *Identity of indiscernibles*: $\forall a \in D : dis(a, a) = 0$,
- *Positivity*: $\forall a, b \in D, a \neq b : dis(a, b) > 0$,
- *Symmetry*: $\forall a, b \in D : dis(a, b) = dis(b, a)$,
- *Triangle inequality*: $\forall a, b, c \in D : dis(a, c) \leq dis(a, b) + dis(b, c)$.

Definition 2. A non-negative binary function $dis : D \times D \rightarrow \mathbb{R}_0^+$ which fulfills the properties of identity of indiscernibles, non-negativity, symmetry and triangle inequality is called a distance function. A distance function in connection with a domain D is called a (distance) metric.

Example 2. One of the most popular distance metrics on the vector space is the Euclidean distance. For two documents $d_j, d_k \in D$ the Euclidean distance is given by

$$dis_{Euc}(\vec{d}_j, \vec{d}_k) = \left(\sum_{i=1}^{|V|} (w_{ij} - w_{ik})^2 \right)^{\frac{1}{2}}.$$

A distance function can easily be turned into a similarity function that maps a pair of documents to an interval $[0, 1]$. One possible transformation function applied in the implementations to this thesis is given by

$$f(x) = \frac{1}{1 + x}.$$

There is a variety of other transformation functions that mainly differ in how significant small changes next to a distance of 0 are treated. A good overview of many example transformation functions is given in [Sch05].

2.2.3 Cluster Analysis

The clustering or grouping of documents is a task that originally comes from the field of *Data Mining* (DM) and *Machine Learning* (ML). Recall, that by “documents” we do not mean only text documents but any data objects that carry information.

In the field of IR, clustering has a variety of applications, reaching from a simple categorization of documents by topic, to advanced indexing methods that enhance the

processing time needed for query answering by comparing the query only to cluster representatives. Depending on the application, a variety of techniques has been proposed in the past. The scope of this section is not to present concrete clustering techniques but to introduce some basic concepts of clustering. Clustering algorithms applied in this thesis will be illustrated later at an appropriate place.

By definition, cluster analysis denotes the process of placing documents that are similar (or related) to each other in a same cluster and separating documents that are unlike (or unrelated) to each other in different clusters. The greater the homogeneity within a cluster and the greater the difference between clusters, the better the clustering is assumed to be.

The notion of what exactly constitutes a cluster is not well-defined and can vary among different applications. This problem is better illustrated by the example in Figure 2.4 showing eighteen points and two different ways of dividing them in clusters. While in Figure 2.4 b) the clusters are broader, Figure 2.4 c) shows a partitioning to finer grained clusters. Both clusterings are justifiable by human perception. Consequently, this example illustrates that the definition of a cluster is imprecise and depends on the nature of the data and the desired results [TSK05].

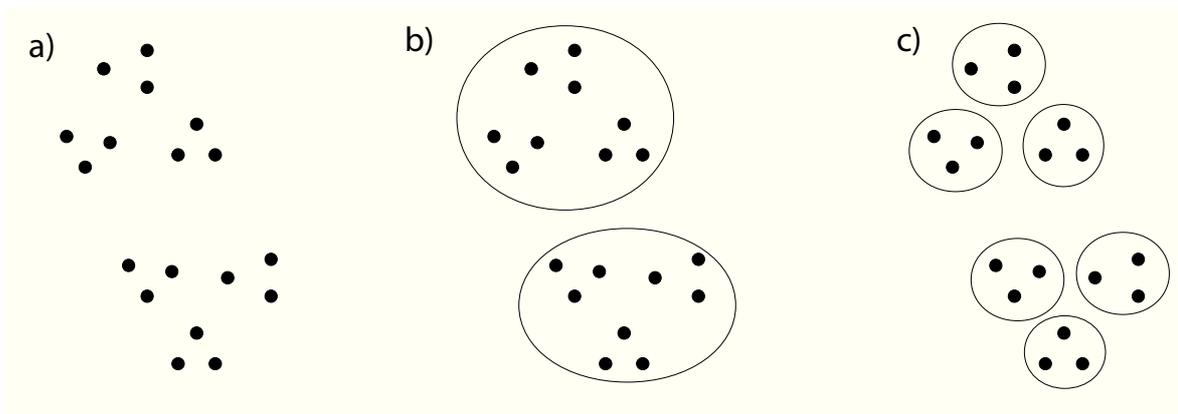


Figure 2.4: What is the perfect Clustering? Different Ways to group Points to Clusters

The result of cluster analysis is a set of clusters, which we refer to as *clustering*. There are different types of clusterings that can be obtained using different techniques. In the following, we distinguish three different types: hierarchical vs. partitional, exclusive vs. overlapping, and complete vs. partial [TSK05].

Hierarchical vs. partitional The distinction here is whether the set of clusters is nested or unnested. *Partitional clustering* divides the set of documents into non-overlapping subsets (clusters), while each document is in exactly one subset. Both clusterings in the example of Figure 2.4 are prototypes of partitional clustering.

In contrast, a *hierarchical clustering* allows clusters to have subclusters, while the set of clusters is organized by a tree structure. The leafs at the bottom of the tree are clusters usually consisting of only one document and the root of this tree is the cluster containing all the documents. Each node (cluster) in the tree encompasses all documents of its children (subclusters). A partitional clustering can be derived by cutting the tree at an appropriate level.

Exclusive vs. overlapping The clusterings in Figure 2.4 are both *exclusive*, as they assign each point to a single cluster. However, there are also situations where the documents could reasonably be put in more than one cluster simultaneously (e.g., if particular clusters represent certain topics and a document includes more than one topic, then the document can be placed in more clusters). As a result, we get an overlapping clustering. A more flexible approach is fuzzy clustering where each document belongs to all clusters (fuzzy sets) with a membership weight that is between 0 (absolutely does not belong) and 1 (absolutely belongs).

Complete vs. partial As *complete clustering*, we understand a clustering in which every document of the data set is assigned to at least one cluster. If not, we talk about a partial clustering. There are many situations where only a partial clustering is reasonable, e.g., when the data set contains noisy documents referred to as *outliers* that do not belong to any real cluster.

Besides the different types for clustering, we can also distinguish different ways to represent clusters that have been proved practical in the past. More precisely, the different cluster types that will be discussed next are well-separated, prototype-based and density-based [TSK05].

Well-separated An arbitrary document of a *well-separated* cluster is more similar (or closer) to any document of the same cluster than to any other document from another cluster. Assuming the clusters as points in a two dimensional plane, well-separated clusters can be of any shape.

Prototype-based A document of a *prototype-based* cluster is more similar (or closer) to the prototype of its own cluster than to any other prototype of the other clusters. For documents with numerical attributes the prototype is usually computed as the average of all documents of the cluster. If the document attributes are, e.g., categorical then the cluster prototype is the medoid, i.e., the most representative document of the cluster. Obviously, such prototype-based clusters are convex.

Density-based A density-based cluster is a dense region of documents surrounded by a region of lower density. In some approaches, the minimum density a cluster

should possess is predefined by a threshold value. As a consequence, the obtained clustering can be partial since documents within a region of low density are detected as outliers or noise. Density-based clusters can also be of any shape.

2.2.4 Evaluation of IR Methods

To estimate the quality of IR methods, a variety of evaluation measures has been introduced. Common evaluation metrics for IR and DM tasks that were prevalently used in this thesis are *accuracy*, *precision* and *recall*, as well as the inferred *F-score*. In the following, we will introduce these and related evaluation metrics and discuss the way in which they measure the quality of IR methods.

An IR system usually gets an user query as input and returns the relevant documents of the collection as result. To some extent, the IR process can be described as a binary classification task, since for each document the IR system decides whether to place it in the positive (relevant) class or the negative (irrelevant) class. In this way, the evaluation of IR can be reduced to evaluation of a binary classification.

Each document in the collection has been either classified correctly or incorrectly based on a *Gold Standard* (or *ground truth*) which defines the desired classification output. The accuracy of a classification task is now simply defined as the ratio of the number of correctly classified documents to the number of all documents in the collection. An accuracy value of 1 means optimal classification. The accuracy measure does not consider the incorrectly classified documents and can thus be insufficient in some situations. If the data is highly unbalanced (as in IR, where the desired documents are mostly only a small fraction of the entire document collection), e.g., 99% of documents are in negative class, then a classifier can achieve 99% accuracy without doing anything but simply classifying every document as “negative”. This is, however, useless.

In such a case, precision and recall are more suitable because they measure how precise and how complete the classification on the positive class is. Before presenting the concrete definition of the measures, it is convenient to introduce the confusion matrix (see Table 2.1, on which the measures are based. The confusion matrix assigns the classification result of a single document to one of four categories: *true positive*, *true negative*, *false negative* and *false positive*. The *TP* (or respectively *TF*) value corresponds to the number of documents that were correctly classified as “positive” (or respectively as “negative”) by the classifier, while *FN* (or respectively *FP*) indicates the number of documents that were falsely classified as “negative” (or respectively as “positive”). The confusion matrix itself is already an evaluation concept that provides an overview of the correspondence of the ground truth and the automatic classification.

Before presenting the definition of the recall and precision measure, we switch the

Table 2.1: Confusion Matrix of a binary Classification

		Automatically Computed Classification	
		positive	negative
Ground Truth Classification	positive	TP	FN
	negative	FP	TN

point of view back from classification to the IR task to clarify the relationship of the four categories in the confusion matrix to basic concepts of IR. Given a query on a document collection, the relevant documents correspond to the positive items in the ground truth (the first column in the confusion matrix), while the answer documents provided by the IR system can be interpreted as the positive classified items (the first row in the confusion matrix).

Definition 3 (Precision and Recall). *For a given query q , the set of answer documents that are returned by an IR system are denoted as A . Further, for the same query q there is a set of relevant documents R in the collection that can be obtained from the ground truth. For this configuration, precision p and recall r are defined as follows:*

$$p = \frac{|A \cap R|}{|A|} \left(= \frac{TP}{TP + FP} \right)$$

and

$$r = \frac{|A \cap R|}{|R|} \left(= \frac{TP}{TP + FN} \right).$$

The formula of precision is ill-defined if the system returns an empty answer set (denominator is 0). In this case the precision is conventionally set to 1 in the sense that the answer set does not contain any irrelevant results. In the same manner the formula of recall becomes ill-defined when R is empty, that is, the collection does not contain any document that satisfies the query. Here the recall is also set to 1 with the explanation that the answer set contains all relevant documents (which were none here).

Precision in the IR perspective quantifies the fraction of correctly computed answers in the complete answer set, i.e. how precise the computed answer set is. Otherwise, recall measures the fraction of correctly computed answers in the set of relevant items. Both values range in $[0, 1]$ while high values are desired.

In some IR applications, the number of documents in the answer set can be manually

determined by adjusting certain parameters. In this way, precision and recall values can be computed for different numbers of answers. However, optimizing both metrics at the same time can be a difficult task. On the one hand, precision obtains the best result when the answer set is empty, on the other hand, recall is optimal if the answer set includes all documents. Obviously, recall and precision behave contradictory in terms of size of retrieved documents set.

For some IR applications, especially these with a huge document set, the number of related documents can not be estimated due to the high effort. In such applications the computation of the recall is not possible.

One evaluation method that combines both measures and presents them in a visual manner are *precision-recall diagrams*. Being able to control the size of the answer set, allows us to compute precision values at different recall levels and plot them as a curve in a diagram where the x-axis is the recall and the y-axis is the precision. The curve is commonly plotted using 11 standard recall levels, 0, 0.1, 0.2, ..., 1. The optimal curve is parallel to the x-axis at a precision of 1. We can plot different curves for different queries and different retrieval algorithms and compare them visually.

The evaluation of precision and recall depends on the application and the goals we want to achieve: if the user need should be just satisfied quickly then a high precision in the first results is more important than a complete result set. Otherwise, if the user wants to get an overview of the most documents to one topic, then we are interesting in getting high recall values whereas false positive results are acceptable.

If both measures, precision as well as recall, should be treated with equal importance then the F-score measure which computes the harmonic mean of precision and recall can be applied.

Definition 4 (F-score). *Let r be the recall value, and respectively p be the precision value in a retrieval scenario. The F-score (or F1-measure) is defined as follows:*

$$F\text{-score} = \frac{2 \cdot p \cdot r}{p + r} \left(= \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \right).$$

As for precision and recall, the F-score reaches an optimum at a value of 1 and the F-score is ill-defined, if both precision and recall are 0 (the answer set has no relevant documents). In this case, the F-score is commonly set to 0, as well, since the harmonic mean should be 0.

A property of the harmonic mean of two numbers is that it tends to be closer to the smaller of the two. Therefore, getting a high F-score expects both precision and recall to be high [Liu07].

2.3 Basics of Text Analysis

2.3.1 Stopword Filtering

The basic objects of text are the words which describe information in a natural language. However, the information value of the particular words can vary highly and thus has an high impact on the application using text. As *stopwords*, we refer to frequently used and insignificant words in a language that are applied for sentence construction but do not carry any information content. Typical stopwords are articles, prepositions, conjunctions and some frequent occurring pronouns. There are many online available lists of stopwords for different languages. For example, the following words might be contained in an English stopword list [Liu07]:

a, about, an, are, as, at, by, for, from, how, in, is, ...

Stopwords should be filtered before a text is passed to other processing steps such as indexing. In the same way, the stopwords contained in a query should also be removed before starting retrieval.

2.3.2 Stemming

A word in a natural language has different syntactical forms. Their usage depends on the context of the sentence. Verbs are conjugated in terms of tense or person, while nouns, adjectives and pronouns are declined depending on case, number and gender. For example, the word “computing” and the word “compute” have both the same ground form “to compute”. However, in a retrieval scenario where a document contains the first flexive and the query the second, the document would not appear in the result set, since the words do not match exactly. Regarding the evaluation metrics that were introduced in the last section, the consequence is a lower recall.

Stemming tries to avoid this problem by reducing different flexives of a word to a common root (*stem*). A stem refers to a segment of a word that is obtained after cutting its suffixes or prefixes. For example, the words “defining”, “defines”, and “definition” are reduced to “defin” which actually is an artificial word. In this way, many forms of a word can be matched in retrieval, which improves the recall and further reduces the size of the index. There are different stemming algorithms for different languages. The probably most popular stemmer for English is the Porter’s stemming technique [Por97], which is based on a set of substitution rules.

2.3.3 Other Pre-Processing Methods for Text

Besides the proposed techniques for text preprocessing, there are some other methods that are usually applied, like modifying the letter case, special treatment of numbers

and the filtering of punctuation marks.

The letter case is usually considered not important in retrieval applications and thus is completely ignored by, e.g., bringing all letters into lower case. However, this method should be applied with caution, since there are some cases where the letter plays an important role – such as names of entities – and the modification of the letters can completely change the meaning of the word. For example, the operating system “Windows” would get a completely different sense by changing the word to “windows”.

Numbers present a highly informative content when they are presented in a context. However, during the indexing process, the context in which the specific terms are placed gets usually lost. Thus the information of numbers in a text gets lost and they become less significant. For this reason, typical IR applications remove numbers from text during the indexing process.

Punctuation marks are used in texts to mark different parts and thus increase the readability. The correct application of punctuation marks is regulated by the grammar of the language. However, these marks seem to be less worthy for indexing and thus are usually ignored. Nonetheless, there are similar circumstances as for the letter cases, where the punctuation marks are part of phrases and filtering them can completely change the sense of the words. For example, the phrase “state-of-the-art” gets a completely different meaning if it is modified to “state of the art” since then each word would become an individual term in the index.

2.3.4 Word Similarity

Many languages contain words that are syntactically different but semantically similar to some extent. For a pair of words, we distinguish three similarity relations:

The *equivalency* relationship (or *synonymy*) between two syntactically different words means that both words describe the same semantic concept and can be used arbitrarily. For example, in English the words “car” and “automobile” are synonyms.

Hierarchical relationships between words describe that one term is a *broader term* (*hyperonym*) or respectively a *narrower term* (*hyponym*) of the other. In English, the word “vehicle” is a broader term of “car” and “electric car” is a narrower term.

Associative relationships exist between words that are neither equivalent nor hierarchically related. For example, “car” and “bike” are associatively related, since both are subterms of “vehicle”.

In such circumstances, the comparison of text based on word-matching is not sufficient and techniques that additionally consider the semantic similarity of words are

required. Information about the relationships of different words of a language is stored in a *thesaurus*. A popular dictionary and thesaurus for the English language is *WordNet* [MBF⁺90], which additionally to a definition comprises the different relationships of words by organizing them hierarchically in a *taxonomy*. In the following, we will utilize the term of “*concept*” instead of “word” to refer to an entry in a taxonomy.

There is a variety of similarity measures for concept pairs based on a taxonomy. We will introduce the *Jiang&Conrath* [JC97] and the *Lin* [Lin98] similarity measures that proved most efficient in time and accuracy in [CM05] and thus were applied in this thesis. However, both of them build on the Resnik [Res95] similarity measure, which therefore will be introduced first.

Definition 5 (Resnik’s Similarity). *For two concepts c_1 and c_2 , let the function $LCS(c_1, c_2)$ return the Least Common Subsumer (LCS) of the concepts in a given taxonomy. Then the measure introduced by Resnik returns the Information Content (IC) of the LCS of the two concepts c_1 and c_2 :*

$$sim_{res}(c_1, c_2) = IC(LCS(c_1, c_2)),$$

where IC is defined as:

$$IC(c) = -\log(P(c))$$

and $P(c)$ is the probability of encountering of an instance of concept c in a large corpus. We have used the British National Corpus (BNC) [Uni11] in our implementations.

The idea behind Resnik’s similarity is to measure the “generality” of the LCS of two concepts. However, this measure does not take any individual properties of the particular concepts into account. The similarity of two concepts a and b that are direct subconcepts of the LCS is the same as the similarity of any other subconcepts of a and b .

Recognizing the shortcomings of Resnik’s measure, the following two concept similarity measures additionally include the properties of the concepts.

Definition 6 (Jiang&Conrath’s Similarity). *Let c_1 and c_2 be two concepts, then the similarity measure proposed in [JC97] is as follows:*

$$sim_{j\&c}(c_1, c_2) = \frac{1}{(IC(c_1) + IC(c_2)) - 2 \cdot IC(LCS(c_1, c_2))}.$$

The similarity measure of Jiang&Conrath computes the inverse of the difference of the IC values of the particular concepts and twice the LCS. In this way, the properties of the concepts are involved in the computation. The more specific the subconcepts are regarding the LCS, the lower the similarity gets. The formula is ill-defined, when both

concepts are the same (then the denominator becomes 0) and the similarity should be maximal, which equals infinity, since the range is not restricted to a maximum.

The next similarity measure which was introduced by Lin normalizes the similarity to a range of $[0, 1]$.

Definition 7 (Lin's Similarity). *For two concepts c_1 and c_2 , Lin defines the concept similarity as follows:*

$$sim_{lin}(c_1, c_2) = \frac{2 \cdot IC(LCS(c_1, c_2))}{IC(c_1) + IC(c_2)}.$$

2.4 Summary

This chapter has introduced the basic concepts and methods of the Web and the IR task that are needed to understand the next chapters. While the Web sets all preconditions for the main task of WICE by defining the formats of the input and providing different methods to process the input, IR can be interpreted as a generalization of the WICE task – this will be shown in Chapter 4 – and thus comprises many useful concepts that can be transferred to the WICE task. In particular, the concepts of similarity and distance, the clustering of documents based on similarity (or distance), evaluation of IR applications and some special processing methods for text which all play a great role in WICE, were briefly introduced.

3

RELATED WORK

Images on the Web come along with valuable contextual information on their hosting web pages. Recognizing these benefits, many researchers started to build first image retrieval applications [STC97, OBMCP00, FSC04, ZXJQH05] based on the descriptions gathered from image context. However, the success was limited by new challenges posed by the nature of the Web: web pages are cluttered with noisy contents to multiple topics that can not easily be distinguished from the contextual information of images. As a result, many WICE methods with different characteristics were proposed in the recent years.

Section 3.1 starts by describing the principal techniques to WICE, followed by a detailed introduction of the concrete extraction methods applied in existing research work. WICE a preprocessing task in many applications and thus has barely been evaluated on its own. In Section 3.2, we introduce the different evaluation methods conducted so far in the context of WICE. Finally, some interesting applications in the field of web image retrieval and web image annotation are briefly addressed in Section 3.3.

3.1 Image Context Extraction Methods

Web pages are semi-structured documents and thus the context of images embedded in web pages is given only implicitly. Nevertheless, to profit from the benefits of web image context appropriate extraction algorithms are inevitable.

Before introducing related work to image context extraction, it is necessary to describe the problem at least informally. The aim of a WICE method is: given a web page and an image (which is contained in the web page) as input, a WICE method

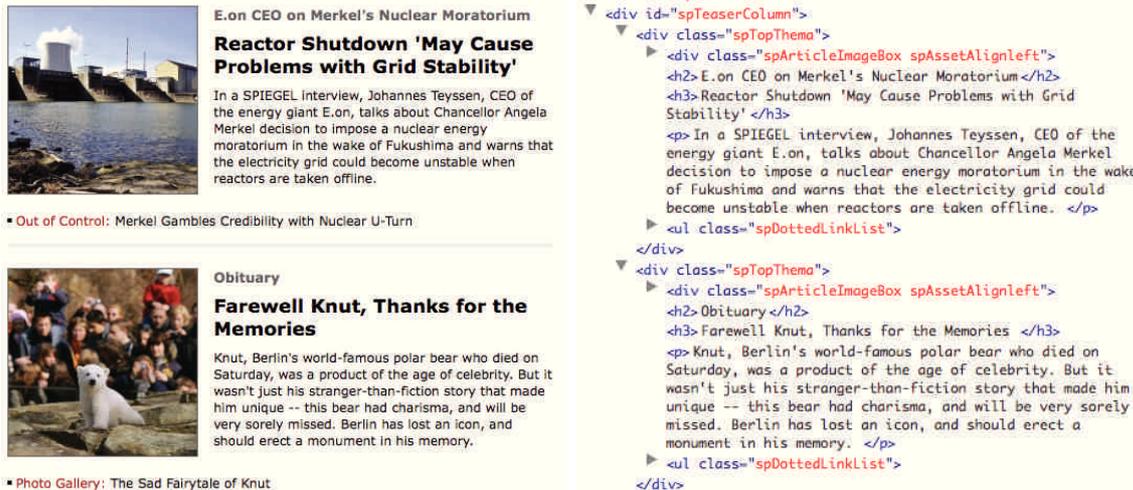


Figure 3.1: Spiegel-Online [Spi11] Excerpt: Browser (left) and DOM (right) Representation

estimates the textual descriptions from the given web page, which are associated with the image. A more detailed description of WICE will be presented in Chapter 4.

There are three main approaches to WICE proposed in the literature: *window-based* approach, *structure-based wrappers* and *page segmentation*. The window-based approach is a heuristic method that extracts text surrounding the image. Different variants of this method have been applied that usually build on different assumptions in respect to length and importance of the surrounding text. The structure-based wrappers can be seen as more adaptive heuristic rules that incorporate structural information from HTML code of the web page to estimate the borders of image context. And finally, the idea of page segmentation methods is to partition a web page into segments of common topics first, and then to associate each image with the textual content of the segment it belongs to.

Figure 3.1 shows an excerpt of an online news page with two short articles containing images on the left, and the corresponding DOM tree on the right, that will be referred to during this section to exemplify the behaviour of the particular context extraction methods.

3.1.1 Window-based Context Extraction

A fast and therefore commonly used method to estimate the context of web images is the window-based method. In this subsection, the *N-Term Window* algorithm that has been introduced in [CCS⁺04, SCS99] is presented first, followed by a variant of this method that uses a shifted and dynamic window as applied in [FSC04].

Algorithm 1: N-Terms Window

Data: Web document d , image I , window size N
Result: Set T of terms surrounding the image

```

1  $S \leftarrow \text{getImagesAndTerms}(d)$ ;
2  $i \leftarrow \text{indexOf}(I, S)$ ;
3 for  $k \leftarrow (i - \frac{N}{2})$  to  $(i + \frac{N}{2})$  do
4   | if  $S[k]$  instanceOf term then
5   |   |  $T.add(S[k])$ ;
6   | end
7 end

```

Window of Surrounding Terms

In the rendered version of a web page, web image context is usually placed in the neighborhood of the corresponding image (as depicted in Figure 3.1). Therefore, the idea behind this algorithm is to extract text that surrounds the image in the HTML source code as image context.

The algorithm of this approach is quite simple and proceeds as follows. As input, we provide the web document, an image of this document and a windows size N . In the initialization step (line 1), the document is transferred into a sequence of terms and images S . The terms are the particular words of the textual content of the given web page and the images correspond to the images in the document embedded by `` tags. The elements in sequence S are ordered by their position in the original HTML code of the document. Further, we denote the i -th element of S as $S[i]$. The transformation can easily be accomplished by a linear scan of the HTML code.

In the next step (line 2), the position of the image in the sequence S is determined and stored as index i . This step is necessary in order to know where to position the window frame in the next step.

The main part of the algorithm (lines 3-7) is a loop which iterates over S from $S[i - \frac{n}{2}]$ to $[i + \frac{n}{2}]$ and collects the visited terms. Additionally, if the window exceeds the borders of S , the iteration index has to be adapted (not contained in the pseudocode algorithm). Provided that the element $S[i]$ is an image, the web image context is estimated as in Figure 3.2.

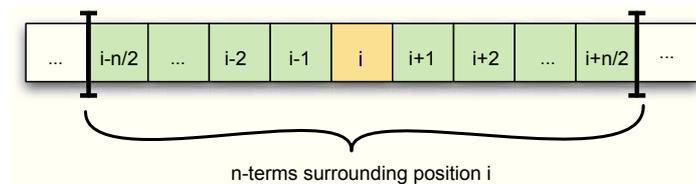
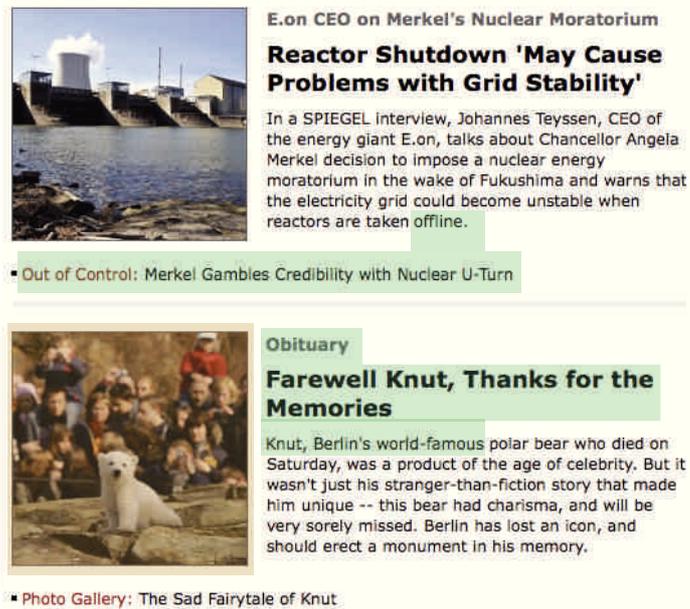


Figure 3.2: Frame of n terms surrounding an image in a list of terms and images.

Example 3. Application of the surrounding terms extraction method on the web page excerpt shown in Figure 3.2 with $n = 20$ delivers the following image context for the second image showing the baby polar bear “Knut”:



E.on CEO on Merkel's Nuclear Moratorium
Reactor Shutdown 'May Cause Problems with Grid Stability'
 In a SPIEGEL interview, Johannes Teyssen, CEO of the energy giant E.on, talks about Chancellor Angela Merkel decision to impose a nuclear energy moratorium in the wake of Fukushima and warns that the electricity grid could become unstable when reactors are taken offline.

▪ **Out of Control: Merkel Gambles Credibility with Nuclear U-Turn**

Obituary
Farewell Knut, Thanks for the Memories
 Knut, Berlin's world-famous polar bear who died on Saturday, was a product of the age of celebrity. But it wasn't just his stranger-than-fiction story that made him unique -- this bear had charisma, and will be very sorely missed. Berlin has lost an icon, and should erect a monument in his memory.

▪ **Photo Gallery: The Sad Fairytale of Knut**

In this example snippet, the green shaded box represents the extracted image context.

Example 3 shows the introduced extraction method in practice. The parameter n determines the size of the frame of terms surrounding the image and has to be estimated. Souza-Coelho et al. [CCS⁺04] have used different frame sizes in their evaluation studies while $n = 20$ has performed best (experiments have been carried out with $n = 10$, $n = 20$ and $n = 40$). Sclaroff et al. [SCS99] have applied a frame of 30 terms, while the number of terms before the image was set to 10 and respectively after the image to 20 terms, which can be well justified by an observation of Feng et al. [FSC04] stating that for 73% of the examined images the context appears after the image and for 27% of the images context appears before the image.

The *time complexity* of the described method is *linear*, since the web page transformation, the image index estimation and the window of terms computation, as well, are sequentially executed and each of them is linear in time depending on the length of the document.

As we can see in Example 3, the computed context contains some of the words that correspond to the image context but also includes terms that are not related to the image. To better adapt the surrounding frame size, Pan [Pan03] has extended the basic method by leveraging HTML tag hints. The method will be introduced next.

Algorithm 2: Adaptive N-Terms Window

```

Data: Web document  $d$ , image  $I$ ,
Result: Set  $T$  of terms surrounding the image
1  $S \leftarrow getTagsAndTerms(d)$ ;
2  $i \leftarrow indexOf(I, S)$ ;
3  $t = 0$  /* terms counter */;
4  $k = i + 1$  /* loop index for sequence  $S$  */;
5 while  $t < 32$  do
6   if  $S[k]$  instanceOf  $TERM$  then
7      $T.add(S[k])$ ;
8      $t = t + 1$ ;
9   else
10    if  $S[k]$  is  $SEPARATOR$  then
11       $\mid$  terminate loop;
12    end
13  end
14   $k = k + 1$ ;
15 end
16  $t = 0$ ;
17  $k = i - 1$ ;
18 while  $t < 32$  do
19   if  $S[k]$  instanceOf  $TERM$  then
20      $T.add(S[k])$ ;
21      $t = t + 1$ ;
22   else
23     if  $S[k]$  is  $SEPARATOR$  then
24        $\mid$  terminate loop;
25     end
26   end
27    $k = k - 1$ ;
28 end

```

Adaptive Window of Surrounding Terms

The window of terms extractor suffers from the fixed-length of the window and thus cannot deal with varying context sizes among web pages. To overcome these problem, Pan has introduced a more dynamic method [Pan03], that involves the structural information contained in HTML. Pan has recognized that there are certain structural tags that are usually used by web designers to separate contents. The proposed list of useful separators is: $\{\langle br \rangle, \langle hr \rangle, \langle p \rangle, \langle table \rangle, \langle tbody \rangle, \langle td \rangle, \langle th \rangle$ and $\langle tr \rangle\}$. These can be used as clues for context borders and thus a better precision of context is expected. Further, Pan sets $n = 64$ (32 terms before and after the image) and refers to a study conducted by Google Image Search [Goo11] that, however, is not cited in their work.

The extraction algorithm only needs an image and a web document as input since the size of the window is 64 at maximum. The first change compared to the n -term window extractor is the initialization of the sequence S that includes all tags additionally, which are ordered in S by their position in HTML code, too. In line 2, again the index i of the image I in sequence S is estimated. Line 3-4 initialize the terms counter and the index for traversing S . Finally, the first while-loop (line 5-14) iterates over the elements of S that are after the image and checks if they are terms or separator tags. In the case when the current element is a term, the element is added to the context set T . The loop finishes, if either the element is a separator tag or when 32 terms after the image were already collected. The loop is repeated in the same manner for the elements that are before the image in sequence S (line 17-28).

This method has successfully been applied by Feng et al. [FSC04]. Also Zhigang [ZXJQH05] has applied a variant of this method in an image annotation application. However, Zhigang's method only looks for tag separators (`<TABLE>`, `<TR>`, `<TD>`, `<DIV>` and `<HR>`) while the context window is not restricted by any fixed parameter.

The time complexity for the different variants remains linear, but since now the tags are included in sequence S , it depends on both the number of tags and text elements.

3.1.2 Structure-based Wrappers

Images on the Web are often used in product catalogs, tagged albums, news articles and other similar environments. All these applications have in common that the particular image is a part of a data record, which is usually stored in an underlying database and displayed following some fixed template. Programs that are able to extract such structured data records from web pages are called *wrappers* [Liu07]. Since the underlying structures of web pages can vary highly, the manual creation of wrappers is a challenging task.

In the scope of WICE, we introduce two wrapper-based approaches, which follow some manually created extraction rules that are supposed to work for the most web images.

Paragraph Extractor

As the name implies, this wrapper (applied in [FSA96, SOT00]) aims to find the nearest paragraph of an image and considers this paragraph to be the image caption.

This is a DOM-based approach that uses the parent-child relation between DOM elements to determine the context paragraph by estimating the parent tag element of the given image element, which includes text elements in its subtree. All text elements under the estimated parent tag are considered as parts of the image context.

The algorithm proceeds as follows: in the initialization step (line 1-2), the DOM tree

Algorithm 3: Paragraph extraction

Data: Web document d , image I
Result: Set T of text nodes representing the image caption

- 1 $D \leftarrow \text{createDOM}(d)$;
- 2 $i \leftarrow \text{find}(I, D)$;
- 3 **while** $\neg \text{containsTexnodes}(i)$ **do**
- 4 $i \leftarrow i.\text{getParent}$;
- 5 **end**
- 6 $T \leftarrow \text{getTextnodes}(i)$;

D of the input document and the image element I corresponding to the input image are computed. The main part is the while-loop, which starting at the image element, walks the tree upwards, until a parent element is reached that includes text elements in its sub-tree. In the final step (line 6), the text elements under the determined parent are collected as image context.

Example 4. In the following Figure 3.3, we can see the result of the Paragraph Extractor when applied on our example web page snippet. On the right side, the corresponding DOM tree is shown with arrows that point to the container with the image and respectively to the lowest parent node that contains text elements.

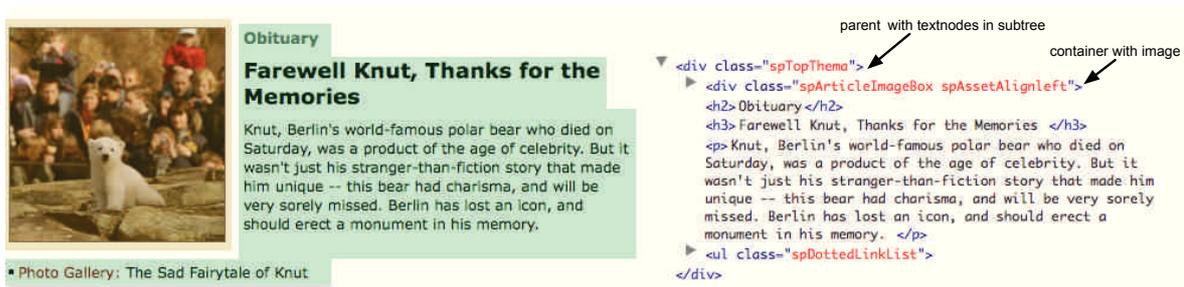


Figure 3.3: The Paragraph Extractor applied on the Example Web Page

As we can observe, for this particular example the Paragraph Extractor performs almost perfectly.

Time complexity of the algorithm depends linearly on the length of the document in regard to building the DOM tree and further the while-loop is at maximum in $O(d \cdot \log(t))$, with d as the maximum degree of the DOM tree and t as its depth.

Monash Extractor

The Monash Extractor [FHB09] can be viewed as an extension to the Paragraph Extractor and has been introduced by Fauzi to handle the different template types in which an image can be embedded in. The basic idea relies on the concept of *list pages*

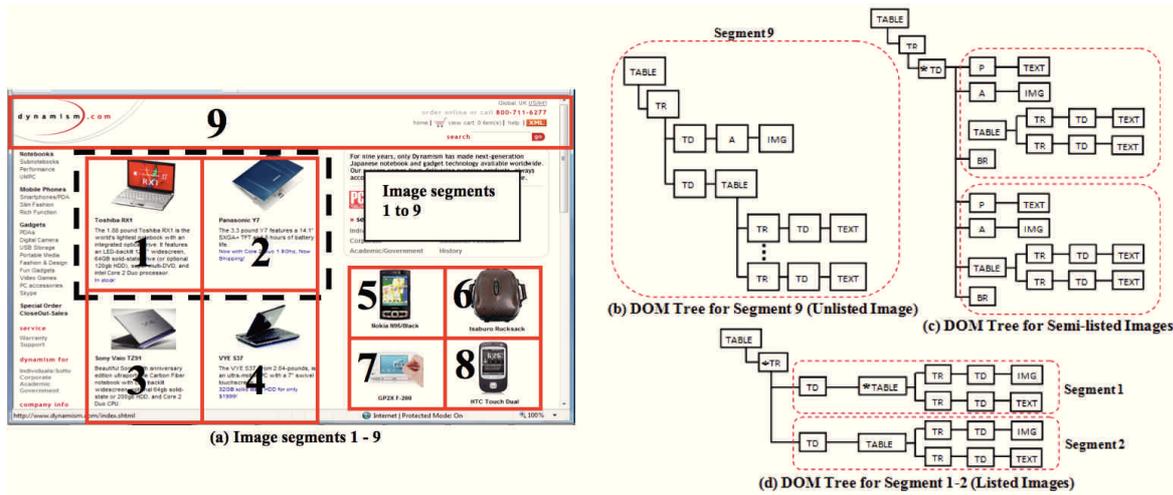


Figure 3.4: Different classes of images in browser and DOM representation [FHB09].

and *detail pages* [Liu07] in which generally extractable data records can be placed in. List pages usually contain a list of many records with similar structure (e.g., a listing of products of a catalog), while detail pages contain detailed information to one particular record.

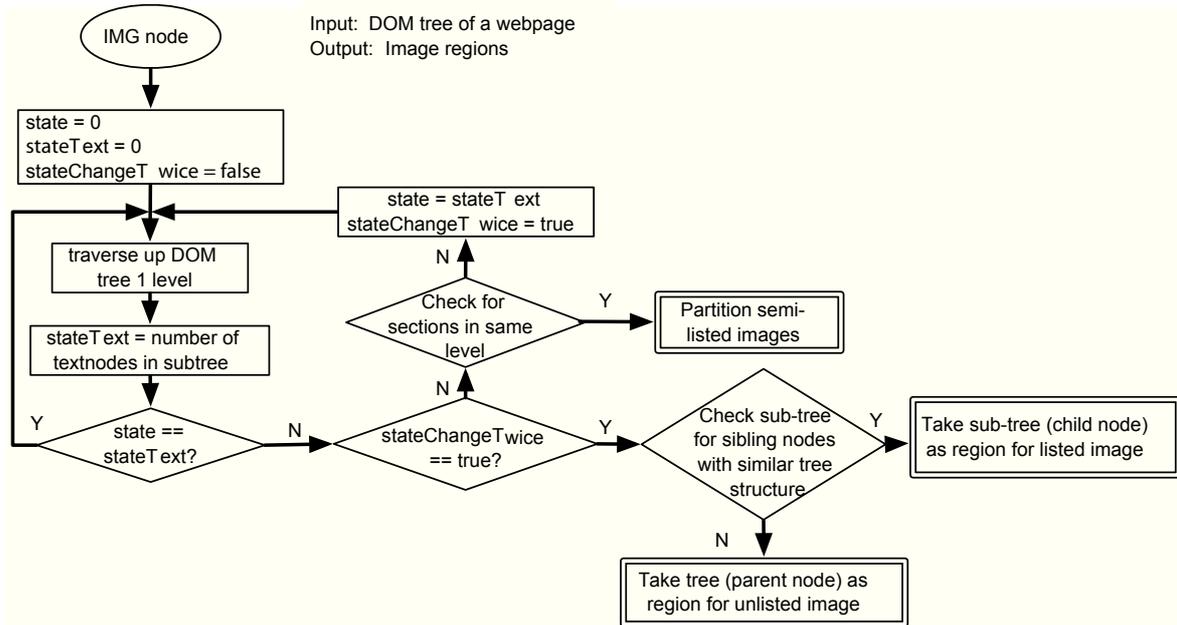
Depending on how listed and unlisted data records can principally be modeled in HTML, Fauzi distinguishes three classes of web images: listed, unlisted and semi-listed images. *Listed images* are two or more images that are ordered within a regular pattern of HTML elements (see Fig. 3.4a, segment 1-8). In the DOM tree each image is placed under one sub-root node (see Fig. 3.4d). *Unlisted images* are standalone images that can be placed on any position in the web page (see Fig. 3.4a, segment 9 and Fig. 3.4b). *Semi-listed images* own the same visual properties as listed images. However, in the DOM tree the segments of particular images are not placed each under one root node, but they are all together under a root node while the visual separations are made by special HTML elements (see Fig. 3.4c).

The algorithm proceeds as follows. The input is a DOM tree and the image node whose context has to be identified. There are three state variables that maintain the current state of the algorithm: `stateText` and `state` keep the current number of text nodes under the actual node and respectively its previous value. Both are set to 0 at beginning. The variable `stateChangeTwice` is true, if the state variable has changed twice during the actual run.

Starting at the image node, the algorithm walks upward the DOM tree, until the number of text nodes under the tree has changed (identified by `stateText` \neq `state`). The algorithm now checks if the number of text contents has changed twice. Since in the first run this is not the case, the current node is checked for typical semi-listed structure with repeating patterns of HTML tags. If such patterns are found, the image

is a semi-listed image and the region of the image is extracted and returned as image context. If no semi-listed structure can be found, the algorithm continues to traverse the tree upwards until a parent node is found at which the number of nodes in the sub-tree changes again. In this case, the number of nodes has changed twice and the sub-tree is checked for sibling nodes with similar tree structure. If such sibling nodes are found, the current image is a listed image and the sub-tree which belongs to the image is returned as image region. Otherwise, the image is an unlisted image, and the complete tree under the actual node is returned as image region.

The algorithm is specified in the following flow-chart adapted from [FHB09].



The Monash Extractor obtains the same extraction result for the sample web page snippet as shown in Example 4. In the mentioned example, the image class is listed image.

The time complexity is the same as for the Paragraph Extractor, since the DOM traversing and text node collecting are also both contained in the Monash Extractor.

3.1.3 Context Extraction By Page Segmentation

As we mentioned earlier, the main content of a web page may consist of multiple topics. Further, beside this main content, a web page can include different functional elements such as navigation and advertisement bars. *Web Page Segmentation* (WPS) describes the task of partitioning a web page into disjoint (non-overlapping) blocks of coherent contents.

WPS has been applied as a preprocessing step in many application areas, i.e.,

Algorithm 4: Context Extraction by Web Page Segmentation

Data: Web document d , image I

Result: Set T of terms representing the image caption

- 1 $P \leftarrow \text{partitionDocument}(d)$;
 - 2 $p_{\text{image}} \leftarrow \text{find}(I, P)$;
 - 3 $T \leftarrow \text{getTextContent}(p_{\text{image}})$;
-

keyword-based web search [LLhPH02, CHL⁺04, YCWM03, FdMRN⁺07], main content extraction [LH02, DMPG05], content de-duplication [CKP08, KN08], web page reformatting for displaying on small screen devices [Bal06, HHMS07, Che03] and web image annotation [HCW⁺07, CHL⁺04, RLL⁺07, LLM⁺06].

In the scope of web image annotation, WPS is a preprocessing step that can be used for WICE. In particular, the algorithm for WICE by web page partitioning proceeds in following 3 steps:

1. The web page is segmented into disjunctive partitions, each including only content that is dedicated to one specific topic (line 1). The result is a set of partitions P .
2. The input image I is sought in the set of partitions P (line 2). The partition containing the image I is denoted by P_{image} .
3. Finally, the complete textual content is extracted as image context from the partition p_{image} (line 3).

A variety of methods to WPS were proposed in the past. We principally distinguish two general approaches that are characterized by their partitioning behavior: *bottom-up* and *top-down*.

Top-Down Page Segmentation

Top-down approaches start with the complete web page as a block and partition this block iteratively into smaller blocks using different features obtained from the content of the web page.

The Vision-based Page Segmentation (VIPS) algorithm [CYWM03] has been applied in several work [HCW⁺07, CHL⁺04, RLL⁺07, LLM⁺06] to extract web image context. VIPS computes for each block a *Degree-of-Coherence* (DoC) utilizing heuristic rules based on DOM as well as visual features. The DoC ranges from 1 to 10, while 10 identifies maximum coherence. Further, the user has to define a *Permitted Degree of Coherence* (PDoc) that stands for the maximum DoC at which partitioning is performed. By using different PDoc values, the granularity of the partitioning can be controlled. As a by-product VIPS computes a *Visual Content Structure Tree* that is

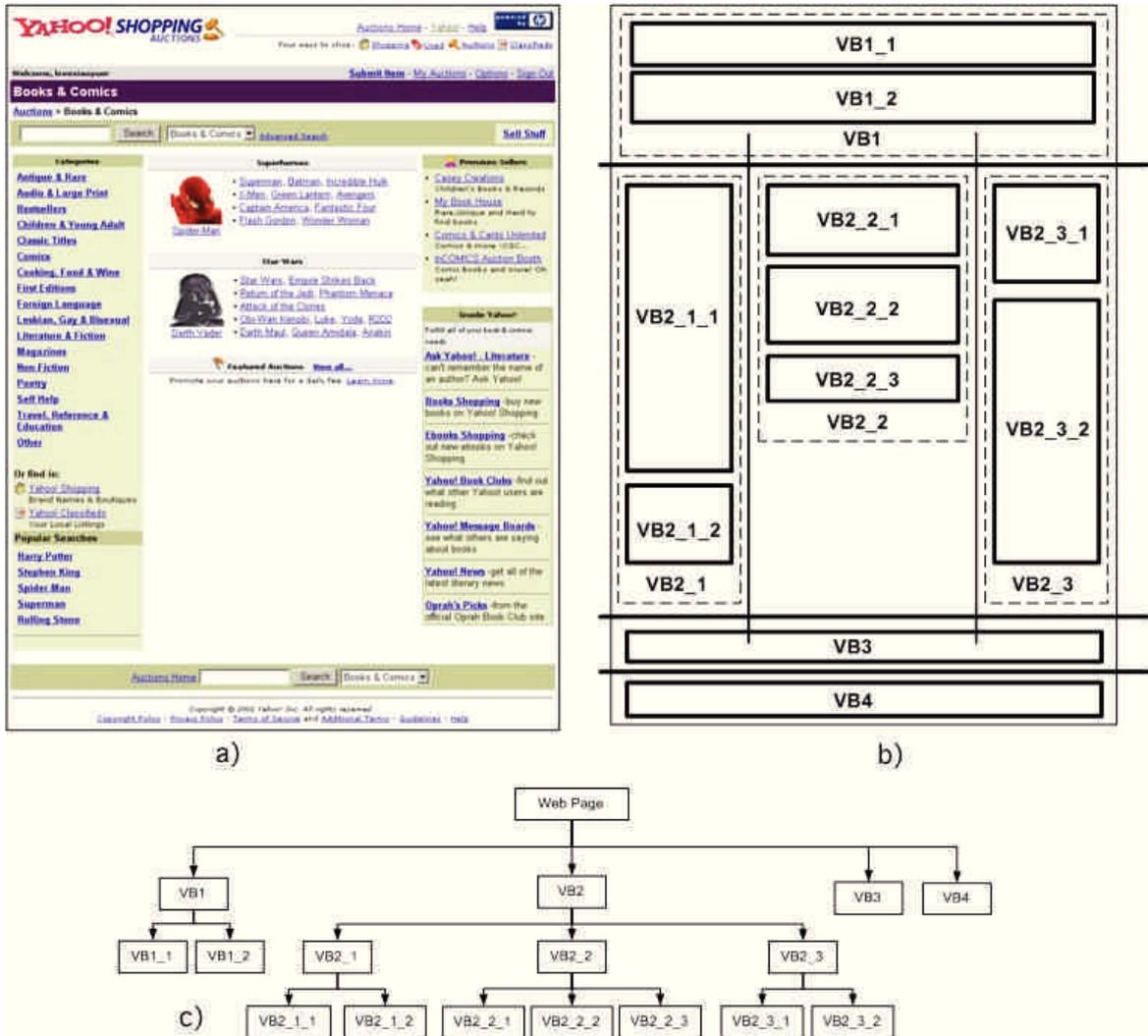


Figure 3.5: Yahoo! Auctions web page a) in browser view, b) as block representation, and c) as tree of visual content structure

assumed to possess a more semantic separation of contents compared to the traditional DOM tree. Figure 3.5 contains an example of a VIPS-segmented web page [CYWM03].

Hattori et al. [HHMS07] combine two different methods for page segmentation. In the first method, a content-distance based on the order of HTML tags is defined and the initial block is iteratively separated at positions where the content-distance exceeds a dynamically estimated threshold. The second method applies heuristic rules that are based on HTML tags.

Lin and Ho [LH02] propose a simple partitioning method based on HTML table structures. Afterwards they compute a *content entropy* to estimate the importance of each block.

Kao et al. [KHC05] define a term entropy, too. Blocks of DOM subtrees are then separated based on the entropies of the contained terms.

Vineel [Vin09] defines a content size and an entropy value that measures the strength of local patterns within the sub-tree of a node. Threshold values are defined for both measures to perform page segmentation.

Chen et al. [Che03] distinguish five high-level blocks, namely, header, footer, left sidebar, right sidebar and main content. Web contents are classified into one of these blocks using heuristics on DOM as well as geometric level.

Baluja's segmentation system [Bal06] divides a web page into nine segments using a decision-tree which employs an *information gain* measure and geometric features.

Bottom-Up Page Segmentation

In the bottom-up approach to web page segmentation the (leaf) nodes of the DOM representation are taken as atomic content units mostly. In the following step, these units are grouped into segments.

Chakrabarti et al. [CKP08] approached the page segmentation problem from a graph-theoretic point of view. DOM nodes are supposed to be nodes in a complete weighted graph and the edge weights estimate the costs needed to put the connected nodes in one block. In this way, the partitioning task could be reduced to a well known optimization problem. A solution to the problem is given by the Energy Minimizing Graph Cuts algorithm. As features, both DOM-based as well as visual features are applied.

Kohlschuetter and Nejdil [KN08] represent atomic content units of a web page by a quantitative linguistic measure of *text-density* and reduce the segmentation problem to solving a 1D-partitioning task. An iterative block fusion algorithm applying methods adapted from computer vision is presented as a solution.

Li et al. [LLhPH02] propose improving web search quality by segmenting web pages into cohesive *micro-units*. The segmentation procedure involves creating a *tag tree* which is similar to DOM and then applying two heuristics to aggregate tree nodes into segments. In particular the heuristics are: merge headings with the following content, and merge adjacent text paragraphs.

3.1.4 Summary

In this Section, we introduced related work to WICE methods from the literature. We have further categorized the approaches in window-based, wrapper-based and WPS-based extraction algorithms. The window-based method is simple and fast, however, as we could see in the examples, the quality of the extracted context is low, since the heuristics are too simple in order to cover the variety of web page designs. Similarly, the DOM wrappers suffer also from the simple extraction rules, which make them only applicable on web documents with well-structured DOM trees. The WPS approaches

are more promising, because they seem to be more adaptive to different web page designs. However, they can not be used without complex modification that can handle the different granularities of web page block containing the image context.

3.2 Evaluation of WICE Methods

In order to decide which extraction method to apply in a certain situation, it is inevitable to evaluate and compare the performance of the different methods. WICE has usually been applied only as a preprocessing step in other applications and therefore the extraction methods hardly have been evaluated on their own.

In this section, we will focus on the few evaluation scenarios that deal with image context extraction to a certain degree. We start in Section 3.2.1 with an approach that directly has evaluated the image context extraction. Then in Section 3.2.2, we briefly introduce different evaluation studies that only estimate the impact of WICE on different applications.

3.2.1 Direct Evaluation of WICE

In [FHB09], Fauzi et al. have proposed the Monash Extractor that we have already introduced in the previous section. In order to evaluate their WICE method, the following testing environment has been created:

Data collection. As data set, 100 web documents were randomly selected across various categories in Alexa Web Directory [Ale11] and manually labeled by 30 volunteers. Each volunteer processed 10 web pages, which means that every page was labeled 3 times. This resulted in 3 labeled sets of data, and as the final set, they took the broadest context from the 3 available. Banners and layout graphics were filtered by checking image dimensions: only images with a width and height greater than 45 pixels and a width-height ratio between $\frac{1}{2}$ and 2 were processed. The outcome is a set of 1,019 image-and-context pairs.

Evaluation Measures. The evaluation of the proposed extraction algorithm is done within a system-based framework where the *Precision* and *Recall* measures are applied. In this context, Precision is the percentage of correctly (test on exact matching) extracted image descriptions over the total extracted image descriptions and recall is the percentage of correctly extracted image description over the total actual number of image descriptions in the dataset. For both measures, the average value over all extracted image-context pairs is computed. Furthermore, the *average processing time* needed to extract all images and corresponding context per web page has been estimated.

Evaluated Methods. Within this study, two methods were evaluated: the Monash Extractor and an extractor based on VIPS. The PDoC value, which sets the granularity

of the segments in VIPS has been varied from 5 to 7, similar as in [CHL⁺04], so that three different extractions were computed using VIPS. This resulted in four precision and respectively recall values at overall. However, the average processing time has only been estimated for the Monash Extractor.

3.2.2 Implicit Evaluation of WICE

Souza-Coelho et al. [CCS⁺04] list four sources of web context within a web document: description text (image filename or alternative text), meta tags (located in the HTML head of the document), full text, and surrounding text passages (see window of surrounding terms in Section 3.1.1). Within an evaluation task they analyze the image retrieval performance based on the different context sources applied for image indexing. They have found that the surrounding text passage consisting of 20 terms before and after the image (passages of 10, 20 and 40 terms were separately inspected) performed best for single evidence ranking. However, other methods to passage extraction are not investigated.

Another system, where the surrounding text passage plays a great role is Image Rover [STC97]. The image indexing in this system is based on textual and visual cues of images, while the textual cues are obtained from plain text of web documents. Different document parts are weighted depending on their parent tag properties, where the surrounding text weight is among the highest. The system performance has been evaluated by applying the *target test paradigm* [VL00], which tests how efficiently a system performs in finding a target image in the data collection. As in [CCS⁺04] the presented evaluation shows only the impact of one context extraction method to the image retrieval task. However, there is no comparison, since no other extraction method has been tested.

Tian et al. [YhTjW05] extract visual, relational, and textual image features for web image classification. The textual information are gathered from the sibling nodes of web images in the DOM representation. Based on the associated features the images are classified and the *classification performance* is analyzed. As an alternative textual context extraction method VIPS [CYWM03] is mentioned but not applied in the scenario due to its high complexity.

The effectiveness of VIPS was only judged indirectly. In [CYWM03], Cai et al. have tested VIPS's segmentation quality employing 5 human users who classified the segmentation results to perfect, satisfactory, fair and bad. He et al. [HCW⁺07] applied VIPS for context extraction and have evaluated web image retrieval and clustering efficiency, without focusing on the context extraction, nor comparing with other extraction methods.

3.2.3 Summary

In this section, we have presented an overview of different evaluation works performed in the context of WICE. A general distinction can be made between direct and indirect evaluation. There is only one direct evaluation approach that could be found in the literature. The method also includes a small comparison of the Monash Extractor to the VIPS-based method. The correspondence of extracted and ground truth image context is tested on exact matching, which could be a too strong criterion. More often, the WICE evaluation has been conducted indirectly in the context of the main application. However, since the quality in these applications depends on many other factors, we can not decide the significance of the applied WICE method.

3.3 Applications of Web Image Context

WICE has valuable benefits for different applications and has therefore been often applied as a preprocessing step. This section gives an overview of different approaches across the literature that involve web image context. In particular, we will focus on the main application, namely web image retrieval and annotation.

3.3.1 Web Image Retrieval

Web Image Retrieval is the most intuitive application for WICE, since the image indexing task can be reduced to the well known text indexing. A variety of approaches has been introduced by different research groups. Kherfi et al. [KZB04] presented a broad survey of many web image retrieval systems. The most significant of them are briefly introduced in the following.

One of the early systems is WebSeek [SC97], which gains the indexing keywords only from the URL of the images. The image dataset used in that work was stored in taxonomic ordered directories. The image filename and the directory names are used as description terms.

ImageRover [STC97] uses the full text of the web page to index embedded images. Terms are weighted using the $tf*idf$ scheme from information retrieval. Additionally, the terms appearing within specific HTML tags are given a special importance by a weight factor. For example, the alternative text of an image gets a much higher weight compared to the title of the web page.

In [CCS⁺04] four sources for image context in web documents are defined, namely description tags, meta tags, full text and text passages are combined using a belief network model. As a *passage*, the 20 terms before and after the images are chosen.

WebMARS [OBMCP00] divides the textual data into two granularities, at the re-

gion level and the complete document. Regions are specialized document parts like titles, citations, paragraphs. The two information sources are used to index images with global and local descriptors.

Diogenes [AY00] is a person photograph search engine which mainly tries to extract people names from the text surrounding the images and associate it with the faces on images detected by a face recognition module.

The VAST (*VisuAl & SemanTic* image search) system [JHTS08] combines visual (region color and texture) and textual (filename, ALT text, surrounding text, etc.) features in a *Semantic Network* to retrieve web images. In particular, first the images are segmented and visual features are computed from the segments. The extracted features are then clustered by k-means clustering [Mac67] and the images serve as a connection layer between keywords and clusters. In the resulting semantic network, links from keywords to visual feature clusters can be derived from the images (Figure 3.6). Link weights from a term to a cluster are computed by summing up the different path weights of a term to a cluster in the original network. Image search is started by a keyword for which the cluster with the highest link weight is determined and the corresponding images are presented.

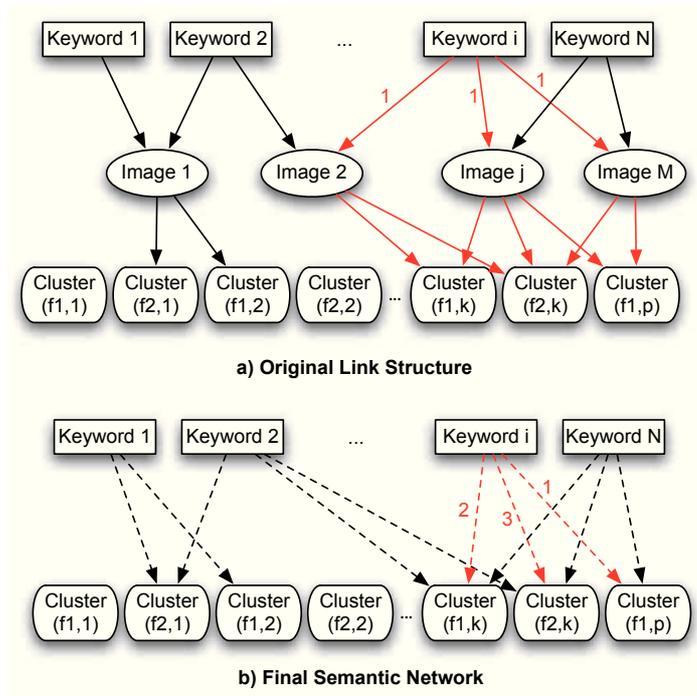


Figure 3.6: Construction of Semantic Network [JHTS08]

3.3.2 Web Image Annotation

As we already mentioned in the introduction, web image annotation terms the task of, given a target image, finding appropriate keywords that describe the semantics of the image best. In this section, we will focus on image annotation approaches that leverage the web image context.

Wang et al. [WZJM06] propose AnnoSearch, an image annotation system that needs an image and at least one term associated with the that image (they propose to extract it, e.g., from the folder name in case of a private image, or from the web context in case of a web image) as input. Given the input term, a keyword-based search is performed on a large-scale web image database (e.g., Google Image Search [Goo11]) and the results are ranked based on visual similarity to the input image. The top N ranked images with their textual descriptors (title, URL, surrounding text) are obtained and the descriptions are clustered. Finally, the key concepts of the top ranked clusters are taken as annotation keywords. Other search-based methods for image annotation have been proposed in [WJZZ06, RYWL07].

Rui et al. [RLL⁺07] introduced a *Bipartite Graph Reinforcement Model* for image annotation. In particular, they determine initial annotation candidates from the web image context. Since the image context contains noisy and incomplete keywords (VIPS has been applied to extract image context), the initial annotation is extended and only a selected subset of all candidate annotations is chosen for final annotation. In particular, the method includes the following steps: first initial annotations are extracted from image context (terms are stemmed, filtered from stopwords, and weighted by $tf * ift$; terms with highest weights are returned). Then, further candidate annotations are derived by submitting each candidate as a keyword and then clustering the search results similar to [WZJM06]. The two kinds of candidates are then modeled as a bipartite graph, on which a reinforcement algorithm is performed to iteratively refine the ranking scores. When the algorithm converges, all annotation terms are re-ranked and only those with the highest ranking scores are selected as the final annotation.

He et al. [HYR08] state the same problems of web image context and present an *Multi-Progressive Model* for annotation of web images. The basic idea of their approach is to leverage word-correlations between image context words and an automatically built vocabulary. In the first step of their method, the initial keywords extracted from the image context are extended by candidate annotation words from the vocabulary. Therefore, for each word of the vocabulary a similarity to all other words from the vocabulary is computed based on the co-occurrence frequency of the words in the same image and the top 20 words according to similarity are chosen as extending word. Then for each word from the initial word list, they compute a visual similarity to the target image: in particular, an image search engine is queried with the word and an average

visual similarity of the Top-N returned images to the target image is computed. The candidate annotation words are ranked using this visual similarity and a joint word probability and the highest ranked words are taken as final annotations.

Jin et al. [JKWA05] apply another strategy to identify irrelevant keywords in the initial web context based annotation by relying on WordNet [MBF⁺90]. In particular, they investigate various semantic similarity measures between keywords and finally combine the outcomes of all these measures together to make a final decision using Dempster-Shafer evidence combination [Sha76].

4

EXTRACTION OF WEB IMAGE CONTEXT

In the motivation of this thesis, we have already introduced the idea of exploiting textual contents that come together with web images on hosting web pages to gain useful image descriptions. We stated that the most valuable image context source – which is the free text associated with the image – is however not explicitly given and needs to be extracted. Facing this aim of determining the web context for a given image, it is inevitable to know what image context looks like and further how both image and corresponding context are placed together on a web page. Both of these issues are addressed in Section 4.1, where the different functional parts of a web page wherein images with context can appear are introduced and discussed in terms of the properties of image context.

After that, Section 4.2 describes the WICE process first in words and then by a formal definition. This definition is interpreted by two more concrete representations, based on the characters in the source code of the web documents and as nodes of the corresponding DOM tree.

Finally, in Section 4.3, WICE is presented as an IR task and many well-known concepts from the field of IR are mapped to the WICE task, and the chapter concludes with a summary.

4.1 The Attributes of Web Image Context

As *web image context*, we understand these textual parts of a web page that belong to a web image and therefore share similar semantics with the image. Although this

informal definition is intuitive and simple, formalizing the concept of web image context is very difficult due to the high variety of shapes that image context can have.

A web page can principally be composed of different parts with different tasks and properties, as for example a header, a navigation bar, the main content, different kinds of advertisements, link lists, and other functional elements. To describe the various attributes of web image context, it is essential to distinguish between the different parts of a web page in which web images can be placed in.

Page header The *header* of a web page usually contains some general information to the web site and other functional elements as a search bar or login information. Figure 4.1 shows an example web page header taken from the CNN web site [CNN11] which is a typical online news portal.



Figure 4.1: The header of the CNN web site

In this example, the only present image is the web site logo and there is no real image context. As a consequence, the textual contents do not share any semantics with the image and would produce false annotations if considered as image context. This example is a prototype for most page headers: logo images with almost no context information which therefore can not be textually described.

Advertisements on web pages are a very important way of marketing. Usually, a web page has a determined space for ads that is sold to marketing companies who control the shown commercial depending on the page content. Images are popularly used in advertisements since they have the ability to show the complete information of the advertised product at once and thus are more effective in catching the users attention. Figure 4.2 shows two different kinds of web advertisement. The commercial on the left contains an image which partly shows



Figure 4.2: Advertisement images and corresponding web context.

the Nokia E7 smart phone and three texts. Although the text could be useful

to describe the image, it is placed into the image and thus can not be extracted as image context. The only text we can extract is the caption „Advertisement“ from the bottom, however it is very general and does not describe the image content. The commercial on the right campaigns for the Yahoo! market place where users can buy smart phones and MP3 players. The corresponding image contains an Apple iPhone which is one of the products that can be found on the marketplace. This time the image context can be extracted from the web page, but the extracted text does not precisely describe the semantics of the image. These two examples show the typical problems with images in ads: either there is nothing to extract or the associated context is semantically only far related to the image.

Link lists are commonly used on web pages to refer to other (related) stories that the reader may be interested in. Usually, the link text is a short and clear statement, which describes the referred story (e.g., the headline of the article). There are also link lists, where the link text is assisted by a small web image describing the referred story. An example of a link list with images is depicted in Figure 4.3. In this example each image has one corresponding context. Although the

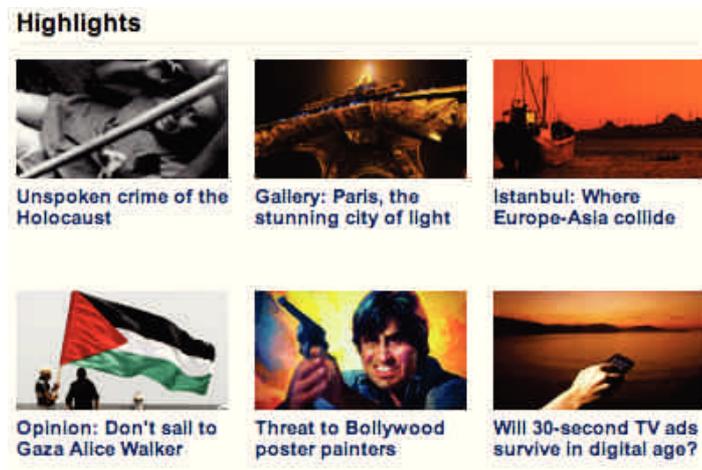


Figure 4.3: Links with web images to other CNN stories.

text does not exactly describe the image semantics, it shares the same semantic background with the image, which is the linked story and thus is worthy as image description.

Main content is the part of a web page which contains the primary information a user is looking for. It can be a story in a news page or a blog, the detailed view of a product on a shopping portal, or the profile view in a social web domain, etc. As a result, there is a high variety in main content appearance and therefore

a) **Selling blood for survival on the rise** Send to a friend
 Saturday, 25 June 2011 11:47



By Edward Qorro and Yvonne Mvanda
 The Citizen Reporters
 Dar es Salaam. The high cost of living, dwindling employment opportunities and economic hardships in general have pushed up survival tactics, particularly among struggling urban dwellers across the country. Everywhere you go or look, people are pouncing on any slightest opportunity to make quick bucks. And more often than not, very few are deterred by the fact that some of these activities could be illegal.

What matters most was survival! And if there was a need, the better for the prospective scavengers. One such fertile ground is apparently within our public health institutions. Over the last one week, The Citizen on Saturday has established that individuals have taken advantage of the huge demand for blood at the health facilities to offer theirs for a price.

Today, for a cost of between Sh40,000 and Sh50,000, you could easily find someone lucking in the precincts of the hospital to offer a pint of blood for a needy relative or friend.

Our investigations found out that a syndicate of selling blood to 'prospective customers' is blossoming among Dar es Salaam's major hospitals. It is not a surprise anymore that such a syndicate is common sight at Muhimbili National Hospital (MNH) as well as at Amana, Temeke and Mwananyamala district hospitals.

[...] (shortened)

b) **Power of Asia**



Megacities rising in the East
 Asia will have seven of the ten largest cities on Earth by 2025. How will this change the continent - and the rest of the world?

c) **The setting sun through the trees**
 by blmiers2 (ausblenden)
 © Alle Rechte vorbehalten
 1.555 Aufrufe
 34 Kommentare
 4 Notizen
 17 Favoriten
 Aufgenommen in Webster, New York (Karte)
 Mit Tag autum, trees, red, usa ...
 Aufgenommen am 17. Okt 2010



Figure 4.4: Image and corresponding context in the main part of web page: a) Long story article (TheCitizens), b) short story article (BBC) and c) detail view of image (FlickrR)

also in image context structure, which reaches from small image captions, to short text articles, and to longer stories. Additionally, the main content might be uncontinuous, i.e., interrupted by advertisements, link lists or other unrelated information. Some typical examples for images that are part of the main content of a web page are depicted in Figures 4.4a)-c). The first example (Figure 4.4a)) is a long story article with an image which contains a short caption. While the complete article only partly shares semantics with the image and thus is imprecise, the image caption is very qualitative and describes exactly what the image depicts. Given that such image caption exists in a long story article, it is a good advice to extract only the image caption as image context. However, image captions are not always available. Then the complete story article or (in very long articles) some parts like the headline and the text next to the image can be used as image context. The second example (Figure 4.4b)) shows an image

in a short article, which typically occurs on the first page of a news page that mostly gives an overview of the top stories of the day. Here, the article is usually composed of one or more headlines and ends with a short story text. In practice, the complete short article has proven to be a good image context. Finally, the detail page of an image in Flickr is depicted in Figure 4.4c). In this example, it is really difficult to estimate the best image context because there are good image tags, which are mixed with image meta tags not related to the semantics of the picture. However, in this case we have to decide whether we just pick the two text objects surrounding the image as image context or the complete text. In the first case, we obtain a high precision with a lower recall, and in the second case we get the opposite.

4.2 A Formalization of the WICE Task

WICE deals with finding the image context of a given web image in the embedding web document. To be more precisely, WICE denotes the process of, given a web image, estimating that textual parts from the hosting web page that belong to the web image – we refer to these textual parts as the *web image context*. The three concepts that play a main role during this task are:

- the *web image*, for which the context has to be determined,
- the *hosting web page*, where the image is contained in, and of course
- the *image context*, which has to be estimated.

Definition 8 (Web Image Context Extraction). *Let I be a web image and D be the web document that contains I . The WICE task can be formulated as a two dimensional function $f : I \times D \rightarrow C$, where C is the image context.*

From this general formalization of the WICE task, we can derive that the web image and the web document serve both as input, while the returned image context represents the output. Now, depending on the concrete representation used to identify the concepts, we can obtain different views on the function f .

Source Code Representation In this representation, we assume that the given document D is represented by a text string which contains the complete HTML code of the document. The image I in this representation is represented by the corresponding image URL. Furthermore, the image context C can be expressed as following:

$$C = \{(a_i, b_i) | a_i \text{ start, } b_i \text{ end markers}\},$$

where the (a_i, b_i) are pairs of *start* and *end markers* with $1 \leq a_i \leq b_i \leq |D|$ ($|D|$ representing the source code length of the document) which identify the image context parts in the HTML source of the document.

DOM-based Representation In the DOM-based representation, the web document D is represented by a DOM tree, and the image I corresponds to an ImageNode element which is contained in the DOM tree D . Now, the image context C can be formulated in the following way:

$$C = \{t_i | t_i \text{ is the content of a TextNode element in } D\},$$

where the image context is assumed to be a finite set of text node contents.

Both representations are suitable to describe the WICE task. The only difference that makes the first representation more general is that in the source code representation the image context can be composed of any character sequences that are part of the HTML source, while in the DOM representation the context must be composed of complete texts that are in a TextNode element. In other words, the smallest unit of an image context part in the DOM representation is on TextNode level, while in the source code representation the smallest image context unit is on character level.

Although the DOM representation is more restrictive, it turned out to be more suitable, since in our investigations the elements of the image context were never a part of the TextNode content, but the complete TextNode contents. Therefore, for the rest of this thesis, we will use the DOM-based representation. Further, the term *text content* will be used to refer shortly to the content of a TextNode element.

4.3 WICE as an Information Retrieval Task

To some extent, WICE can be interpreted as a special IR task. The three main concepts of WICE from Definition 8 can be straightly mapped to concepts of the IR task:

- the input image I corresponds to the query concept in IR,
- the set of text contents T_D of the web page D includes all potential candidates for the image context and hence can be interpreted as the document collection in IR, and finally
- the sought image context which is a subset of T_D can be understood as the answer set in IR.

With this interpretation, we can apply the methods and concepts of IR directly to WICE and try to transfer similar solutions for the problem.

One of the main issues in IR is the common representation of query and retrievable documents. In case of WICE, we mean the representation of input image and the text contents of a web page. In this thesis, four representations are distinguished:

1. HTML-based: both, image and text contents are included in the HTML code of the document and have an absolute position therein, which orders them sequentially.
2. DOM-based: images and text contents identify element nodes in the DOM tree. There are special characteristics that can be derived from the position of the elements in the DOM tree.
3. Visual-based: web documents are presented to users in a browser, where images and text contents are rendered and placed in a two dimensional plane. The positions can be extracted from the rendering application and used to represent the objects.
4. Semantic-based: textual contents can be semantically represented using methods that were introduced in Section 2.3. However, trying to automatically estimate the semantics of an image immediately confronts with the Semantic Gap, which we described as a hard problem in the introduction of this thesis. Anyway, the images have some other hints, that can deliver a semantic representation for the present: these are the alternative text (if present) or the image URL. Since both are textual, again common text-based methods can be applied to obtain an useful representation.

The next important issue in IR is the document relevance (see Section 2.2.2), which is the question for a suitable similarity (or distance) measure between query and documents in the document repository. In the transferred sense on the WICE task, the document relevance can be interpreted as the rate to which extent a text content belongs to an image.

There are different similarity measures to compute the proximity of image and text contents, which can be formulated based on the different representations of the concepts that were previously presented. The concrete measures used in this thesis will be introduced in the later chapters, when they are applied.

Having defined a similarity function, an IR system usually computes the similarity between the query and all documents in the collection, and finally returns all documents with a similarity greater than a predefined threshold value; or else, another method is to deliver the top N documents with the highest similarity as the answer set. Both methods can be used in the WICE task, too. However, since the web context consists usually of a determined set of text contents and one text content more or less, can

significantly affect the quality, it is very difficult to find an appropriate threshold for the similarity, or respectively an appropriate N , that is suitable to all kinds of image context. For example, the length of a context with respect to the number of text content can vary highly as we have discussed in Section 4.1.

A very useful method that we introduced in Section 2.2.3 is cluster analysis, which groups documents of similar content to same clusters. In the IR process with a pre-computed clustering the query has to be compared only to the cluster representatives and only the documents of the cluster with the highest similarity to the query are returned as answer. If we turn our view again to the WICE task, we can recognize the benefits of this method: using an appropriate clustering method, we group the text contents to different image context candidates, and finally that context candidate with the highest similarity to the image is chosen as the image context. This method is more scalable to the WICE problem, since it does not depend on any heuristically predefined thresholds.

The field of IR exists already for many years and there is a variety of methods, which were investigated in the past to evaluate and compare the performance of IR methods. We have introduced some basic concepts and methods of IR evaluation in Section 2.2.4. The evaluation methods can be transferred to the WICE task without any modification. The particular evaluation measures and the way how they are applied for WICE evaluation will be the scope of the next chapter.

4.4 Summary

In this chapter, we have introduced the main problem of this thesis, which is the extraction of the web image context. To understand the difficulty of the problem, it was necessary to introduce the different attributes of web image context by presenting and discussing the different parts of a web page in which the image context usually appears. After this, we have given a formalization of the WICE problem where the three basic concepts were defined and possible representations were discussed. Finally, we turned the view on WICE from another perspective by interpreting the problem as a special IR task. Here, the different concepts of IR that were introduced in Chapter 2 with respect to WICE were presented.

5

EVALUATION FRAMEWORK FOR WICE

The main objective of this thesis is to develop new WICE algorithms that will precisely extract web images and their textual context from a hosting web page. In order to be able to decide which extraction method performs best under various circumstances, we need to find a way to objectively measure, evaluate and compare the performance of the different extraction approaches. As we already mentioned, WICE is a preprocessing step in many applications and therefore has hardly been evaluated on its own. We want to bridge this gap by offering a new evaluation framework that is adapted to fit the special requirements of WICE methods.

Some of the preliminary results of this chapter have been published as an article [AC10] at the Multimedia Data Mining workshop (MDMKDD'10), which was a part of the ACM KDD conference.

In the first section, we will discuss the different existing evaluation methods applied in the literature and try to point out why a new evaluation framework is essential. Section 5.2 introduces the evaluation framework and its major components in detail. Finally, in Section 5.3 the proposed framework is tested with existing WICE extraction methods from the literature and the results are presented and discussed.

5.1 Why a new Evaluation Framework?

In the related work chapter, we already gave a brief overview of existing evaluation tasks presented in the literature that deal with WICE. The approaches were roughly categorized into *direct* and *implicit* evaluation methods referring to the way in which the WICE output has been investigated during evaluation; “direct”, as the name implies, means that the subject of evaluation was the output of WICE itself, while “implicit”

means that the results of another application that uses WICE in a preprocessing step (such as image search) were in the main scope of the evaluation task. However, each of the evaluation approaches has some drawbacks that can affect the evaluation quality and expressiveness.

The major drawback of the implicit evaluation of WICE is that it depends highly on the main application and its properties. For example, the image retrieval application that uses WICE to obtain image descriptions relies on the quality of the context coming with the images. If the context is of high quality, then the retrieval usually performs well although the context extraction might be very imprecise. Otherwise, a perfect extracted image context from a website where the context is of low quality leads to low retrieval performance. Similar observations can be made in other applications like image clustering or classification that use WICE to obtain image descriptors. Obviously, the example above shows that there are cases where indirect evaluation of WICE is insufficient and not reliable.

On the other hand, the direct evaluation of WICE investigates the context itself and thus seems more promising. First, we want to discuss the manual direct evaluation, where a user is shown a website and the outputs of a WICE method and values the extraction quality by classifying the image-context pairs to “good” or “bad” (further intermediate levels are possible). This method suffers from the subjectivity of the user, since it is very difficult to classify the quality of context extraction to discrete values. An extraction that is only “satisfactory” for one user could be “excellent” for another. An additional drawback of manual evaluation is the high effort which is usually tedious and time consuming and therefore not applicable for larger evaluations which are necessary in order to obtain more representative results.

Another direct evaluation method is to manually determine the image context for a predefined set of websites (*gold standard*) and then to compare the output of an extraction method to this gold standard as applied by Fauzi et. al [FHB09]. A very important issue is how the comparison of gold standard and extract is performed: Fauzi has chosen to compute precision and recall on a web site level, where precision refers to ratio of the number of correctly extracted (based on exact match) image-context pairs of a web page over all extracted pairs, and recall is the percentage of correctly extracted pairs over the number of image-context pairs in the gold standard. However, this method turns out to be very inflexible, since extracted data and gold standard are tested on exact matching and every small divergence is penalized with an complete miss. Instead, another comparison method should be applied that is able to handle partial correspondence between extracted context and gold standard. Furthermore, since the gold standard has to be created manually it again suffers from subjectivity of the user and the high effort needed to prepare representative data sets. However, the determination of the image context can be done more objectively if some common

rules are defined at the beginning, e.g., if image caption is available then take only the caption as image context. On the other hand, we could think about how the creation of the gold standard could be partially automatized in order to reduce the manual effort.

In the evaluation task performed to measure the quality of the Monash extractor [FHB09], the authors have included also the VIPS-based extractor for comparison. Both methods seem to have different input and output formats, that need to be standardized in an evaluation framework.

The discussions about the drawbacks of the existing approaches should now enable us to build an own evaluation framework that avoids the major mistakes and thus guarantees the best performance analysis. The resulting framework will be presented in the following section.

5.2 Evaluation Framework Design

In Figure 5.1, we have depicted a flow diagram showing the basic parts a direct evaluation environment should be composed of.

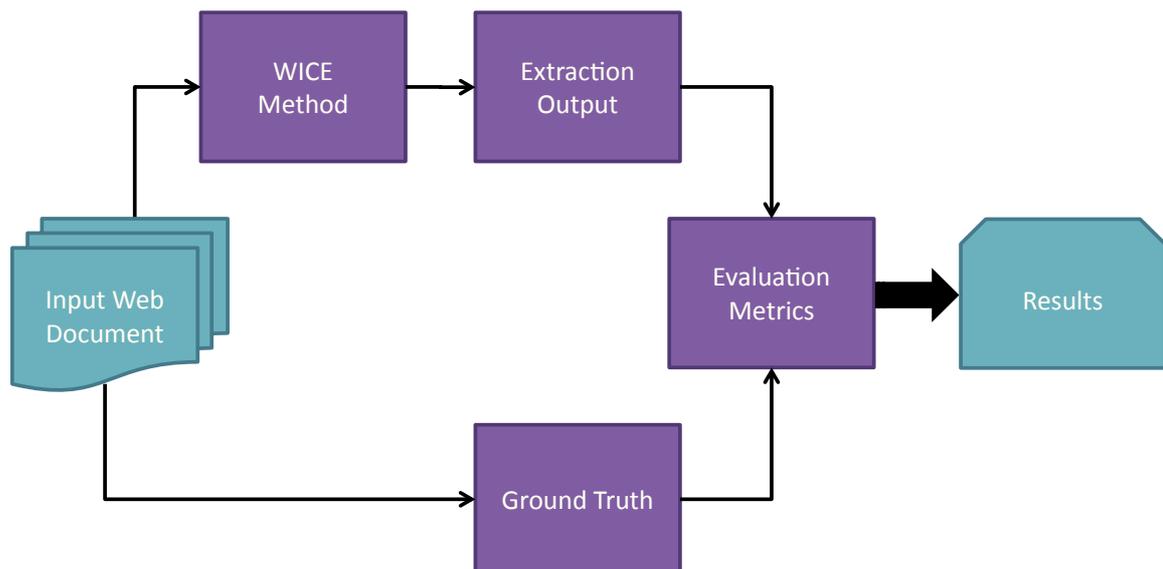


Figure 5.1: Overview of the different Modules of the Evaluation Framework

The evaluation process starts with web documents as input which are real documents collected from real web servers. For these documents, the images and the corresponding contexts are determined in a controlled way assisted by a human expert resulting in the ground truth data (or the gold standard) of the evaluation. On the other side, there are different WICE methods that analyze the input web documents and automatically compute images and image-context pairs that are assumed to be-

long together. Both outputs, the gold standard as well as the extraction results are of the same format. In the next step, these outputs are compared to each other by applying suitable evaluation metrics. The result of the evaluation framework is the output gained during that final comparison step which can be presented to the user as different graphs.

In the following, the particular modules of the framework will be presented in more detail.

5.2.1 Web Documents

The evaluation framework estimates the performance of WICE methods by applying them on real web documents, that is, the documents serve as input data for the WICE methods and are therefore an essential part of the framework. As we already mentioned in the introducing chapters, web documents are mostly written in HTML by different authors around the world with different programming experience and designing skills. This variety is further encouraged by the loosely restricted standard of HTML which allows the author to produce a needed output in different ways, which can be misleading for the later analysis, e.g., a bold tag with an increased font size can produce the same output as an header tag. Even more challenging are deficient HTML documents that are affected with missing (unclosed) tags but are accepted and correctly presented in browsers, which apply different techniques to repair the ill-formed documents. In this framework, the input documents are therefore passed through the JTidy parser, which is able to generate valid HTML code as the parsers in most browsers do. This preprocessing method is imperative at this step since without a well-defined HTML structure it is not possible to build the DOM tree which is the core presentation for further analysis.

Another difficulty for the extraction process becomes apparent when we look at the desired contents in an input document. As described in the introduction, web documents are cluttered with different kinds of content which belong to functional, structural and the real information content. This implies that there are also images that belong to these different parts. It is therefore necessary to find out which images belong to the real content of the web page and which are only structural or navigational, because it does not make sense to search for the image context of non-content images. In our implementations, we have therefore applied a rule-based image filter, which detects the unwanted images and excludes them from being analyzed. Structural and functional images serve as background or click-areas in different parts of a web page. We empirically found out that these images mostly are characterized by certain image dimensional properties which allow us to define filtering rules based on these values. We have applied the following filtering rules which are very similar to those suggested

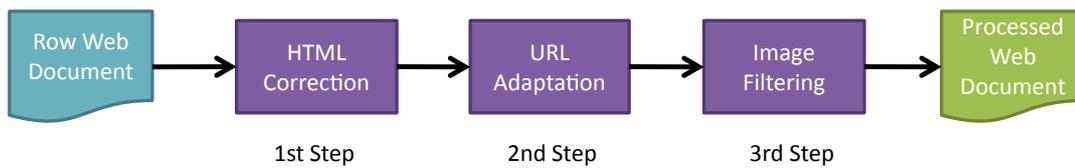


Figure 5.2: Web Document Preprocessing

in [FHB09, FSC04, CCS+04]:

1. Filter images with a width or height smaller than 60px.
2. Filter images with a width-height ratio greater than 2.5 or smaller than 0.4.
3. Filter images of Graphic Interchange Format (GIF).

The first rule excludes images too small to be processed since these images are assumed to be either decoration images or button areas. The second rule filters images whose width-height ratio exceeds a predefined range. The affected images are mostly background graphics, website logos, or banners which do not belong to the main content of the web page. Finally, the third rule filters all GIF images which mostly represent advertisement images and rotating banners.

Web documents are hypermedia documents which contain a lot of links to other documents and also include many other media types by referring to the URLs of these data objects. For example, images are included in a web document using the source attribute of the image tag that gets an URL of the image as input. For our evaluation framework it was necessary to collect a set of web documents and to store this set locally in order to ensure a reliable and fast access on the documents. However, URLs in web documents are mostly encoded relatively in respect to the actual location of the document on the web server. If we download the web document source to store it locally, the relative links and paths become invalid and have to be adapted. It is possible either to copy all the needed resources to the disc in the same directory schema as they are presented on the web server, or – the way we have chosen for this evaluation framework – to convert all relative links and URLs to absolute ones and thus to provide the access on the desired resources (scripts, images) without the need to store them locally.

To sum up, an input web document is preprocessed in three steps in our evaluation framework as depicted in Figure 5.2. First, if the document is validated and possible validation errors are corrected using the JTidy Java library. In the second step all URLs are converted to their absolute form. And finally, some images that are not of the main part of the web document are filtered. “Filtered” does not mean that they

are removed from the original web document but just that they are not passed to the next analysis steps.

5.2.2 Gold Standard

The gold standard (or ground truth dataset) plays a very important role during the evaluation process, since it defines the desired extraction result that a WICE method should output. However, the creation of such a “perfect” ground truth dataset that is further representative for all the documents on the Web, comes with several problems that will be described next.

The convenient way to create the ground truth data is to let a human expert do it manually. Although the determination of the context of an image can be subjective and therefore not always deterministic – there are parts of a web page that in some situations belong to the context, in others not (e.g. the complete article text can be omitted, when an detailed image caption is present) – we assume that the human expert at least has a good idea of what the image context should be. On the other hand, the effort needed to determine the image context manually is tremendous. The average time needed to process a web page containing 20 image-context pairs using an extraction tool [Tri10] that alleviates the process took 183 sec. As a consequence, the processing of 1,000 documents would take approximately 50 hours of work, and even then the collection may be not representative enough. This brought us to the decision to think about an automation of the process.

Finding a general extraction method that is able to create our ground truth dataset seems to be impossible since if such a method would exist, the problem of WICE would be solved. On the other hand, for one specific web document, it is possible to manually write an extractor application (supported by a human expert) that exactly determines the image context for the images within this document. This extractor can be based on rules determining the desired DOM structure.

Example 5. *Rule-based Web Data Extraction*

An example extraction rule for collecting the image context in a web document:

```
// 1st rule
IF ImgTag has ParentNode with Attribute a = "b"
  THEN
    extractAllTextNodes under ParentNode as ImageContext;
    exit;
// 2nd rule
...
```

The order of the rules is very important, since the images covered by different rules

may overlap. For some templates, rules can be much more complex, e.g., they can exclude particular textnodes under a parent node.

Example 5 contains a typical extraction rule that is based on particular HTML tag properties. The example rule is performed for a given `ImgTag` which contains the image for which the context is sought. The rule condition proves if the given `Tag` contains a parent node with an attribute „a“ having a value „b“ and collects all text nodes under the estimated parent node as the image context.

Instead of creating a tailored image context extractor for each web document – this would be even more time consuming than directly estimating the image context – we observed that there are collections of web documents for which one specific image context extractor would fit. These are web documents that are part of a common web site (e.g., Wikipedia) that is maintained using a *Content Management System* (CMS). Usually these documents are based on a common structural *template*. There are two different methods to gather documents that share a common template, either by crawling a specific web site or by recalling a web page that changes very often such as the start page of many news portals.

We decided to apply the latter method to collect the data collections. To be more detailed, the algorithm for building the collection is as follows:

```
LOOP until NrOfCollectedDocuments = M
  current := load source from www.example.com

  IF (difference(current, last) > threshold)
    THEN
      store current to disk
      last := current

  wait N minutes
```

The algorithm consists of one loop that is repeated until the desired number of documents M has been collected. Inside the loop, first the source code of the web document (here *www.example.com*) is downloaded as `current`. Then the difference of the current document to the last stored is estimated. We have tried different algorithms for computing the difference of two web page sources and we decided to compare the sources line-by-line. If the difference is greater than a predefined threshold, we store the current document to disk since it seems to differ significantly from the last stored. After that the “last” content is overwritten by the current and the algorithm waits a small period before repeating the whole process.

Finally, the result is a collection of web document that have different contents that are based on one template. For this template, a human expert can now define rules



Figure 5.3: Three different pages from the CNN News Page based on the same template

that extract the images and corresponding context. To do this, the expert only needs to analyze a small portion of the web documents, which is an enormous reduction of the needed effort compared to manual image context extraction. Figure 5.3 depicts different pages from CNN News that are based on the same template. As we can see, there are many elements as the header, the footer and the navigation bar that hardly differ. Also there are high structural similarities between the content articles.

The properties of the resulting test collections are summarized in Table 5.1. According to these values, to our knowledge this is the greatest existing test collection of extracted image-context pairs.

5.2.3 Extraction Methods

At this moment, we are not interested in concrete context extraction methods but more in the interfaces they should implement. We already described the input data as web documents from real web servers. In the last section, we further introduced the way how the documents are collected and stored.

All extracting methods can be based on a DOM parser that processes the string representation of a documents and translates it to a DOM tree. The further processing can then be accomplished on this tree structure. However, there are also extraction

Collection	#Documents	#Images
BBC	1000	7013
CNN	1000	13286
Golem	1000	3879
Heise	100	1776
MSN	500	12352
New-York Times	500	9826
Spiegel	1000	33745
Telegraph	500	9908
The Globe and Mail	1000	21507
Wikipedia	3000	6728
Yahoo! (english)	4000	44067
diverse (manual)	100	1140
total	13700	165226

Table 5.1: Test collections with total number of documents and images

methods that additionally investigate the geometrical properties of the DOM elements. In this case, a parser that is coupled to a web browser (e.g., Mozilla Firefox or Internet Explorer) should be applied.

Finally, an extraction method delivers the extracted images with corresponding context. The storage of these image and context pairs needs the definition of a unified output format. We prefer storing the complete text representation of the context over the storage of begin and end markers referring to absolute positions in the source code of the original documents, since the first does not require any retainment of source documents.

For each document the image URL and the corresponding context are stored in a Comma Separated Values (csv) file. This format was used for both collections, the extracted results as well as the ground truth. The two main advantages for csv-files are the easy access (since no special parser is needed) and the high portability.

5.2.4 Performance Measures

An objective comparison of the extracted context with the gold standard needs appropriate performance measures which will reward any congruence between the two data

sets and penalize any divergence.

The specification of image context can even be not deterministic among human experts, thus testing on exact matches between extracted and ground truth context poses a too strong criterion. Instead, a partial accordance between the experts judgments and the output of a context extracting algorithm should be considered.

In Section 4.3, we have shown that the concepts of the formal definition of WICE (see Definition 8) can be transferred to the basic concepts of IR. In doing so, the WICE task can be interpreted as a special IR task and thus the evaluation of WICE can be viewed from the IR perspective. In other words, common IR measures can be used to quantify the performance of WICE methods.

In Chapter 2, we have introduced the IR performance metrics of precision and recall. From the data extraction point of view, the concept of precision P is defined as the ratio of the correctly extracted objects to all extracted objects, and the concept of recall R is the ratio of the correctly extracted objects to all relevant objects.

Since these measures complement each other, the harmonic mean of both comprised in the F -score provides a combined performance measure to compare the extracted objects to the relevant objects. It is defined as follows:

$$F_{\text{score}} = 2 \cdot \frac{P \cdot R}{P + R}.$$

The usage of the mentioned IR measures in the context extraction scenario requires a specification of what the extracted and relevant objects are. As the context is written in natural language, we can simply use the individual words of the document as these objects. The context can then be represented as a sequence of words. To compute the intersection of two word sequences – this is a required step for both precision and recall – one could either compute the intersection of the sets of words contained in the sequences, or, what has proven more effective, one could compute the *Longest Common Subsequence* (LCS) of the two word sequences. A fast algorithm for LCS that was used in the implementations has been proposed by Hirschberg in [Hir75].

Further, in order to analyze the stability of the WICE methods within a specific document collection C , we compute the *standard deviation* of the F -scores within a collection C , which is defined as

$$\sigma(C) = \sqrt{\frac{1}{N} \sum_{i=1}^N (F_i - \mu)^2},$$

where F_i is the average F -score computed over all image-and-contexts of document $d_i \in C$ and μ is the average F -score over the complete collection C . Further N is the number of documents in C .

During the evaluation studies that we performed to measure and compare the extraction quality of the common WICE methods from the literature, we noticed that a standard deviation value lower than 0.03 signalizes a stable F -score result for a given collection. For values over 0.15, we observed very alternating F -scores meaning that sometimes very high F -scores were reached and sometimes they were significantly lower.

The translation of the concept of precision-recall diagram is not suitable for evaluation of the WICE task, since it requires special adjusting parameters in the WICE methods that specify the length of the extracted context that are hardly given.

5.2.5 Framework Output

The final output of the evaluation framework are the different evaluation results that were computed for different combinations of WICE methods, evaluation metrics and ground truth collections. For example, one data block in the output could be the recall value of the extraction done with the N -terms extraction method, where the CNN ground truth collection was used.

Here, the output is again generated as a csv-file allowing us to use the data as input in a visualization software that creates appropriate diagrams in order to better present the results to a user. Or the output data can be passed to other applications where other useful computations can be performed (e.g., the computation of the standard deviation).

5.3 The Evaluation Framework in Practice

The functionality of the proposed framework has been tested with the common WICE methods from the literature, that were introduced in Chapter 3.1, namely, the *N-Terms window* (NT) extractor, the *paragraph*(PAR) extractor, the VIPS-based extractor (VIPS), the *Monash*(MON) extractor, and – as a baseline to clarify the performance gain of using extraction methods – the *Full-Text* (FULL) extractor.

The Full Text extractor does not contain the complete web document but only its plain text without any code. For the N -Terms window algorithm, we have implemented the standard fixed-window version as presented in Algorithm 1 with a window size of 10 and 20 terms. The paragraph extractor is an exact implementation of Algorithm 3. Also the Monash extractor implementation corresponds to that what has been described in the final state diagram in Figure 3.1.2.

As stated before, the VIPS algorithm is a web page segmentation method that divides a web page into blocks. After that, the extraction-by-page-segmentation method can be applied (see Algorithm 4). For the page segmentation step, the publicly availa-

ble demo [Cai11] of the VIPS algorithm has been adapted to handle a list of documents and applied as a preprocessing step. In particular, VIPS gets a web document as input and produces an XML-like document, which embeds the original page content into containers that represent the single page blocks. In the next step, the generated XML file is passed to a rule-based wrapper that assigns to each web image the complete text of the common block. The PDoC value that specifies the desired granularity of the blocks was set to 5, 6 and 7 (cf. [FHB09]).

All extraction methods were executed and evaluated in the same environment under the same conditions in respect to input and output specification and document preprocessing.

5.3.1 Time Analysis

Additionally to the quality metrics that were proposed in the last section, we have also measured the average running time needed to process a single page for each extraction method on a system with a Core 2 Duo processor at 3,2 GHz and 2 GB RAM memory. Since the input documents are locally stored, there is (almost) no loading time for the document source included. However, the VIPS method needs to render the complete web page and therefore all contents such as images and required scripts had to be loaded in advance (these were not stored locally by the crawling process). Furthermore, it is important to mention that the VIPS method consists of two steps, the segmentation and the WICE step, that are separately executed in different environments, therefore we have firstly estimated the average time for each step and summed the values up to an overall average time. The results of the time analysis is depicted in Figure 5.4.

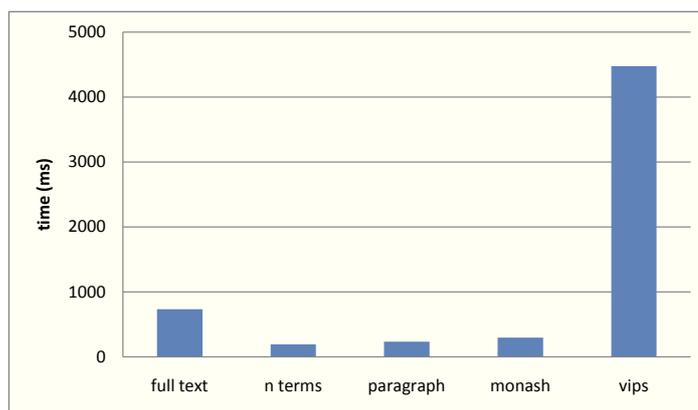


Figure 5.4: Average running times for different WICE methods over all collections

The first observation is that the running time of VIPS is significantly higher compared to the other methods. The reason for this is surely the additional time needed to load different page resources and to render the web page as we already mentioned

above.

The more interesting observation is that all extraction methods (except vips) are faster than the baseline method. This encourages us to replace the full text extractor by WICE methods in real world applications that need image descriptors and used full text until now.

5.3.2 Precision, Recall, F -score

The evaluation results based on the proposed evaluation metrics were computed for each ground truth collection. For a better overview, we will explain our observations on only three significant collection results. The other collections gained similar results and the corresponding diagrams can be found in the appendix. The selected collections were chosen based on their complexity. We will start with the Golem-Collection, go on with the Spiegel-Collection, and finally present the results obtained for the Manual-Collection.

Analysis of Low-Complexity Web Pages

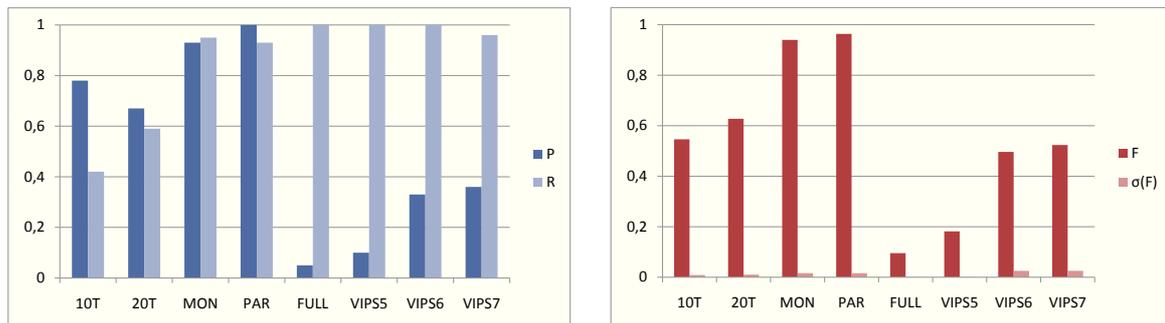


Figure 5.5: Golem-Collection Results: (left) precision (P) and recall (R), (right) F -score (F) and standard deviation of F -score ($\sigma(F)$)

The Golem-Collection has been selected as a low complexity collection because the structure of the image context on the Golem pages is (almost) always the same: the context consists of one or two headers and the main article. The evaluation results are summarized within Figure 5.5 that additionally to F -score and its standard deviation, contains a diagram with the corresponding precision and recall values. Precision and recall were depicted for a better explanation of the F -scores.

The first observation is that all extraction methods perform better than the full text extractor. Obviously, while the recall value of full text is always optimal (the sought context is always contained within the full text), the precision is very low (the more content a page has, the lower the precision), which consequently results in a very low F -score.

The F -score results of the N-Term Extractor range in the middle third. A source code analysis reveals that the image context is mostly placed between a short header (in avg. 6 terms) and a longer text (in avg. 50 terms). Thus the precision for the 10-Term Extractor is high, because in average there are only 4 of 20 terms that are falsely assigned to the image context. However, the recall for the 10 Terms Extractor is significantly low, because a great part of context that is placed after the image is withdrawn from context. On the other side, the 20-Term Extractor encompasses a greater portion of the text after the image and has therefore a higher recall value. However, at the same time the 20-term window contains more falsely assigned terms and therefore has a lower precision. It is also interesting to see that the standard deviation of the F -score for both N-Term methods is very low indicating the very similar structure of the image contexts within this collection.

The VIPS extractor with a PDoC value of 5 has a result that is quite similar to the full extractor: a low precision and a perfect recall. This indicates that the page segmentation algorithm produces too broad page blocks that always contain the right context but additionally a lot of other content. With a higher PDoC value (6,7), the granularity of segmentation gets finer resulting in a higher precision. However, while the recall for VIPS 6 is still perfect, it gets lower for VIPS 7 which means that some page blocks are already smaller than the image context block. An eligible question is whether the F -scores get higher when we apply a higher PDoC value next; in our experiments, applying a higher PDoC value mostly resulted in a very fine partitioning with blocks that only consisted of the image. For this reason a higher PDoC value than 7 has been omitted (see also [FHB09]). The standard deviation of the F -score is very low for VIPS 5 and increases with a higher PDoC. This is because for some image-context pairs the current segmentation granularity is suitable while for others not. This is a general problem of VIPS that sets one global block granularity for all images within a web page.

The DOM-based methods outperform the other methods with nearly perfect extractions. Both methods benefit from the simple and clean structure of the Golem documents – the image context has usually one common parent with the image – and that is exactly the strategy that both methods follow. We recognize a small advantage for the Paragraph method which starts with the image node and travels up the tree until a parent node is reached that contains textnodes in its subtree that are then collected as image context. In more than 90 percent of image-context pairs this is the perfect extractor. There are just a few cases where the image context is broader. This observation justifies the perfect precision and the high (but not perfect) recall. On the other hand, the Monash Extractor has different extraction rules that either produce the same output as the Paragraph Extractor (listed image) or select a broader environment (unlisted image). Therefore the recall value of Monash is higher compared

to Paragraph but because sometimes the context is chosen broader than necessary the precision is lower. For both methods the standard deviation of the F -score is low, which justifies their stability on this collection.

Analysis of Intermediate-Complexity Web Pages

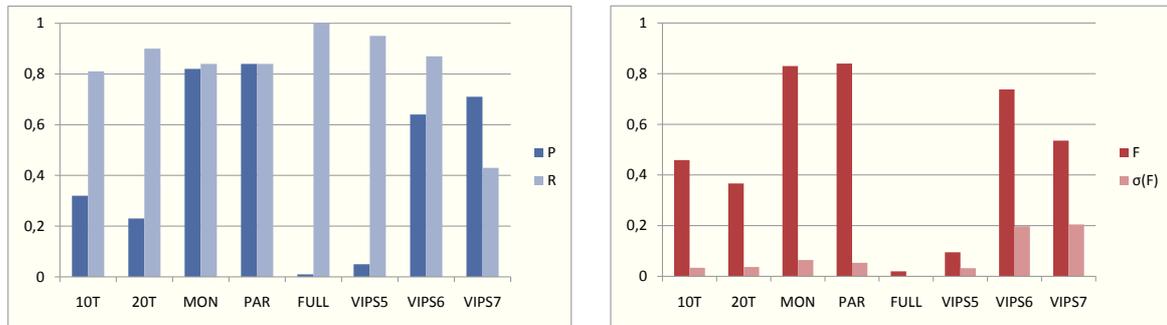


Figure 5.6: Spiegel-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

Figure 5.6 depicts the evaluation results performed on the Spiegel-Collection which we have categorized to an intermediate complexity collection. The reason for this categorization is the following; the documents contained in this collection are long and include many images. However, most image-context pairs are placed within the same structure patterns. Usually, there are 4-5 different structure patterns on each document: a top article, further articles, video articles, advertisements.

The N-terms methods deliver quite opposite evaluation results compared to that on the Golem-Collection: the recall is relatively high while the precision is low. The reason for this phenomenon is the high occurrence of images with short contexts that appear with video-preview images and advertisements on Spiegel documents. If the image context consists only of few terms either before or after the image then the surrounding window of terms includes the context and the recall of this extraction methods is very high. On the other hand, if the context is either before or after the image then the precision of the surrounding window approach is at best 0.5, and much lower, if N is greater than the length of the context.

The VIPS-based extractor with a PDoC of 5 has a similar output as when applied on the Golem-Collection. The reason for the high recall and very small precision lies again in the too broad page blocks returned by the page segmentation. The output gets better with a PDoC value of 6 since the precision increases because the blocks that include the context get smaller. However, for a PDoC value of 7 the blocks get too small and although the precision increases a little, the recall decreases significantly and causes that the F -score is lower than for the output of VIPS with PDoC of 6. It

is interesting to mention that for VIPS with PDoC of 6 and 7 the standard deviation of the F -score is very high that indicates the instability of the VIPS method on this collection: for some image-context pairs the segmentation fits optimal and for others either the blocks are too granular or too fine.

Again the DOM-based approaches gain the best results with a small impairment compared to the output of the Golem-Collection. To describe it with the terms used in the Monash algorithm, the Spiegel-Collection contains many listed as well as unlisted images and the rules to recognize these image types as applied by the Monash Extractor are sometimes misleading.

Analysis of High-Complexity Web Pages

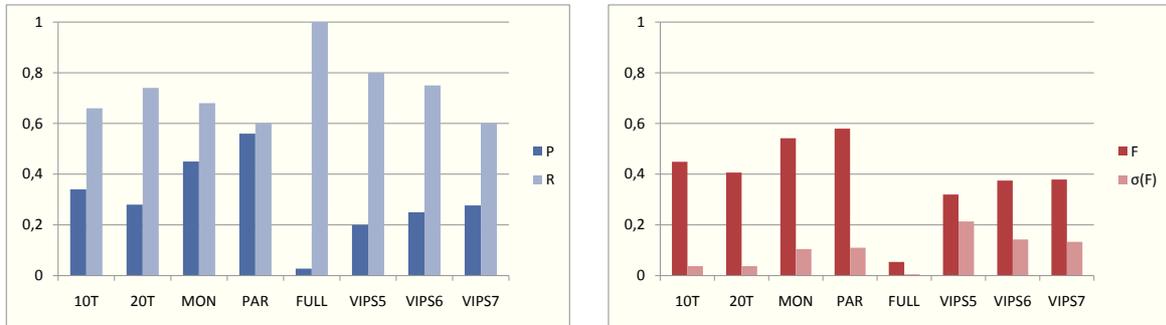


Figure 5.7: Manual-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

The Manual-Collection consists of several documents from different web sites with different design structures and from different web document categories. Although this collection is relatively small, we believe that its variety covers many real world document patterns and thus represents the Web best.

Starting with the N-Term extractors, we observe that they again range in the middle third. Similar to the results on the Spiegel-Collection, the recall is again high, while the precision is lower. We cannot investigate all documents to exactly estimate the reason for this behavior as we did for the other collections, but from the experience with the Spiegel-Collection we can conclude that in this collection should be lot of short image contexts.

The VIPS methods perform even worse: while the recall is ordinary high, the precision is lower which at the first glance indicates a coarse segmentation. However, if we look at the high standard deviation values of the F -scores, we are faced the principal problem of the VIPS Extractor, which can not handle the several image-context sizes that are placed in blocks with different granularities. A possible improvement could be reached with a dynamically selected PDoC value for each image in the web page

which is beyond the scope of this thesis.

The next interesting observation is that the DOM-based approaches that gained the best results on the other collections, loose much in performance on the Manual-Collection. While the other collections contained only well structured documents from news sites, the documents in the Manual-Collection are poorly structured and partly not well-defined. Of course this is a disadvantage for the methods that highly rely on this document structure.

5.4 Summary

In this chapter, we presented the evaluation framework for image context extraction tested with the common WICE methods from the literature. The framework has been built very modular and therefore remains highly extensible. Each module (i.e., the collections, the WICE methods and the evaluation metrics) can be extended by more features.

The first practical application of the framework for testing the existing WICE methods demonstrated the abilities of the framework and showed its benefits for the analysis of WICE algorithms. In the following chapters, we will apply this inevitable tool to estimate the performance of our new proposed WICE methods and to compare their outputs with that of other existing methods.

6

WEB IMAGE CONTEXT EXTRACTION BY NEAREST ARTICLE DETECTION

As we mentioned in the introduction, the web image context can be very useful for different applications provided that the image context can be properly determined and extracted. In the last chapter, we have presented our evaluation framework for WICE and discussed the evaluation results obtained by testing common image context extraction algorithms. The main outcome of the evaluation was that firstly almost all methods gain a much better F -score compared to the baseline full-text extractor, and secondly that each method has some shortcomings and there is still space for improvement. In particular, our investigations during the evaluation process discovered a high variability of image context structure among the documents. We were faced with different image context styles ranging from small image captions to long news articles. Thus the extraction quality of static approaches like the N-Term Extractor are limited.

In this chapter, a novel approach to image context extraction is presented that is more adaptive and thus able to handle the high divergence of image context styles.

The basic idea of this approach has been introduced as a part of the master thesis [Alc07] and as a contribution to the GvDB Workshop [Alc08] in a condensed version. The extension of the image association method by the 2-DOM model was published at the 3rd International Workshop on Semantic Media Adaptation and Personalization [AC08].

This chapter is organized as follows: first the basic idea of the proposed method is described in Section 6.1. Then, Section 6.2 describes the article detection process, which is further subdivided in *content extraction*, *content classification* and *content grouping*. Following that, Section 6.3 presents the image-to-article mapping process,

starting with the general idea to estimate proximity of contents in a web document, followed by the description and our solution for the DOM table problem. Finally, the method is evaluated and compared to other algorithms in Section 6.4.

6.1 The Idea behind WICE by Article Detection

Images contained in a web document usually belong to a textual part of the document which we refer to as an *article* in this chapter. Hereby an article is not necessarily a traditional news article but can also be a short text like the image header or a longer image description. The main scope of the proposed approach is to find a general way to detect all articles contained within an arbitrary web page. Once the articles are determined, an appropriate method that assigns the images to the detected articles is applied. The basic steps of the suggested approach are visualized in Figure 6.1.

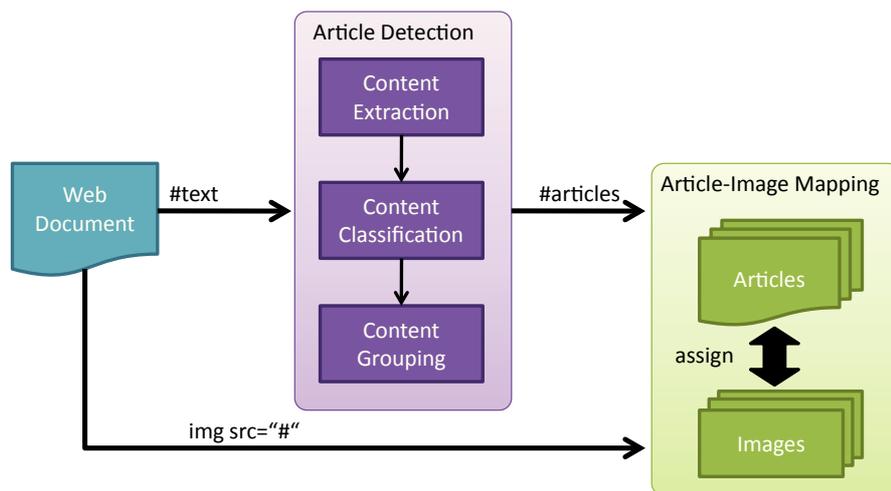


Figure 6.1: Image-to-Article Mapping: A Schema of the main Concepts.

As shown in this figure, the article detection plays a central role since it is responsible for finding the candidate contexts that are later associated with the images. In particular, first the contents are extracted from the web page source; these contents are then classified into different logical parts that an article can be composed of (e.g., main title, subtitle, abstract, main paragraph). This classification is performed using classification rules based on simple structural features extracted from the corresponding HTML. In the next step, the different contents are grouped to candidate articles with a grouping algorithm that exploits the previously assigned content classes. Finally, the image-to-article mapping module takes the web images that can be easily extracted from HTML and maps them to one of the computed candidate articles based on a proximity measure.

6.2 Article Detection

The article detection comprises three processing steps: content extraction, content classification, and finally, content grouping. In the following each of these steps will be described in detail.

6.2.1 Content extraction

The textual contents of a web page are extracted using an HTML parser (see Chapter 2.1.2), which collects all text nodes under a documents root node. This is done by traversing the DOM tree of the web page in *preorder* and whenever a textual node is detected, the node is stored into a content vector. Consequently, the nodes in the content vector are sorted by the parser. This guarantees that contents, which are closely to each other in HTML, will also be the same in the content vector.

However, not all text nodes belong to the displayed content of a web page, e.g., meta information, comments and script contents are also handled as text nodes by common HTML parsers. These contents are not part of any article and thus should be filtered. The corresponding parent nodes allow to detect such text nodes easily.

The last important task in the extraction process is to group contents that were separated in DOM due to formatting properties, but logically belong to the same structural part. This may happen when within a paragraph some specific words are especially emphasized. In this case, the HTML parser detects separate text nodes and treats them as they would be different contents. In order to avoid a separation of such contents, we exploit the *breaks flow* property of HTML tags. Only if a specific text node “breaks the flow” in the document, it should be placed in a new content, otherwise it extends the previous text node. Text nodes that break the flow are for example child-nodes of *H1-H6* or *P* tags. The Java HTML Parser [Osw06] maintains a specific property for each node that indicates if it breaks the flow or not. The following example should provide a better view on the extraction process.

Example 6 (Content Extraction). *Figure 6.2 shows two news articles from Spiegel ONLINE as displayed in a browser and the corresponding source.*

At the beginning, the text nodes are extracted by the parser and are put into a content vector where they are further processed. In the filtering step, nodes that do not contribute to the displayed content are filtered: here, we remove the script tags. The final step seeks for non-breaking parent tags and puts the corresponding text contents together: in our example, the text nodes that are under the paragraph nodes are collected to one text content since their parents are non-breaking tags. The resulting text content vector is (②, ③, ④, ⑥, ⑦, ⑧).



Figure 6.2: Two news articles from Spiegel ONLINE: (left) browser representation, (right) corresponding HTML source

6.2.2 Content Classification

The extracted textual contents possess different clues that allow to classify them to different logical parts of an article. In our approach, we decided to build a simple classifier which uses only the HTML title tags $h1$ - $h6$ and the content length (number of characters) as features to classify them to logical classes.

In order to determine the logical classes, an empirical study was conducted, where the logical structure of various web articles has been investigated. As a result, we define nine different classes, to which the textual contents can belong to (see Table 6.1).

The type number indicates the order in which the contents appear in articles and can also be used as a short identifier. Following the type numbering, an article is expected to start with one or more headings, then it is continued by a short abstract which is followed finally by a longer paragraph (also more of these longer paragraphs can follow). The benefits of the type numbers will be more clear in the next section.

Example 7 (Content Classification). *Our example in Figure 6.2 produced the content vector (2), (3), (4), (6), (7), (8). According to the classification rules presented in Table 6.1, we get the following class assignment for the contents.*

<i>Content</i>	②	③	④	⑥	⑦	⑧
<i>Content Type</i>	2	3	9	2	3	9

Type	Logical class	HTML Clue
1	Headline 1	Parent is a H1 -node
2	Headline 2	Parent is a H2 -node
3	Headline 3	Parent is a H3 -node
4	Headline 4	Parent is a H4 -node
5	Headline 5	Parent is a H5 -node
6	Headline 6	Parent is a H6 -node
7	Short text	Parent is not a headline and the $textlength < 40$
8	Abstract	Parent is not a headline and $40 \leq textlength < 100$
9	Long text	Parent is not a headline and the $textlength \geq 100$

Table 6.1: Logical classes of text contents appearing in most news articles

6.2.3 Grouping Contents to Articles

Before presenting the algorithm that groups the contents to articles, we need to formalize the given concepts.

Definition 9. *Web Content Ordering*

- *Let N be the set of content nodes of a web page – nodes that are either image nodes or text nodes. For $n_i \in N$ the index $i \in \{1, \dots, |N|\}$ determines the position of n_i in the source code of the document. In this chapter, the set N will be referred to as vector because the elements of the set are ordered.*
- *A text content is a vector of text nodes. The text node order within a text content is derived from the order of text nodes in source code.*
- *An article is a vector of sequential text contents. The text content order within an article can also be derived from the order of text nodes in source code.*

The content grouping is described in Algorithm 5. As input, we get a vector of ordered text contents \mathcal{T} of the document. The algorithm walks iteratively through \mathcal{T} and groups sequential text contents to articles. The key function in this algorithm $extendArticle(a, t)$ decides if a given article a should be extended by a text content

Algorithm 5: Building Articles from Text Contents**Data:** Vector \mathcal{T} of text contents of a web page**Result:** Vector A of articles

```

1  $a = \text{new Vector}();$ 
2 while  $\mathcal{T}.\text{hasMoreTextContents}$  do
3    $t \leftarrow \mathcal{T}.\text{getNextTextContent};$ 
4   if  $a = \emptyset \vee \text{extendArticle}(a, t)$  then
5      $a.\text{add}(t);$ 
6   else
7      $A.\text{add}(a);$ 
8      $a = \text{new Vector}();$ 
9      $a.\text{add}(t);$ 
10  end
11 end

```

t . In the following, we will first explain the idea behind this function and then give a formal definition.

As we have already mentioned in the previous section, the logical structure of an article usually follows a pattern. For example, if an article starts with a title and is followed by a paragraph of text, then it can not be continued by a title. We exploit this fact to divide sequential text contents of a web page into separate articles. The text content type defined in Subsection 6.2.2 is used as decision feature. The decision whether an article should be extended by a textual content depends on the type number of its last text content. Text contents of a type which is smaller than 9 (longer text paragraph) can only be followed by a text content of a greater type. However, a longer text paragraph (type 9) can be followed by another longer text paragraph. These decision rules are formalized in the following boolean function.

Definition 10. Let t be a text content and a an article, which is an ordered set of text contents. Further t_{last} is the last text content in article a and consider the notation t^* as the type (see Table 6.1) of the text content t . The boolean function $\text{extendArticle}(a, t) \rightarrow \{true, false\}$ that checks whether the article a should be extended by a textual content t is defined as follows:

$$\text{extendArticle}(a, t) = \begin{cases} true & : (t_{last}^* < t^*) \vee (t_{last}^* = t^* = 9) \\ false & : \text{otherwise.} \end{cases}$$

Example 8 (Building Articles). Going back to our example from Figure 6.2, the article grouping algorithm gets the content vector $(\textcircled{2}, \textcircled{3}, \textcircled{4}, \textcircled{6}, \textcircled{7}, \textcircled{8})$ as input and creates a new article a_1 with content $\textcircled{2}$. In the next step, a_1 is extended by $\textcircled{3}$ because the type of $\textcircled{3}$ is greater than the type of $\textcircled{2}$. Then a_1 is again extended by $\textcircled{4}$ because the type

of ④ is greater than the type of ③. However, in the following step a_1 is completed and a new article a_2 is created with content ⑥ because the type of ④ is greater than the type of ⑥ and the output of the `extendArticle` function is false. In the following a_2 is twice extended by ⑦ and then by ⑧, until the algorithm finally terminates. As output we get two articles $a_1 = (②,③,④)$ and $a_2 = (⑥,⑦,⑧)$.

6.3 Image-to-Article Mapping

The output of the article detection process is a list of candidate articles extracted from a given web document. On the other hand, we can easily determine the web images of the same document by extracting all image contents. In order to find the correct article in the article list that an image belongs to, we can exploit the positioning information of image and articles provided in the source code. For this purpose, we need to define a distance function, that can measure the proximity of articles to the web images.

Definition 11 (Distance Function). *Consider the vector N of content nodes of a web page. According to Definition 9 the nodes in N are ordered by their position in the corresponding source code. Let n_a, n_b be two nodes in N , then the distance $d(n_a, n_b) \rightarrow \mathcal{N}$ is computed as follows:*

$$d(n_a, n_b) = \|a - b\|,$$

which is the absolute distance between the indexes of the two nodes.

Using this metric, we are able to compute a simple proximity of two web contents that reflects their relationship in the source code of the document. The contents of a web document are enumerated by their appearance in the document and in this way mapped into a one dimensional space, where the defined distance function can be applied. The distance function corresponds to the Euclidean Distance in one dimension.

Example 9 (Measuring Distance). *Consider the following example content vector:*

$$(①, ②, ③, ④, ⑤, ⑥, ⑦, ⑧)$$

The distance of content ③ to content ⑦ is computed as follows:

$$d(③, ⑦) = \|3 - 7\| = 4,$$

that just corresponds to the absolute distance between the indexes of the contents.

This measure can be applied to compute the distance between an article a (which is a vector of text contents $a = (a_1, \dots, a_{|a|})$) and an image i of a web document by taking the minimum distance of i to any text content a_k of article a , with $k \in \{1, \dots, |a|\}$:

$$d(i, a) = \min_k d(i, a_k).$$

In the final step, this measure is used to map the images of the document to the computed articles that we obtained as output of Algorithm 5. To be more precise, an image i from the document is mapped to the article A_j that has the minimum distance $d(i, A_j)$, with $j \in \{1, \dots, |A|\}$.

6.3.1 The TABLE Problem

The distance function defined in the last subsection exploits the ordering of the content in the source code to compute their distance. While the contents are represented in one dimension in the source code, the contents are usually displayed in a two dimensional plane in the browser representation. Many web page designers misuse the HTML Table concept to place the contents on an appropriate position within this two dimensional panel. However, in the HTML source code, a TABLE is defined in a one dimensional manner and the problem depicted in Figure 6.3 occurs.

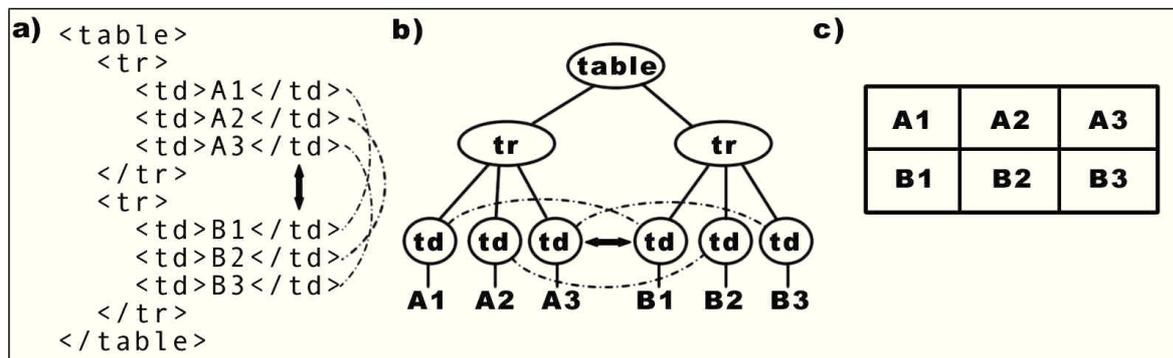


Figure 6.3: A TABLE in a) source code, b) DOM tree and c) browser representation. The dotted lines are relationships that can not be detected in the one dimensional representations. The solid line is a relationship that does not exist in the browser representation.

The problem means that there are relationships in the browser representation that can not be found in the other representations and vice versa, which leads to false assumptions performed by the defined distance function. For example, if there is an article in cell A1 and an image in cell B1, the distance function is not able to reflect that A1 and B1 are next to each other. However, A3 and B1, which are far from each other in the browser representation, are neighbored in the DOM and source code representation.

This problem has been addressed in [AC08] and there an extension of the traditional DOM model to a *Two Dimensional Object Model* (2-DOM) has been presented, which maintains for each node information about its direct neighbors in all directions (north,

west, south, east). In the 2-DOM model, the TABLE example from Figure 6.3 is represented as shown in Figure 6.4.

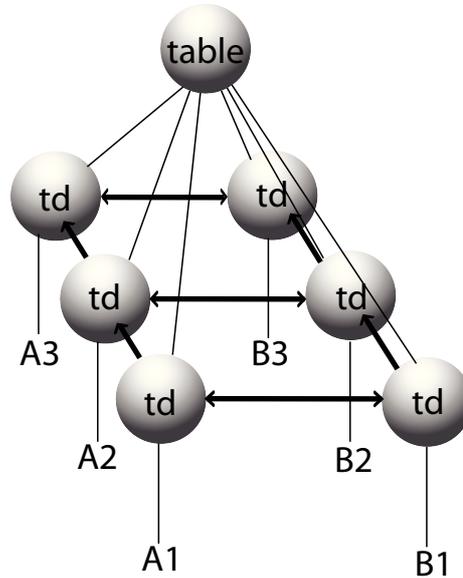


Figure 6.4: An example of an HTML table in 2-DOM representation.

The advantage of the 2-DOM representation over the traditional DOM representation is self-evident: the relationships between the table cells in 2-DOM correspond to the relationships in browser representation.

6.3.2 Measuring the Relationship of Contents in 2 Dimensions

The ideas behind the 2-DOM model can be easily implemented by treating each content node as an element in a two dimensional grid represented by two indexes (x, y) . For each content node that is not within a table cell, the x -value is determined by the position in the source code (vertical ordering). For a content node within a table cell, we parse the table structure and use the y -value to identify the ordering in horizontal direction.

In the example from Figure 6.4, the following grid representation would result:

Content Node	A1	A2	A3	B1	B2	B3
Grid Coords	(1,1)	(1,2)	(1,3)	(2,1)	(2,2)	(2,3)

Now we can define a distance function for content nodes that is based on the new two dimensional representation.

Definition 12 (Two-Dimensional Distance Function). *Let n_a, n_b be two content nodes in a web document and (a_x, a_y) and (b_x, b_y) be the corresponding grid representations of n_a and n_b . Then the distance $d(n_a, n_b) \rightarrow \mathcal{N}$ is computed as follows:*

$$d(n_a, n_b) = \|a_x - b_x\| + \|a_y - b_y\|.$$

This distance measure is also known as the *Manhattan-Distance* on a 2-dimensional vector space. If no tables are used, then this function is reduced to the distance function defined in Definition 11.

The distance function can be applied in the above algorithms to map a web image to its next article and in this way to find the corresponding image context.

6.4 Evaluation Studies

Until now, we described the article-based context detection approach from a theoretical point of view. In particular, the extraction algorithm and its extension to handle the HTML table problem has been introduced; however, there is still no information about the practical appliance and usability of the algorithm.

In this section, first some issues are addressed that became apparent when the algorithm was applied on real web documents. Following that, methods that determine how to deal with these issues will be presented.

Then the proposed WICE method is evaluated and compared to other existing WICE methods within the evaluation framework introduced in Chapter 5. Finally, significant results are presented and discussed in more detail.

6.4.1 Practical Issues of Article-based WICE

As we described in the previous sections, the article-based context extraction method consists of two main steps: the article detection and the image-to-article mapping. For the latter step, a distance function has been proposed that computes the proximity of image to all candidate articles and selects the article with minimal distance as image context. However, this mapping step is not always deterministic: the distance between image and article can be minimal for more than one article and in this case, the algorithm has to decide arbitrarily. As an example, we refer again to Figure 6.2 where the computed articles are (②, ③, ④) and (⑥, ⑦, ⑧). The distance of the image ⑤ is 1 to both articles and both could be assigned to the image.

In order to solve this problem, we have formulated the following additional rules that have to be fulfilled:

- each article is mapped to exactly one article, and vice versa,
- images for which the association is deterministic are processed first,
- remaining images are mapped to the longer articles.

In the example of Figure 6.2, we would first assign article (②, ③, ④) to image ①, and the remaining article (⑥, ⑦, ⑧) to image ⑤.

6.4.2 Results

Extraction Quality

The standard Article-based Extraction Method (ART) and its extended version (ART2D), which handles the discussed HTML table problem, have been applied on all 12 test data collections that are part of the evaluation framework. Instead of presenting all result graphs for each test collection and extraction method, we will select only a few significant graphs that describe the behavior of the methods best. Again, we refer the reader to the appendix for complete evaluation details.

BBC Collection First, the results for the BBC Collection which contains web documents that are structured by HTML tables are presented. Figure 6.5 contains the precision, recall and F -score results for the two novel article-based methods and the common methods from the literature (for N-terms and VIPS method, we only depicted the method that achieved the highest quality).

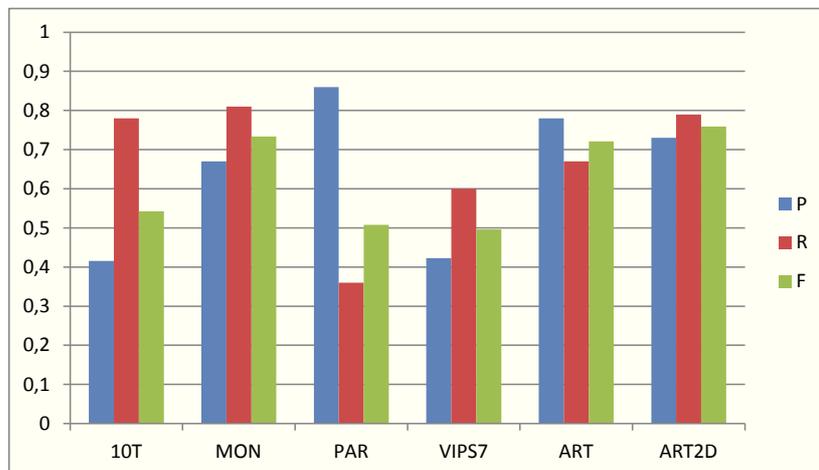


Figure 6.5: Precision, Recall and F-score Results for the BBC Collection

The results show that the article-based methods reach the highest F-scores. This indicated that the defined rules for article detection are able to find most of the article groups in the web documents. It is further noticeable that the extended method that handles HTML table constructs performs slightly more efficient, since the method is able to map images to articles that are dispersed over different table cells. However, it seems that only a small portion of images and article pairs is hit by this extensional construct.

Heise-Collection The results for the Heise Collection are depicted in Figure 6.6. Heise documents have a really simple HTML structure, in particular the image-article pairs follow almost always the same structural pattern.

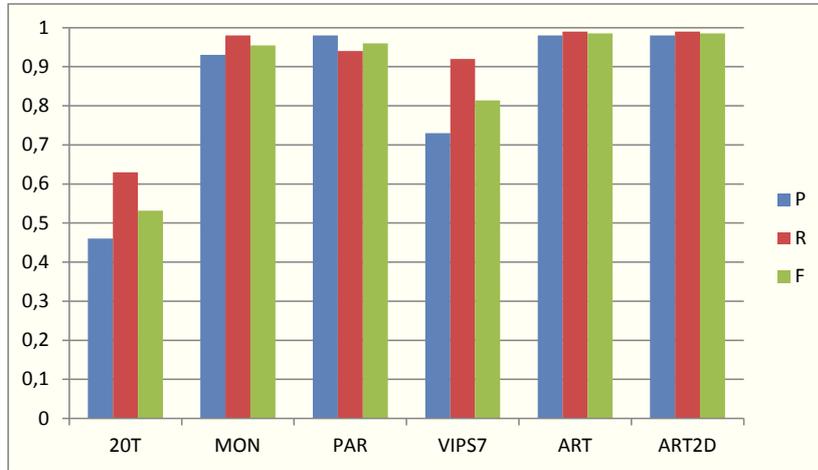


Figure 6.6: Precision, Recall and F-score Results for the Heise Collection

The graphic reveals that again the article-based method reaches the highest quality in terms of F -score (the extraction is nearly perfect), which is a consequence of excellently fitting rules for article detection and a well-fitting mapping strategy. However, this time, we do not observe a quality improvement by applying the extended algorithm (ART2D). This is reasonable, since the Heise documents do not contain any TABLE elements.

Manual Collection Figure 6.7 contains the results for the Manual Collection, the collection where we can not describe the web design patterns of the documents, since the documents were collected arbitrarily from different domains.

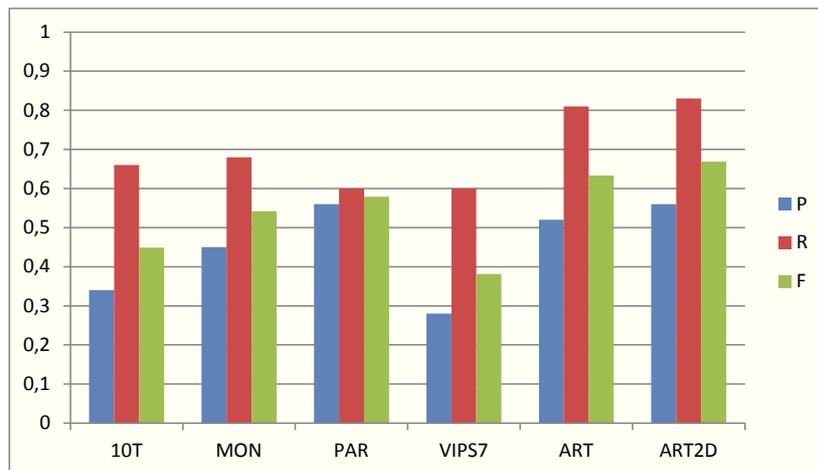


Figure 6.7: Precision, Recall and F-score Results for the Manual Collection

This arbitrariness can be recovered in the evaluation results of all methods applied on the Manual Collection: all results are significantly lower compared to the results obtained on the other collections. Surely, one reason for this observation is the less quality HTML source of documents that were partially collected from private home pages that usually are manually coded. Nonetheless, again the article-based methods are most effective while the extended version of the algorithm reaches the best values. Hence we can infer that in the Manual Collection there are documents that contain table-based structures.

Wikipedia Collection Finally, we present the evaluation results for the WICE methods applied on the Wikipedia Collection in 6.8. In order to better understand the chart, it is important to know how image and corresponding context look like in the documents of the collection: images are placed within a box which contains a short description (image caption). Usually, the web document contains a longer article that covers the main topic an image belongs to. Table are not used for layout and no advantages for the extended method is expected.

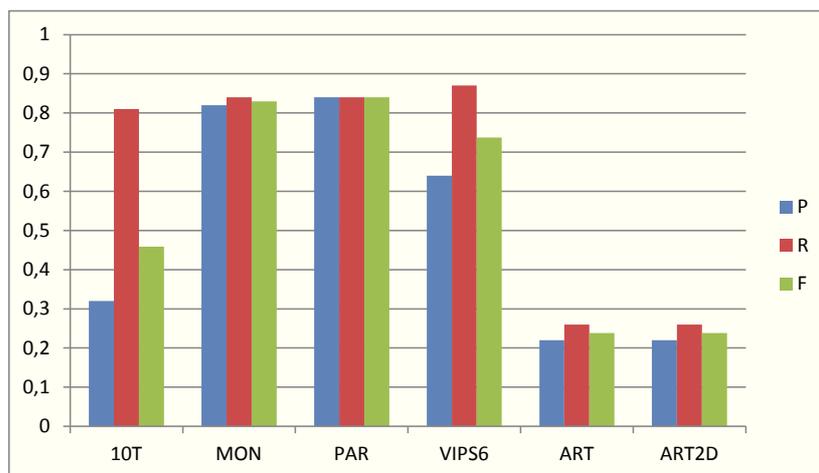


Figure 6.8: Precision, Recall and F-score Results for the Wikipedia Collection

The image-and-article pairs of the Wikipedia documents drive the article-based extraction methods to their limits: the quality of the context extraction performed by the article-based methods is significantly lower compared to the other extraction methods from the literature. N-terms is almost twice better, and the others up to four times.

The reason for that low quality of the ART method lies in the mapping strategy. In a typical Wikipedia document, we find an image surrounded by two articles: one is the image caption (the correct context), the other is a longer part of the main article. In this case, both articles have the same distance to the image and the selecting rule chooses the longer article which is the wrong decision in this case. The same scenario

is repeated for most of the images in this collection. However, the opposite decision would yield to false decisions in most other data collections.

Repeatability

As we explained before, the repeatability of a WICE method describes how far the single quality results of each image-context pair within a collection differ from each other, i.e. if the WICE method reaches similar quality results for all image-context pairs or not. For this purpose, we compute the standard deviation of the F -scores over all documents from a collection. The results obtained for the standard article-based extraction are depicted in Figure 6.9.

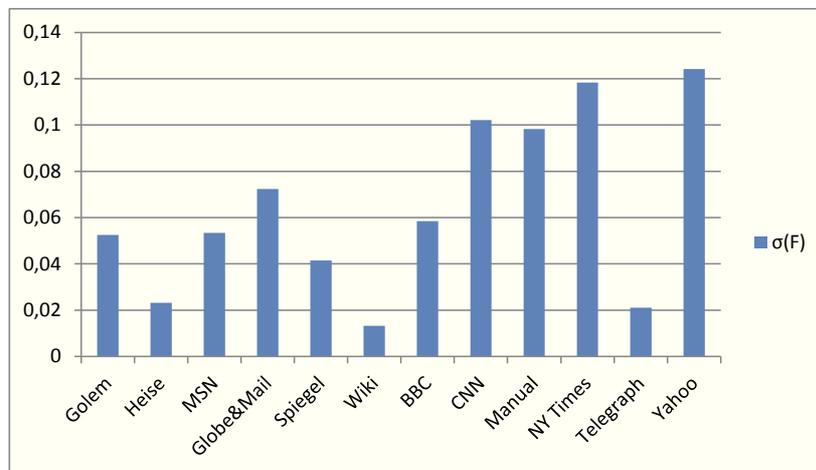


Figure 6.9: Standard Deviation of F -scores of Article-based WICE on different Collections

As we can observe, the standard deviation of the F -scores is variable over the different collections. The lowest values are achieved on the “Wikipedia”, the “Heise” and the “Telegraph” collection. This does not mean that the extraction quality on these collections is high but that the obtained F -scores – low or high – are reasonably stable. In other collections like the “New York Times Collection” and the “Yahoo Collection”, the quality results are not repeatable. This can be justified with the high variance of different article structures in the corresponding web documents.

Running Time

The chart in Figure 6.10 depicts the average times needed to process a single document. The average has been computed over all documents from the entire ground truth dataset. Beside the novel article-based extraction methods, there are also the processing times for other extraction methods from the literature included.

As we can observe, the average time needed to process a document with the article-based methods is slightly over one third of a second. The extended article-based method

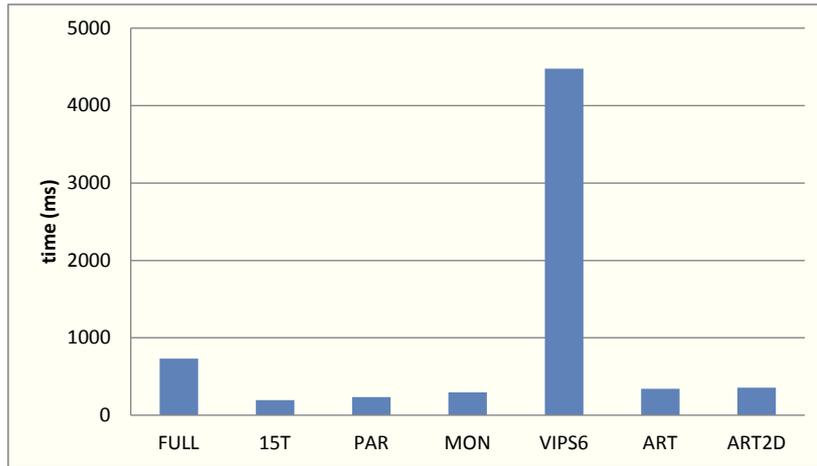


Figure 6.10: Average Processing Times over all Collections

does hardly affect the running time since in average it needs only 16 milliseconds more than the standard method. Compared to the other methods, the article-based extractor is arranged in the middle field, since it slightly slower than the N-terms method and the DOM-based methods, but faster than the full-text extraction and the much more time consuming VIPS method. In practice, the article-based methods could both be applied, since as we mentioned, the needed time is still under the time needed by the full-text extractor which has been applied in many real world applications.

6.5 Summary

In this chapter, we have introduced a novel WICE method that exploits the logical structure of the textual parts of a web document to build text articles. The articles are assumed to be possible candidates for an image context. A rule based algorithm that groups the text contents to articles has been proposed. Once the articles of a web document are detected, they have to be mapped to the images. This is accomplished using a proposed distance measure. The measure is one dimensional and lacks of performance when images and articles are placed into HTML tables, which are two-dimensional constructs. For this problem, the distance measure has been extended to work on a two-dimensional model that is able to handle HTML tables. Finally, the proposed methods have been evaluated using the evaluation framework that was introduced in Chapter 5.

7

WEB IMAGE CONTEXT EXTRACTION BY PAGE SEGMENTATION

What we call a web document today is in truth a big container for web content blocks that differ in semantics, functionality and importance. Besides the main content of a web document, we usually find different meta information related to the hosting website (e.g., company logo, legal notice), functional content increasing the usability (e.g., navigation bars, links to related stories) and different kinds of commercials. Each of these page blocks may include web contents of different media types such as text, image and others.

From the WICE point of view, the partitioning of a web document into blocks of web contents that share common semantics would solve the extraction problem, since then each web image can be associated with the textual contents of the common block.

Humans are able to identify such page blocks unconsciously by analyzing different web content properties, like their visual appearance and their semantics. However, the automatic segmentation of a web page is a complex task due to the high variety of web document designing structures. Moreover, web documents can be hierarchically structured and the granularity at which a block should be detected can vary from page to page.

Most existing page segmentation algorithms (see Chapter 3) apply simple heuristic rules that decide whether to break a segment at a specific position or not. A heuristic rule typically employs structural features of web contents as well as geometric properties gathered by rendering the contents in a browser application. In this way, each rule covers a specific design pattern and proposes an appropriate solution. However, the design patterns used to present contents on the Web are unlimited and in general can

not be covered by such simple rules.

In Section 7.1, we will first introduce a novel clustering-based approach to web page segmentation which is more adaptable and thus more suitable to the WICE problem. For this purpose, we investigate different web content properties, such as DOM-based, geometrical and semantical information. Some of the results of this analysis were published at the International Conference of Web Intelligence, Mining and Semantics (WIMS'11) in a condensed version [AC11b]. Based on that investigations, we develop an optimal page segmentation algorithm that should solve the WICE problem best, which will be presented in Section 7.2. The principal ideas of this novel WICE method have been published as a contribution [AC11a] to the Multimedia Conference (MMEDIA'11).

7.1 Page Segmentation by Clustering

7.1.1 Problem Formulation

Page Segmentation denotes the process of partitioning an input web document into basic segments of common functionality, structure and/or semantics.

In order to specify the page segmentation task formally as a clustering problem it is essential to define two important terms that are used in this context.

Definition 13 (Web Content). *As web content, we refer to the smallest indivisible content of a web page. In the DOM tree representation of a web document, the web contents correspond to the leaf nodes that are visible in the browser view of the document.*

Definition 14 ((Page-)Block). *The term (page-)block denotes a group of web contents that share a coherent topic, style and/or structure.*

Using this terminology, the web page segmentation task can be described as a *grouping of web contents to blocks*. Recalling the definition of clustering from Chapter 2.2.3 –

“Cluster analysis denotes the process of placing objects that are similar (or related) to each other in a same cluster and separating objects that are unlike (or unrelated) to each other in different clusters.”

– we can find a close relatedness to the description of the web page segmentation task. As a consequence the problem of page segmentation can be reduced to clustering of web contents. But prior to that we need to specify some concepts and methods that are required for clustering.

7.1.2 Web Content Representation

One major question of cluster analysis is how the objects (here the web contents) should be represented. Web contents own different properties that represent them in a different context. First, they are a part of a complex DOM tree (see Chapter 2.1.2) that can be directly derived from the HTML of their hosting page. This tree describes an hierarchical structure in which the contents are placed. The web contents can thus be represented by their positions in the DOM tree.

Secondly, once the contents are rendered in a browser application, they possess geometrical properties like an exact position in the output panel (see Chapter 2.1.1). A geometric representation of a textual web content or a web image could be a specification of the bounding rectangle that surrounds the content.

The third representation of web contents relies on the semantics of the web contents. While the semantics of a text can be easily extracted as annotations consisting of terms from the text (see Chapter 2.3), it is much more difficult to get annotations for images. However, instead of trying to solve the semantic gap for images, we extract the alternative text, the image title text or at least the image URL.

All three representations can reasonably be applied to describe the web contents in a web content clustering scenario. In the following, we are going to analyze which of the representations is most suitable for this task.

7.1.3 Distance between Web Contents

The different representations for web contents that were introduced above allow us to define different similarity or distance measures (see Chapter 2.2.2). These will be presented in the following.

DOM-based Distance

In the DOM tree of a web document the web contents are the leaves that are grouped by tags which are the inner nodes of the tree. Our aim is to provide a distance measure that will exactly map this tree structure into one distance value. We postulate the following preconditions for a DOM distance measure:

- Adjacent sibling leaf nodes possess all the same distance.
- The minimal distance of leaves belonging to different parents must be greater than the maximum distance of these leafs to their siblings.

The above preconditions are presented in Figure 7.1 in a visual manner.

A formalization of a novel distance function that satisfies these preconditions is given below.

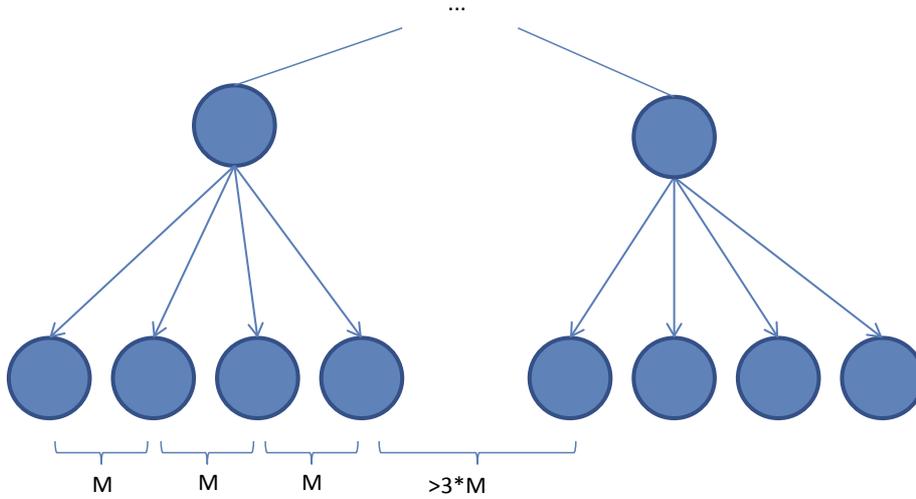


Figure 7.1: Preconditions for the DOM-distance function

Definition 15 (DOM-based Distance). *Let w be a web document and $D_w(N, E)$ the corresponding DOM tree, with N as a set of nodes and E as a set of edges. Further let $P = (p_1, \dots, p_n)$ be the sequence of all nodes in N ordered by traversing D_w in preorder traversal. The index $i \in \{1, \dots, n = |N|\}$ gives the order of $p_i \in N$ in the sequence P . Based on this formulation, we define the DOM Distance $d : N \times N \rightarrow \mathbb{R}$ for two nodes $p_a, p_b \in N, 1 \leq a \leq b \leq n$ (if $b < a$, wlog. switch p_a, p_b) as following:*

$$d(p_a, p_b) = \sum_{i=a+1}^b w_{p_i}, \quad (7.1)$$

where w_{p_i} is the block weight of element p_i , which has to be further specified. It can be easily shown that the DOM-based distance is a metric.

The block weights w_{p_i} can be explained as the cost needed to reach the content block of p_i from its preorder predecessor p_{i-1} . Therefore w_{p_i} corresponds to the distance $d(p_{i-1}, p_i)$ of two neighbored contents p_{i-1}, p_i . For nodes on the same tree level l , the block weights are all equal. On a lower level $l - 1$ the block weight has to be at least greater than the maximal distance of two sibling nodes at level l . The maximal distance of two sibling nodes on level l corresponds to the degree of the nodes on level $l - 1$ which is the maximal count of children of the nodes at level l .

Definition 16 (Block Weights). *Under the mentioned preconditions, the block weight function $w : N \rightarrow \mathbb{R}$ can be formulated by following recurrence:*

$$w(l) = \begin{cases} c & : d_l = 0 \\ d_l \cdot w(l+1) & : d_l > 0, \end{cases} \quad (7.2)$$

where d_l refers to the maximal degree of nodes at level l and $c > 0$ is the smallest distance between two leaves on the deepest level (can be set to 1 and does not affect the behavior of the measure). To apply w in Equation 7.1, we define function $l : N \rightarrow \mathbb{N}$ that delivers the tree level of a node. Thus the block weight w_p for a node $p \in N$ can be expressed by $w_p = w(l(p))$.

For clearance, consider the following example DOM tree in Figure 7.2.

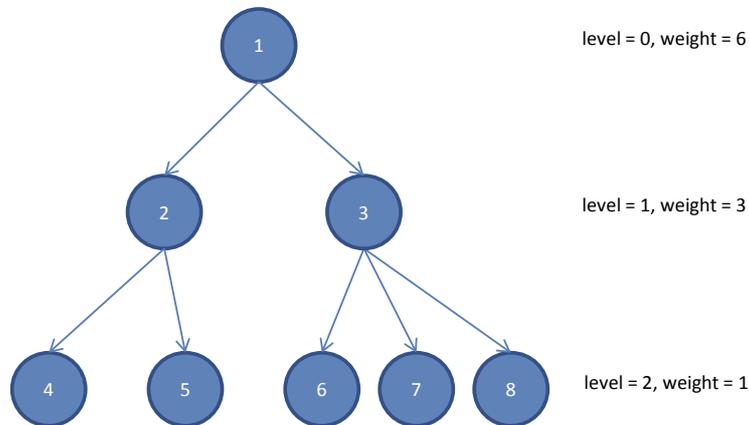


Figure 7.2: A simplified example of a DOM tree. Weights are computed using formula in Equation 7.2 with $c=1$.

To compute the distance of node 4 to node 8, we have to sum up all weights of the preorder successors of node 4 until node 8 is reached. This can be written as $d(4, 8) = w_5 + w_3 + w_6 + w_7 + w_8 = 1 + 3 + 1 + 1 + 1 = 7$.

Geometric Distance

In the rendered output of a web document, contents are presented to web users in a two dimensional plane. In particular, each content is placed within a bounding rectangle with specified *width* and *height* and further the rectangle has a *position* within the general browser plane. In the following, the bounding rectangle of a web content a will be represented by two points, referring to its two diagonal edges, e.g., [(left,top), (right,bottom)]. There are different ways to define a distance between rectangles. We can compute the minimal or the maximal distance between the rectangles, or even the distance between the center points of the rectangles. In this work, the minimal distance proved to be most suitable.

Definition 17 (Geometric Distance). Let R and S be two web contents and $r = [(r_x, r_y), (r_{x'}, r_{y'})]$ be the bounding rectangle of R , and respectively $s = [(s_x, s_y), (s_{x'}, s_{y'})]$ the bounding rectangle of S . The geometric distance of two contents R and S is computed as the minimum distance between the corresponding rectangles $r = [(r_x, r_y), (r_{x'}, r_{y'})]$ and $s = [(s_x, s_y), (s_{x'}, s_{y'})]$. In general, this distance can be defined as

$$\text{mindist}(R, S) = \sum_{i \in \{x, y\}} t_i^2$$

where

$$t_i = \begin{cases} r_i - s_i' & \text{if } r_i > s_i' \\ s_i - r_i' & \text{if } r_i' < s_i \\ 0 & \text{if otherwise.} \end{cases} \quad (7.3)$$

In Figure 7.3 we show an example with two rectangles and the minimum distance between the rectangles.

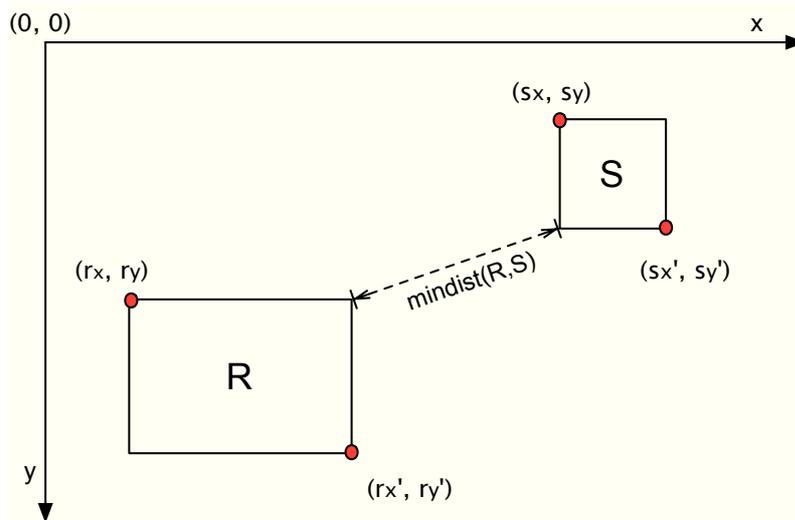


Figure 7.3: The minimal geometric distance between two rectangles R and S .

Semantic Distance

Web contents can be of different media types as text or images. To estimate the semantic distance (or dissimilarity) of web contents, it is necessary to determine a common representation of the semantics of different media types. Due to the fact that text is the most used media object in web documents and furthermore there already exist approved measures to compare the similarity of text contents (see Chapter 2.3), it is obvious to choose text as common representation. Images can be represented by

their alternative or title text, or – if such not available – by their URL string. Based on the textual representation, we can apply common metrics to compute a semantic distance of contents.

There is a variety of similarity metrics for texts in the literature. The simpler methods adopted from information retrieval rely on some kind of lexical matching between units of one text to the others (see Chapter 2.2.2). While these approaches can be successful to a certain degree, they fail to identify the *semantic* similarity of texts. For example the synonyms *Plane* and *Aircraft* have a high semantic correlation which cannot be detected without background knowledge.

To overcome these limitations a variety of more sophisticated metrics were proposed in the literature (see Chapter 2.3.4). Instead of estimating a word-to-word lexical matching degree, the words are mapped to corresponding concepts in a knowledge base and a concept-to-concept accordancy is computed. An overview of different semantic metrics which uses the WordNet taxonomy [MBF⁺90] as background knowledge is given in Patwardhan et al. [PBP03]. We decided to apply the similarity metric introduced by Lin [Lin98] to compute the accordancy of concepts due to its eligible performance and low computational costs. The definition of the similarity function is as follows:

$$\text{sim}_{\text{Lin}} = \frac{2 \cdot \text{IC}(\text{LCS})}{\text{IC}(\text{concept}_1) + \text{IC}(\text{concept}_2)} \quad (7.4)$$

where LCS is the least common subsumer of the two concepts in the WordNet taxonomy, and IC returns the information content [Res95] which is defined as:

$$\text{IC}(c) = -\log P(c)$$

and $P(c)$ is the probability of encountering an instance of concept c in a large corpus.

This metric computes the semantic similarity on the concept level and should now be extended to a text paragraph level. But before this can be done, some preprocessing is needed to bring the paragraph in the right format. We first apply a sentence detector followed by a word level tokenizer and finally do Part of Speech (POS) tagging to identify the word class for each word using the Open NLP Library [Ope10]. Afterwards, the terms are stemmed using the WordNet stemmer and mapped to suitable WordNet concepts if possible.

The text similarity can now be computed as proposed by Corley et al. [CM05]. Their approach is as follows: for each noun (verb) in the set of nouns (verbs) belonging to one text, they identify the noun (verb) in the other text with the highest semantic similarity (*maxSim*) according to the concept similarity sim_{Lin} . For the other word classes a lexical matching to their counterparts in the other text is performed. The

resulting similarity function between two texts T_1, T_2 is defined as follows:

$$sim(T_1, T_2)_{T_1} = \frac{\sum_{w_i \in T_1} maxSim(w_i, T_2) \cdot idf(w_i)}{\sum_{w_i \in T_1} idf(w_i)} \quad (7.5)$$

with $idf(w_i)$ identifying the inverse document frequency of the word w_i in a large corpus. Sparck-Jones introduced the idf in [SJ88] to estimate the *specificity* of a term. With the defined similarity metric in Equation 7.5, a directional similarity score is computed with respect to T_1 . The score from both directions can be combined into a bidirectional similarity as proposed in [CM05]:

$$sim(T_1, T_2) = sim(T_1, T_2)_{T_1} \cdot sim(T_1, T_2)_{T_2} \quad (7.6)$$

This similarity score has a value between 0 and 1, and thus can be easily converted to a normalized distance function:

$$dist_{sem}(T_1, T_2) = 1 - sim(T_1, T_2) \quad (7.7)$$

7.1.4 Clustering Methods

As we stated in Chapter 2.2.3, cluster analysis is a primary method for discovering groups of similar objects in various data repositories. In this section, we will briefly describe three basic clustering approaches that can be applied to group web contents. For more details please refer to the original articles.

Partitioning Clustering

Partitioning clustering is a classical approach to data clustering. The most popular representatives are k -means [Mac67] and k -medoid. The k -means algorithm starts by randomly partitioning the data points to k clusters and computing the initial cluster centroids. In the following step each data point is assigned to its next centroid and afterwards the centroids are recomputed. This step is repeated until the centroids do not change in a run. Since web contents are non numerical, the cluster centroid should be represented by a web content which has the minimum distance to all other contents of the cluster. This variant of partitioning clustering techniques is named k -medoid.

One critical issue with partitioning clustering is the parameter k , since the number of clusters varies among different web pages. Nevertheless, to estimate a good performing k , the algorithm can be run several times with a different k and clustering validity can be measured by computing the *sum of squared errors* or the *silhouette coefficient*.

In our evaluation studies, we used labeled web pages. As a consequence the number of blocks (or clusters) was implicitly given and did not need further estimation.

Partitioning clustering can be run with all proposed distance functions with no need to be adapted to the properties of the functions.

Hierarchical Agglomerative Clustering

Unlike the partitioning methods, hierarchical clustering approaches do not simply separate the data into clusters, but they generate an hierarchical representation of the data, which then can be used to derive a clustering. The clustering result is given in a dendrogram, a tree-like diagram which displays both, the cluster-subcluster relationships and the order in which the clusters were merged.

The basic algorithm to compute the dendrogram is as follows: starting with individual web contents as clusters, the inter-cluster distance is computed and the two closest clusters are merged to a single cluster. This merging step is successively repeated until only one cluster remains that comprises all clusters.

To compute the distance between sets of web contents, we apply the single link strategy, which corresponds to selecting the minimum distance of a web content from the first set to a content of the second set [Sib73]. Once the dendrogram is computed, the final clustering can be derived by cutting the dendrogram at a given level.

Density-based clustering

Density-based clustering assumes clusters to be data regions of a higher density that are bordered by data regions of lower density.

In DBSCAN [EKSX96] the minimal desired density is defined by two input parameters **Eps** and **MinPts**. Starting with an arbitrary data point, the point is checked on being a *core object*. A data point is called a core object, provided the number of data points within its **Eps**-environment is at least **MinPts**. If the actual data point is a core object, a new cluster is created with the actual point as its first member and the cluster is expanded by all points that are density-reachable by the actual data point. A data point a is *density-reachable* by a data point b if there is a sequence of connected core objects starting with a and ending with a data point in the **Eps** environment of b . Data points that are neither core objects nor density-reachable by any core object are marked as “noise”. The algorithm finishes when all data points are either clustered or marked as noise.

In experimental studies, we have recognized that the DOM-based distance spreads the web content clusters over different densities. Thus the combination of DBSCAN and DOM distance is not suitable because DBSCAN can not handle clusters on different density levels. In order to meet the special requirements of the DOM-based distance, a variation of DBSCAN [FSS⁺09] has been applied that estimates **EPS** and **MinPts** dynamically depending on the environmental density.

That is, the semantic and the geometric distance are combined with the common DBSCAN algorithm, while the DOM distance was applied together with the extended DBSCAN algorithm.

7.1.5 Experimental Studies

In the previous sections we have introduced different representations and distance measures for web contents, and also three different clustering approaches which could be applied to group the web contents. In this section, we focus on a detailed evaluation of the distance measures and the clustering approaches in order to estimate their eligibility for the WICE task.

Distance measures

In the first part of the evaluation, we will focus on properties of the proposed distance measures and show some empiric results. As example web page, the start page of the WIMS 2011 conference has been employed, which is depicted in Figure 7.5. Preliminary, we have extracted all text and content image nodes (too small images or images with a width-height-ratio exceeding a threshold were ignored) of the example page and they were put into a list ordered by their position in the HTML representation. Further, the web page in Figure 7.5 has been manually partitioned and the partitioning blocks are depicted as rectangles. The numbers in the rectangles indicate the id-ranges of the web contents.

Computational Time The curves in Figure 7.4 depict the computational times (in seconds) needed to compute a distance matrix with different distance measures depending on the number of web contents. While the matrices of the DOM-based distance and the geometric distance are computed reasonably fast, the computation of the semantic distance matrix is more time consuming. Instead of computing distance-matrices for different web pages and averaging over the computed times, we decided to show the results for only one document in order to show that the semantic distance does not only depend on the number of contents, but also on the length of the contents. On the other side, the DOM and the geometric distance are not affected by any node properties and thus produce smooth graphs.

Distance-Matrix Visualization The computed distance matrices of web contents from the WIMS page are represented in Figure 7.6 in a graphical way. The web contents are arranged in the same order from left to right in the columns and top down in the rows. Further, this order also corresponds to the order in which the contents appear in the HTML. Each pixel in an image represents the distance between two contents. A

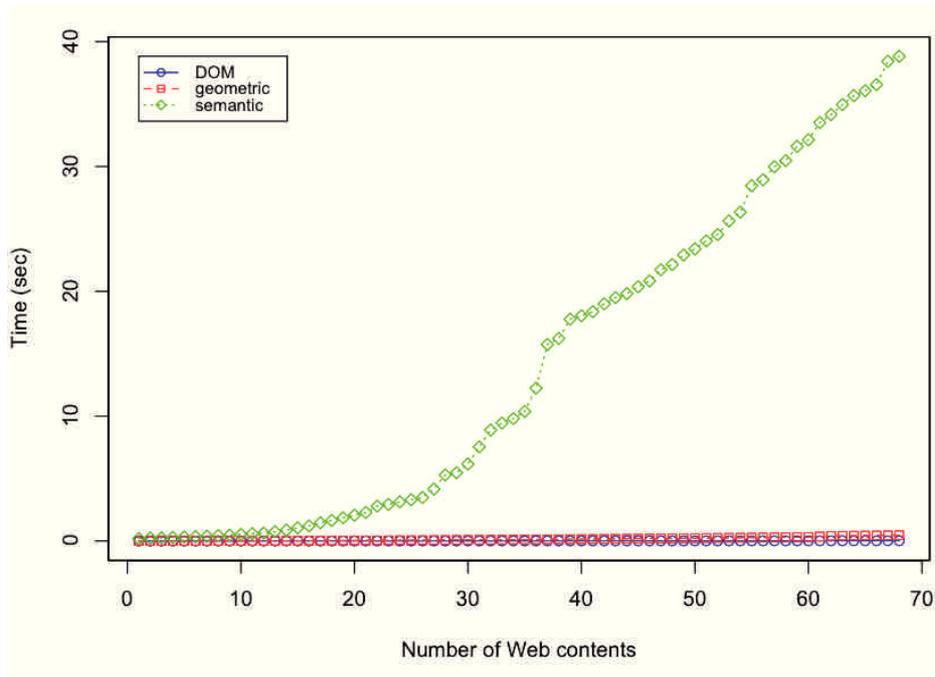


Figure 7.4: Computational times for the distance-matrices of different distance measures.



Figure 7.5: The start page of WIMS 2011 and possible blocks in which the page could be segmented. The numbers are the id's of the contents arranged in the order as they appear in HTML.

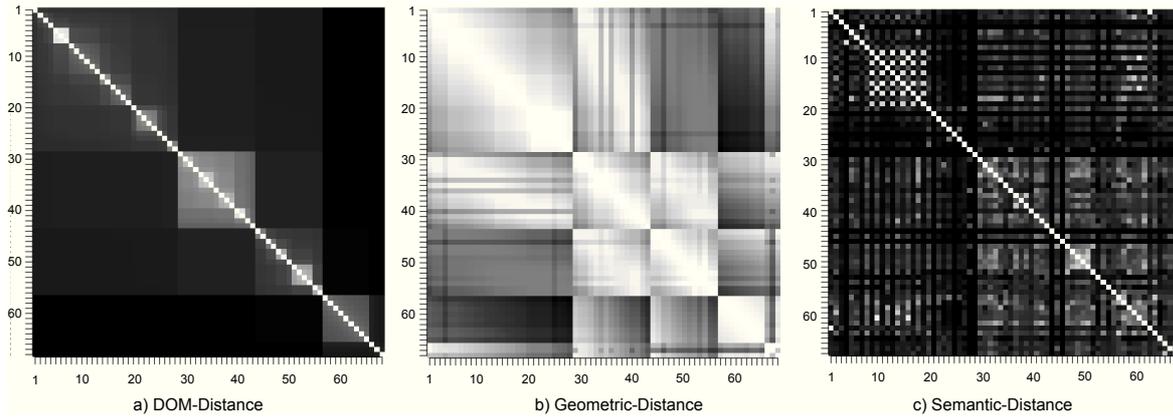


Figure 7.6: Visualized distance matrices for a) DOM distance, b) geometric distance and c) semantic distance.

bright pixel color corresponds to a smaller distance, while darker pixels represent high distances.

Since the range of the DOM distance can reach extraordinary high values (depending on the DOM tree depth), the matrix-values were transformed into logarithmic scale for visualization. The matrices of the geometric and the semantic distance were not processed.

Obviously, all matrices have a white diagonal because the distance of a content to itself is 0. To start with the DOM distance, the graphic shows clear rectangles of nearly constant brightness in the matrix diagonal which are further hierarchically organized: less brighter rectangles surround brighter ones. We can clearly recognize the rectangles which correspond to the blocks depicted in the WIMS start page. Similar patterns can also be found in the diagonal of the geometric distance matrix. However, there are many other bright regions in the geometric distance matrix, e.g., $(1 - 19, 29 - 43)$ because the corresponding blocks are next to each other in the browser output. The semantic distance matrix reflects only partially the contained structures. The rectangle $(8 - 19, 8 - 19)$ has a pattern similar to a chessboard. This is because each label content is followed by a date content; dates have a strong similarity to each other, while the contained labels are semantically unrelated to dates. It is also interesting, that contents distributed over the whole web page can be very similar, as e.g., $(1, 63)$. Comparing the corresponding labels, which is “Important Dates” in both cases, clarifies the reason.

Clustering Performance

In this part of the evaluation, we investigate the performance of different clustering approaches combined with the proposed distance metrics.

Dataset We collected a sample of 78 web documents from 8 different categories of the Yahoo! directory resulting in 23,819 web contents (text and content images) as data collection. The web contents of the documents were grouped manually by 3 volunteers. They got the instructions to create structurally and semantically coherent groups. Although this process lacks of objectivity, we suppose that the volunteers at least have a good idea of what coherent blocks are. The different groupings were combined to a final ground truth grouping by majority decision.

Parameter estimation The parameter k in k -means clustering can be set to the number of groups determined by the ground truth. Similarly, for single link clustering, the appropriate level at which the dendrogram is cut, can be chosen by considering the number of clusters to be produced. However, in DBSCAN the parameters `Eps` and `MinPts` had been set empirically. For this purpose, we run DBSCAN iteratively with different parameter configurations, and the best parameter settings in regard to the Rand statistics were applied.

Cluster Separation In order to investigate the separation value of the different clustering configurations, we applied the Dunn index [Dun74]. The Dunn index expresses, how well the web contents are separated in a given clustering result. This measure is defined as the ratio of the smallest distance between two web contents that are not in the same cluster to the highest distance of contents that are in the same cluster. The higher the Dunn index value, the better the clusters are separated in regard to a distance measure. Values over 1 indicate a good separation. We should further notice that the Dunn index does not express anything about the quality of the clusterings with respect to the ground truth dataset.

Table 1 summarizes the average Dunn index over the complete collection for the different clustering approaches with different distance measures.

	DOM-based	geometric	semantic
k-medians	1.03	0.34	1.05
single link	1.12	0.43	1.02
DBSCAN	-	0.24	1.18
DBSCAN (ext.)	0.35	-	-

Table 7.1: Average Dunn index for different Distance Measures and different Clustering Techniques

The first remarkable observation is that the semantic-based distance has Dunn index values over 1 which indicates that in the metric space derived by the semantics

of contents, the clusters are well separated.

The DOM-based distance gets similar results for k-medoid and single link clustering, however for the extended DBSCAN algorithm, the Dunn index is very low. That is exactly what we have expected: the extended DBSCAN finds clusters in different density levels. Therefore the intra-cluster distance on the lowest density level is expected to be much greater than the inter-cluster distance of two clusters on highest density levels.

The geometric distance measure reaches lower separation values with all clustering methods. This result can be explained with the simple example depicted in Figure 7.7.

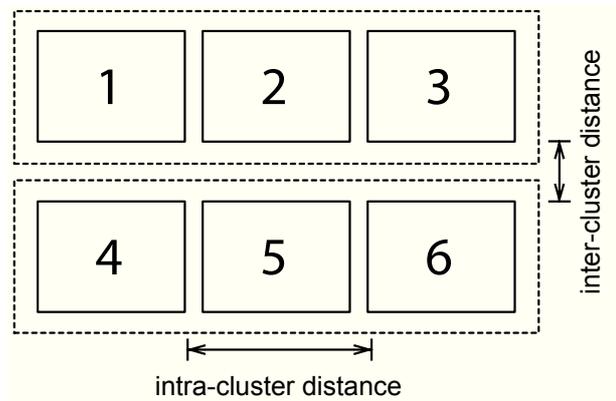


Figure 7.7: Partitioning six Web Contents into two Clusters using Geometric Distance

In this example, six web contents (here presented by their visual boxes) have to be grouped in two clusters. A good clustering is given by the dotted rectangles. However, in this case the inter-cluster distance is much smaller than the intra-cluster distance, which results in Dunn index lower than 1.

Rand statistics To compute the correspondence between computed and manually defined clusters (Ground Truth), the following contingency table shows four relations that can occur between pairs of contents:

	Same cluster	Different cluster
Same class	f_{11}	f_{10}
Different class	f_{01}	f_{00}

In this table, we distinguish between content pairs that are classified by volunteers to belong together or not, or respectively computed by a clustering method to belong together or not. Based on this values, we can compute the Rand statistic, which is

defined as follows [TSK05]:

$$\text{Rand statistic} = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}}.$$

As for the Dunn index, the higher the Rand statistic value, the better the clustering performance. Table 2 summarizes the average Rand statistic results for different clustering approaches with different distance measures.

	DOM-based	geometric	semantic
k-medoid	0.45	0.47	0.25
single link	0.52	0.41	0.24
DBSCAN	-	0.43	0.27
DBSCAN (ext.)	0.61	-	-

Table 7.2: Average Rand statistic for different Distance Measures and different Clustering Techniques

For the single link and extended DBSCAN method the DOM-based distance gets the highest accuracy while for k -medoid clustering the Rand statistic gets lower. This might be a consequence of the hierarchical distribution of the contents by the DOM-distance. This hierarchy is not considered by the k -medoid algorithm.

The geometric distance reaches the highest accuracy combined with the k -medoid method which might be justified by the property of the geometric distance to spread the web contents over a geometric space. In tests with single link and traditional DBSCAN the accuracy values were lower.

Finally, a last interesting observation is that the Rand statistics of the semantic distance were low independent of which clustering method was used. A possible explanation might be given by analyzing the distance-matrix visualization example from Figure 7.6. The semantic distance can assign high distance values to web contents that, e.g., in the browser representation are next to each other because they do not share common semantics. On the other hand, contents that are spread over the complete document can get very low distance values because their semantics are almost equal.

Discussion of Results

After the observations during the experimental studies presented above, we have decided to build a clustering-based approach to WICE which applies the DOM-based distance function and a density-based clustering. The reasons for our decision will be drafted shortly in the following.

The semantic distance was very promising since the web contents in a web content block should share the same semantics. However, the results observed in the visualized distance matrix already predicted that the groupings using the semantic distance would fail. The main disadvantage of the semantic distance is that contents that belong together (e.g., a title and the corresponding paragraph) can be composed of different terms that yield to different semantics. Similarly, contents that do not belong together but contain the same terms by chance have a high similarity. Another disadvantage of the semantic distance is the high time complexity.

In the visualized distance matrix of the geometric distance function we could find all page blocks as bright rectangles. However, we discover also bright regions in the distance matrix between web contents that do not belong to common blocks. This phenomenon is typical for the geometric function: in the browser output web contents can be distributed over different columns, each independent of the other; but visually the contents can have a small geometric distance. Further, the determination of the geometric properties of the web contents needs that the examined document is rendered which can be very time consuming.

The DOM distance has reached the best Rand index results and also the corresponding distance matrix was most promising since the bright rectangles which identify the page blocks could be found in the diagonal and also there were no other misleading bright regions in the matrix. However, the bright rectangles in the matrix are of different brightness levels that indicate that the page blocks have different density levels. Thus a clustering algorithm that is able to dynamically adapt to the different density levels would reach the best results.

7.2 Image Context Extraction by Web Content Clustering

In the last section, we investigated different web content representations and clustering methods to find out which fits best for the page segmentation task. Now these findings will be applied to approach the WICE task.

7.2.1 Properties of the DOM-based Distance

The DOM-based distance measure achieved the most promising results in the experimental studies to web content clustering. Since the different clusters are spread over different density levels, the extended DBSCAN algorithm that is constructed to deal with clusters of varying densities reached the best results. However, the DOM-based distance has another property that can be exploited in order to create a more efficient

clustering method. This property will be introduced by an example.

Recalling the formulation of the DOM-based distance in Definition 15, we can see that first all nodes of a DOM tree are put into a list which is ordered by traversing the tree in preorder. Each node gets then a weight and the DOM distance of two nodes a and b is just a sum of these weights starting with the successor of a and terminating at b . In this way, the DOM-based distance maps the web contents of a document into an one dimensional space. This mapping can be better clarified by an example.

Example 10 (One dimensional Mapping). *Consider the sample DOM-tree from Figure 7.2. The web contents in this tree are represented as leaf nodes. Using the DOM-based distance, we get the following distances for adjacent web contents: $d(\textcircled{4},\textcircled{5}) = 1$; $d(\textcircled{5},\textcircled{6}) = 4$; $d(\textcircled{6},\textcircled{7}) = 1$; and $d(\textcircled{7},\textcircled{8}) = 1$. Since the distance function is additive, we can compute other distances from these basic distances: $d(\textcircled{4},\textcircled{6}) = d(\textcircled{4},\textcircled{5}) + d(\textcircled{5},\textcircled{6}) = 5$, etc. In this way, the web contents are mapped into an one-dimensional space. If the origin is set to be at web content $\textcircled{4}$, then the distribution shown in Figure 7.8 arises.*

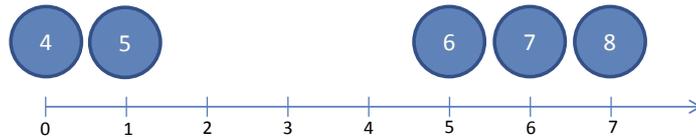


Figure 7.8: Distribution of Web Contents over an one-dimensional Space

From this new perspective, the partitioning of web contents to clusters is reduced to finding separators in a one-dimensional space. At the same time, the WICE is reduced to a simpler problem that will be formulated next.

7.2.2 A new Formulation of the WICE Problem

Given an image I in a Web document d , Web Image Context Extraction (WICE) denotes the process of determining the textual contents t_i of document d that are associated with the image I . The proposed DOM distance maps the basic content units of a web page to a 1D space. By setting cuts at appropriate positions in this space, contents are partitioned (or clustered) to content blocks. The image I can now be associated with the textual contents t_i of the block, I belongs to. Thus the WICE problem is reduced to clustering in 1D space, i.e., to estimating suitable positions in this space to separate contents.

7.2.3 1D-Clustering based on Distance Threshold

The idea to a clustering method for 1D data points will be motivated by the example DOM tree from Figure 7.2. Consider the simple one-dimensional optimization problem in Figure 7.8: we want to find a good clustering for the five data points into two clusters so that the variance of the distances between each pair of adjacent points in the same cluster is minimized. It is not hard to understand that by cutting at the largest distance between a pair of adjacent points, we will find a good solution to the clustering problem. Actually, if we are able to define a threshold that the distance of adjacent contents should not exceed, the clustering can be done as described in Algorithm 9, even without knowing the target number of clusters.

Algorithm 6: 1D-Clustering by Thresholding

Input: Sequence of web contents $S = (s_1, \dots, s_n)$, threshold t

Result: Set of computed clusters C

```

1  $c = \text{newCluster}(s_1)$ ;
2 for  $i = 2$  to  $n$  do
3   if  $d(s_{i-1}, s_i) > t$  then
4      $C.\text{add}(c)$ ;
5      $c = \text{newCluster}(s_i)$ ;
6   else
7      $c.\text{add}(s_i)$ ;
8   end
9 end

```

The algorithm starts by initializing a new cluster c with the first element s_1 . Then a loop iterates over the sequence S , in which the elements are ordered by their original position in the documents source, and computes the distance between every pair of adjacent contents. If the computed distance is greater than a predefined threshold t , the actual cluster c is completed and put in the set of clusters C and c is initialized again as a new cluster with content s_i as the only element. Otherwise, content s_i expands the actual cluster c .

7.2.4 Threshold estimation

A static threshold value t could be computed by averaging over all distances of adjacent content pairs:

$$t = \frac{1}{n-1} \sum_{k=2}^n d(s_{k-1}, s_k).$$

This baseline threshold might work well for web documents that consist of content clusters with similar density. However, since web contents are usually distributed over

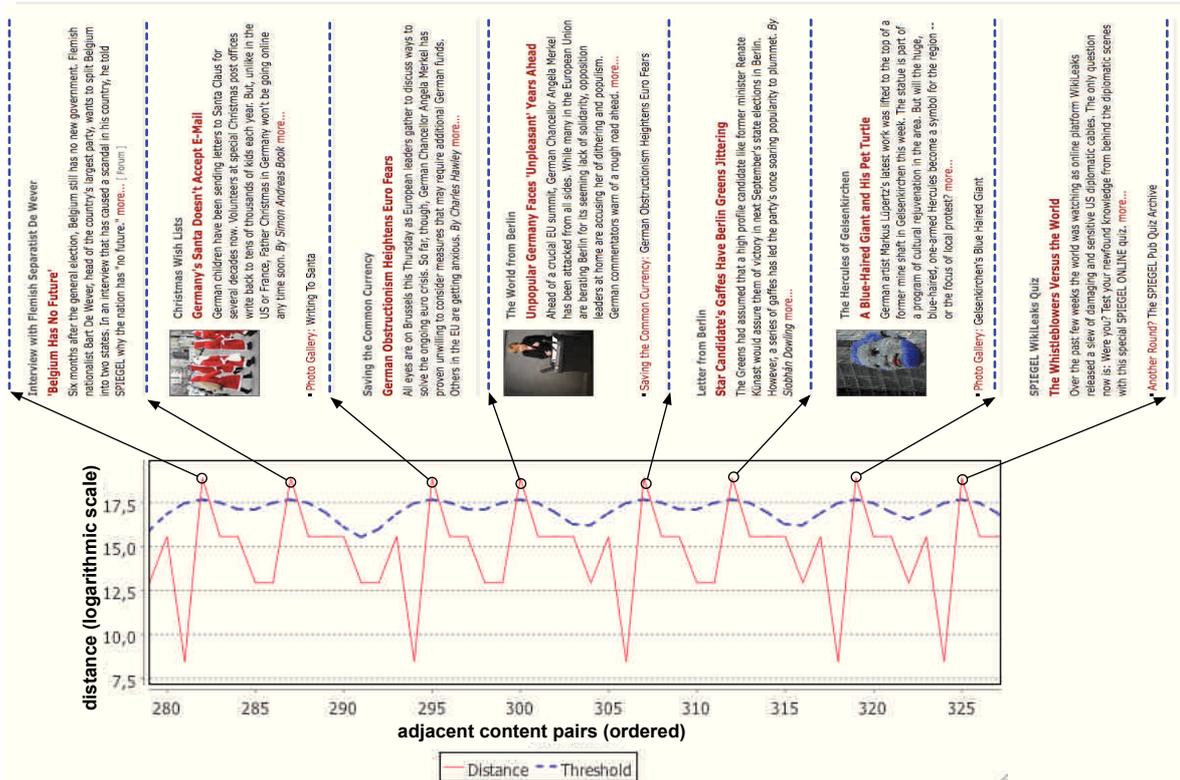


Figure 7.9: Distances of adjacent contents and the threshold values computed by weighted averaging with gaussian. The peaks at which the distance exceeds the threshold point to the corresponding separators in the browser output of the document.

different tree levels, the cluster density can significantly differ among different clusters. Thus the threshold has to be more adaptive to distances in the environment.

To meet these requirements, we propose to use a gaussian weighted threshold function $t : \mathbb{N} \rightarrow \mathbb{R}$:

$$t(k) = \frac{1}{\sum_{i=2}^n G(i, k, \sigma)} \sum_{i=2}^n G(i, k, \sigma) d(s_{k-1}, s_k),$$

with the gaussian function $G(x, \mu, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$. The remaining parameter σ^2 is the variance (the measure of the width of the gaussian peak) and has to be considered empirically.

7.2.5 Visual Representation of Clustering

To give a more intuitive description to the proposed algorithm, we visualize its main components. The solid-line curve in Figure 7.9 refers to the distances of adjacent web contents, while the values on the x -axis correspond to the index of the contents in the contents sequence (e.g., x -value 280 means the distance of adjacent contents 280

and 281). The dashed-line curve in the same plot is the gaussian smoothed version of the solid-line curve, and corresponds to the threshold t . For each peak of red (distances) function exceeding the blue (threshold) function, we have drawn a circle at this function value and pointed with an arrow to the corresponding position in the browser representation of the document. This example shows illustratively the quality of our method, since all blocks were properly recognized.

7.2.6 Evaluation

The clustering-based approach to WICE (CLUS) was also evaluated using the evaluation framework that was introduced in Chapter 5. Some significant results of the evaluation will be presented in the following. The complete results are attached in the Appendix.

Running Time

As for the other WICE methods, we have measured the average time that CLUS needed to process a web document. The result is depicted in Figure 7.10 together with the results of all other extraction methods for comparison.

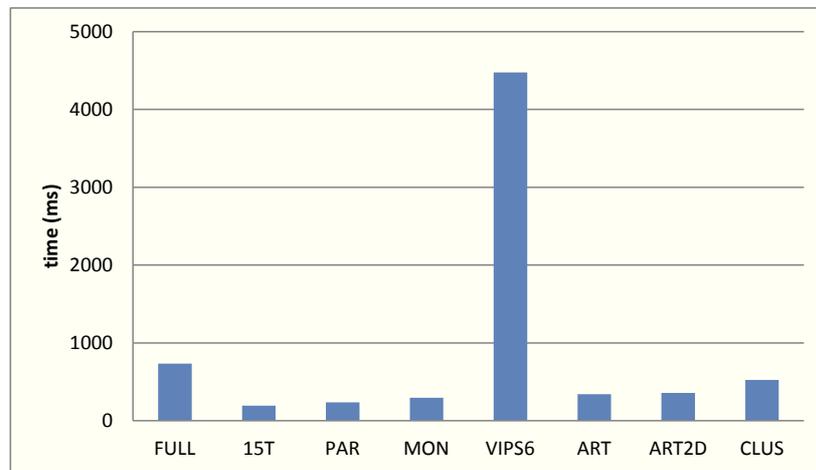


Figure 7.10: Average Processing Times over all Collections

The running time for the CLUS method includes the time needed to create a special index for the DOM-based distance, the time needed for partitioning the web page and also the time needed for context mapping. We can observe that although the CLUS method is more time consuming than other extraction methods, it is still faster than the full text baseline, which – as we mentioned earlier – has been applied in many real applications. Thus we conclude that the CLUS method can also be applied on real applications that work with web image context.

Parameter estimation

The clustering-based method to context extraction has one open parameter σ^2 that has to be specified. σ^2 is part of the gaussian smoothing kernel and determines its variance. In order to avoid overfitting, the parameter was trained iteratively on a smaller subset of our test data consisting of five documents of each collection. The maximal average F -score was reached when σ^2 was set to 1.25.

Extraction Quality

The quality of the CLUS algorithms has been measured by computing precision, recall, F -score and F -score standard deviation on the 12 different test collections. In the following, we will present and discuss some significant results obtained on four different test collections.

Golem Collection The “Golem” Collection contains simple structured documents. Especially, the web images in this collection are almost always placed using the same design pattern – Headline, Image, Longer Text. The results that were computed for this collection are depicted in Figure 7.11. As we can see in this chart, the CLUS method

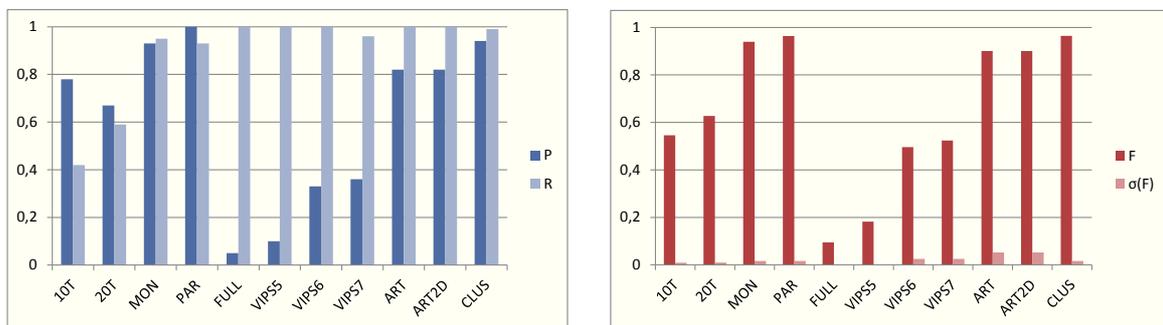


Figure 7.11: Golem Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

reaches very high quality results with an F -score over 0.96 that is even better than the articles-based approach and similar to the results of the paragraph extractor which performed almost perfect on this collection. One of the main reasons for this success is the well-structured HTML in which most articles are on a similar tree level. The articles are well separated by peaks of high adjacent-content distances that are easily detected by the proposed CLUS algorithm. The standard deviation value of the CLUS method is small and indicates that the method performs stable on this collection.

Globe&Mail-Collection The “Globe&Mail” Collection seems also to be very well structured and poor in design variations. Similar as for the Golem Collection, we

can find two (or three) different design patterns for articles with images within the document and thus the expectations concerning the extraction quality were the same as for the Golem Collection. However, the results depicted in Figure 7.12 are not fully complied with our prospects. We can observe that while the recall values are relatively

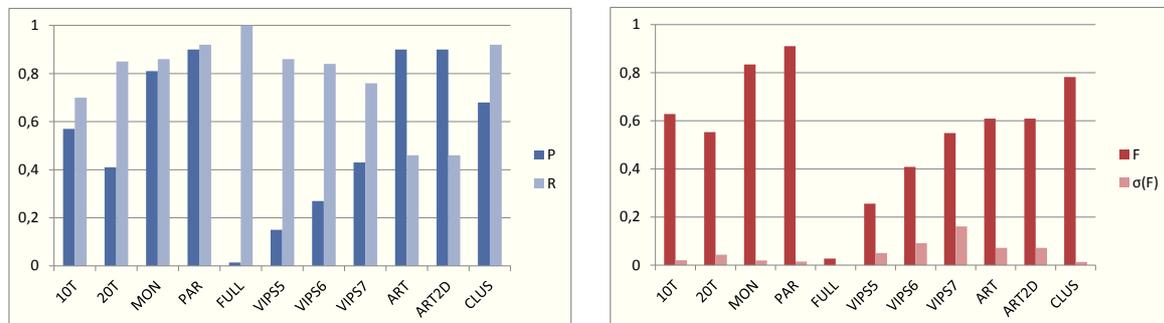


Figure 7.12: The Globe&Mail Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

high, the average precision over this collection reaches a value less than 0.7. Such a result is a sign for a too broad segment, meaning that there is text that does not belong to an image but was additionally included in the extracted context. A deeper analysis of the clustering process by a detailed plotting of the DOM-based distances of adjacent web contents and the gaussian smoothed threshold function in Figure 7.13 shows the reason:

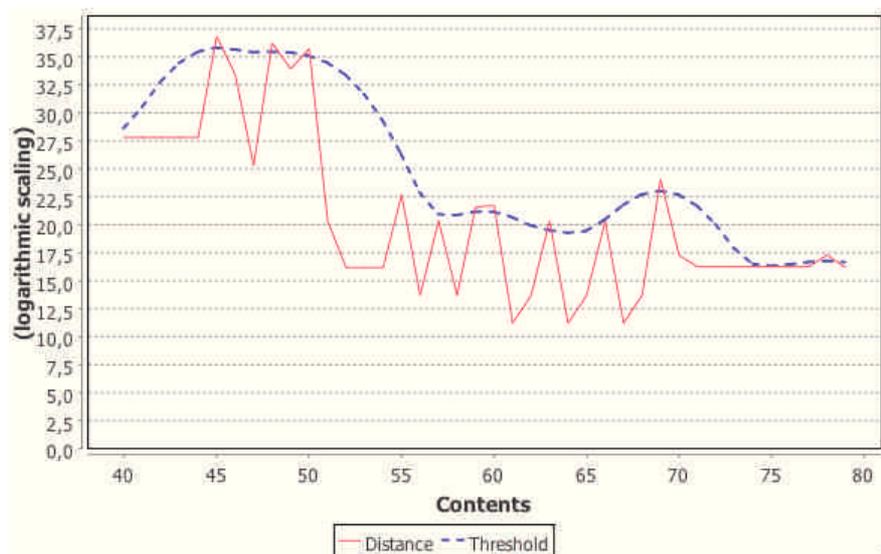


Figure 7.13: Plotting of adjacent Web Content Distances (solid-line) and their Gaussian-Smoothed (dashed-line)

the peaks at 63 and 66 show two positions at which a cut should be set. However, while the threshold value at 63 is lower than the distance – that yields to a cut at 83

– we observe at 66 that the threshold is a little higher than the distance, which results in a false decision not to separate at this position. This situation occurs because the distance values after 66 get significantly higher which affects the threshold. We could fix the problem by adjusting the σ^2 parameter (setting to a smaller value), but then other constructs in other collections would be destroyed.

Anyway, compared to the other extraction algorithms, CLUS seems to perform very well. Only the rule-based algorithms, Monash and Paragraph, are slightly better. Also the standard deviation of the F-score remains low that indicated the repeatability of the clustering-based method.

BBC Collection The “BBC” Collection is a more complex collection with many different image use-cases: web images are used in articles, in video previews and in commercials. Although some image usage constructs are based on the main template of the collection, there are many variations which make the context extraction difficult. Nevertheless, we want to present the obtained evaluation results (see Figure 7.14) and to discuss them in detail. Our main observation is that again while the recall-value of

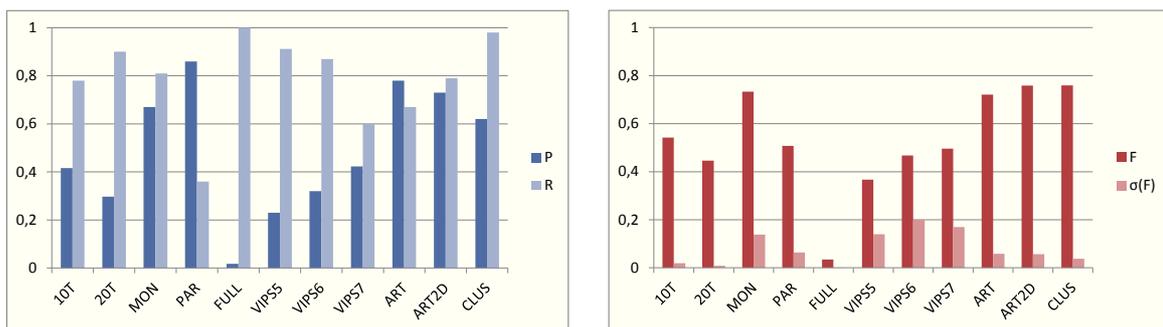


Figure 7.14: BBC Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

the CLUS method is high, the precision is significantly lower. The detailed view on a clustering example acknowledges our assumptions: the page segments are too broad. The reason is quite the same as for the Globe&Mail Collection, but this time the problem can not be fixed just by changing the σ^2 parameter: the images and contexts are spread over various tree levels and make the page segmentation very difficult. The quality variance can also be detected in the standard deviation value of the F-score which is a little bit higher this time.

Manual Collection As we mentioned earlier, the “Manual” collection consists of various documents from different domains. We can not really describe the structure of the included documents because they are all different. Nevertheless, it would be very interesting to see how the CLUS method performs on a such mixed collection.

The evaluation results are summarized in Figure 7.15. Although the average F -score

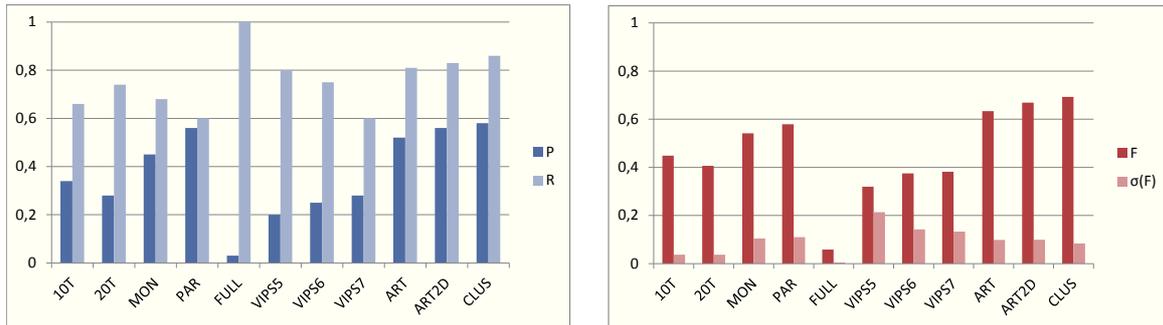
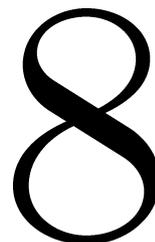


Figure 7.15: Manual-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

of the CLUS method is slightly under 0.7, it is the best result compared to the other methods. Again the average precision of CLUS is lower than its recall that indicates too broad segments. However, by setting the σ^2 parameter to lower values, we are able to get higher precision values, but at the expense of falling recall values; in overall, we get slightly the same F -scores. The standard deviation values of the F -score are again relatively small and acknowledge the stability of the CLUS results on this collection.

7.3 Summary

In this chapter, we have first proposed to approach the problem of page segmentation by a cluster analysis method, since both problems are very similar to each other. In order to use a clustering for web contents, it is necessary to decide how the web contents are represented and how their (dis)similarity is expressed. We have introduced three kinds of representations, namely DOM-based, geometric, and semantical, and based on these representations we defined three distance measures for web contents. Then these distance measures were combined with three different clustering approaches (k-means, single link and DBSCAN), and evaluated with the aim to find the best combination. The most promising distance measure was the DOM-based distance combined with an extended DBSCAN algorithm, that is able to handle web content clusters on different density levels. Therefore, in the second part of this chapter, we have introduced a simplified page segmentation algorithm that exploits the properties of the DOM-based extraction method. Finally, this WICE method has been evaluated and compared to other methods using the evaluation framework proposed in Chapter 5.



CONCLUSION AND FUTURE WORK

This chapter presents the conclusion of this thesis. Section 8.1 highlights the contributions of this work and discusses the obtained results. Finally, some interesting directions for future research are illustrated in Section 8.2.

8.1 Summary and Conclusions

Compared to conventional images in albums, web images are preferred, because they come along with valuable textual descriptions on their hosting web pages. However, web pages are usually cluttered with many topics and the right context of the images has to be detected. In this thesis, we have dealt with different topics related to this problem, which will be summarized in the following.

WICE has been rarely addressed directly in research work so far and to our knowledge there are no surveys to this research field. We filled this gap by giving an extensive overview of existing WICE algorithms applied in different related work. As an additional part of this overview, we have further included the various approaches to WICE evaluation, reaching from involving human experts over particular application specific tests to general purposed evaluations, each with different techniques to measure the extraction quality and to compare the different WICE approaches. Nonetheless, all of the discussed approaches had drawbacks. The major aspects were: the lack of subjectivity, the inability to automate the evaluation process, and hence, the impossibility to test a WICE method extensively on a representative web document collection, as well as the absence of suitable evaluation measures that go beyond exact matching of extracted and ground truth image context.

Motivated by the idea of improving the mentioned drawbacks, we contributed with

a new evaluation framework for WICE methods. In particular, we presented a new method to generate huge test data collections with only little manual effort. Web documents relying on the same template were crawled, while extraction rules were defined manually for the template: In this way, thousands of web documents could be processed automatically. Furthermore, in order to estimate the quality of the extraction methods, the WICE task has been reinterpreted as a special IR task and common IR metrics have been adapted. The first comparison of the evaluation results showed that the DOM-based wrappers (Monash and Paragraph Extractor) reached the best quality on almost all test collections. But while for the simple and well-structured document collections, the obtained results were almost perfect, the quality decreases with the increasing complexity of the web page structure. The window-based extractors (10/20-terms) range in the middle third for almost all collections. Since the web images are mostly either before the context or after the context, the N-term window often includes half of falsely assigned context terms. The VIPS-based extractor relies on a page segmentation algorithm, which usually selects a too broad partitioning and thus results in an almost perfect recall, but low precision. However, if the parameters are adjusted to gain a finer partitioning, most of the image blocks consist only of the image alone. Comparing the execution times, the VIPS algorithm needed up to twenty times longer than the other algorithms to process one single document, because the segmentation step renders the document in a browser application, which is very time consuming. The other algorithms are even faster than the baseline full-text extractor and thus applicable in real scenarios. Although all methods showed some drawbacks, every method proved better than the full-text baseline, which means that a performance gain over the full-text context is expected no matter which method is applied. Typical problems for the context extraction were images with no context at all, contexts that additionally were partitioned by non-context parts, and tables as layout elements, where the relationship between different lines can not be derived in the source code.

Our next contribution was a new WICE method based on article detection. Thereby the extraction process was divided into two tasks: article detection and image-to-article mapping. We further extended the mapping step by concepts allowing to handle tables used as layout elements. The proposed algorithm achieved a very high extraction quality on almost all test collections. However, there are documents like the Wikipedia-Collection, where the article detection rules fail and therefore the overall quality suffers. The extended table handling algorithm slightly improves the extraction quality on some collections, where tables are used. However, surprisingly most of the documents of the news sites today do not apply tables but div elements to place the web contents. Of course, for these collections we do not see any increase of quality of the extended approach.

Most of the proposed WICE methods are based on simple heuristic rules, which

in general are incapable to cover the variety of web document designs on the Web. Therefore, we take up the idea of applying page segmentation as a preprocessing step to WICE. Since existing segmentation algorithms are unable to separate the documents into appropriate image context blocks without modifications, we have carried out own investigations to find the best parameters for a clustering-based page segmentation, where the atomic web contents of a document are grouped into clusters. Three different representations for web contents, namely DOM-based, geometric and semantic, were combined with a partitional, a hierarchical and a density-based clustering. As a result, the DOM-based distance combined with a density-based clustering, which is able to find clusters on different density levels, reached the best results, and we finally introduced a new clustering-based WICE method exploiting the new findings. Since the DOM-based distance maps the web contents to a one-dimensional space, we proposed a simple clustering algorithm, which similar as the winning density-based clustering finds clusters on different density levels. In order to use the web content clustering approach as a WICE method, we just needed to associate the images with the text contents of their cluster. Our clustering-based WICE achieved high quality on all collections, and especially on the most variable “Manual” collection, it reached the best results.

8.2 Future Work

In the context of this thesis, we have focused on the extraction of web image context from web pages, and we presented methods that perform this task with acceptable quality. However, there are different documents, like PDF- or Open-Office-files that similarly include images with corresponding context and it would be interesting to see, if the proposed algorithms could also be applied on such documents.

Another direction of future research are applications that make use of the web image context. First of all, we would need further investigations of the quality of web image context, since even if we are to extract the image context perfectly, this is not a guarantee that the image context shares semantics with the image. Based on these investigations, we could build a classifier that is able to roughly estimate the quality of the image context relying on certain context and image properties, like image dimensions or context length. Then, there are different applications which can benefit from web image context. At the Database and Information Systems Institute, we are following already an approach to build a visual dictionary from annotated images. The entries of this dictionary are low-level image features, which are mapped to high-level semantic concepts. Instead of using annotated images that are associated with a high manual effort, we could apply high-quality image-context pairs.

REFERENCES

- [ABC⁺98] Vidur Apparao, Steve Byrne, Mike Champion, Scott Isaacs, Ian Jacobs, Arnaud Le Hors, Gavin Nicol, Jonathan Robie, Robert Sutor, Chris Wilson, and Lauren Wood. Document Object Model (DOM) Level 1 Specification. <http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001/>, 1998.
- [AC08] Sadet Alci and Stefan Conrad. 2-DOM: A 2-Dimensional Object Model towards Web Image Annotation. In *Third International Workshop on Semantic Media Adaptation and Personalization (SMAP2008)*, pages 3–8, Prague, Czech Republic, 2008. IEEE Computer Society.
- [AC10] Sadet Alci and Stefan Conrad. Measuring Performance of Web Image Context Extraction. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining, MDMKDD '10*, pages 8:1–8:8, New York, NY, USA, 2010. ACM.
- [AC11a] Sadet Alci and Stefan Conrad. A Clustering-based Approach to Web Image Context Extraction. In *The Third International Conferences on Advances in Multimedia (MMEDIA 2011)*, April 2011.
- [AC11b] Sadet Alci and Stefan Conrad. Page Segmentation by Web Content Clustering. In *International Conference on Web Intelligence, Mining and Semantics (WIMS11)*, May 2011.
- [Alc07] Sadet Alci. Improvement of Content-based Image Retrieval Quality using Automatic Annotation, April 2007. Master Thesis.
- [Alc08] Sadet Alci. Verbesserung der Suche nach Webbildern durch automatische Annotationen. In *Workshop zu Grundlagen von Datenbanken*, pages 91–95, 2008.
- [Ale11] Alexa. The web information company. <http://www.alexa.com>, 2011.
- [AM10] Murray Altheim and Shane McCarron. XHTML 1.1 - Module-based XHTML - Second Edition. <http://www.w3.org/TR/xhtml11/>, 2010.

- [AY00] Yuksel Alp Aslandogan and Clement T. Yu. Diogenes: A web search agent for person images. In *MULTIMEDIA '00: Proceedings of the eighth ACM international conference on Multimedia*, pages 481–482, New York, NY, USA, 2000. ACM.
- [Bal06] Shumeet Baluja. Browsing on small screens: recasting web-page segmentation into an efficient machine learning framework. In *Proceedings of the 15th international conference on World Wide Web, WWW '06*, pages 33–42, New York, NY, USA, 2006. ACM.
- [BYN99] Ricardo A. Baeza Yates and Berthier R. Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.
- [Cai11] Deng Cai. Download Site of the DEMO of VIPS Algorithm. <http://www.zjucadcg.cn/dengcai/VIPS/VIPS.html>, 2011.
- [CCS+04] Tatiana Almeida Souza Coelho, Pável Pereira Calado, Lamarque Vieira Souza, Berthier Ribeiro-Neto, and Richard Muntz. Image Retrieval Using Multiple Evidence Ranking. *IEEE Transactions on Knowledge and Data Engineering*, 16(4):408–417, 2004.
- [Che03] Yu Chen. Detecting Web Page Structure for Adaptive Viewing on Small Form Factor Devices. In *In Intl. World Wide Web Conf. (WWW)*, pages 225–233. ACM Press, 2003.
- [CHL+04] Deng Cai, Xiaofei He, Zhiwei Li, Wei-Ying Ma, and Ji-Rong Wen. Hierarchical clustering of WWW image search results using visual, textual and link information. In *MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia*, pages 952–959, New York, USA, 2004.
- [CKP08] Deepayan Chakrabarti, Ravi Kumar, and Kunal Punera. A graph-theoretic approach to webpage segmentation. In *Proceeding of the 17th international conference on World Wide Web, WWW '08*, pages 377–386, New York, NY, USA, 2008.
- [CM05] Courtney Corley and Rada Mihalcea. Measuring the Semantic Similarity of Texts. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment, EMSEE '05*, pages 13–18, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

- [CNN11] CNN. The official web site of Cable News Network . <http://www.cnn.com>, 2011. Access on March 15th, 2011.
- [CYWM03] Deng Cai, Shipeng Yu, Ji-Rong Wen, and Wei-Ying Ma. VIPS: a Vision-based Page Segmentation Algorithm. Technical report, Microsoft Research (MSR-TR-2003-79), 2003.
- [DJLW08] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40:5:1–5:60, May 2008.
- [DMPG05] Sandip Debnath, Prasenjit Mitra, Nirmal Pal, and C. Lee Giles. Automatic Identification of Informative Sections of Web Pages. *IEEE Trans. on Knowl. and Data Eng.*, 17:1233–1246, September 2005.
- [Dun74] J. C. Dunn. Well separated clusters and optimal fuzzy-partitions. *Journal of Cybernetics*, 4:95–104, 1974.
- [EK SX96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proc. of 2nd International Conference on Knowledge Discovery in Databases*, pages 226–231, 1996.
- [FdMRN⁺07] David Fernandes, Edleno S. de Moura, Berthier Ribeiro-Neto, Altigran S. da Silva, and Marcos André Gonçalves. Computing block importance for searching on web sites. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, CIKM '07*, pages 165–174, New York, NY, USA, 2007. ACM.
- [FHB09] Fariza Fauzi, Jer-Lang Hong, and Mohammed Belkhatir. Webpage segmentation for extracting images and their surrounding contextual information. In *ACM Multimedia*, pages 649–652, 2009.
- [FSA96] Charles Frankel, Michael J Swain, and Vassilis Athitsos. WebSeer: An Image Search Engine for the World Wide Web. Technical report, University of Chicago, Chicago, IL, USA, 1996.
- [FSC04] Huamin Feng, Rui Shi, and Tat-Seng Chua. A bootstrapping framework for annotating and retrieving WWW images. In *Proceedings of the 12th annual ACM international conference on Multimedia, MULTIMEDIA '04*, pages 960–967, New York, NY, USA, 2004. ACM.

- [FSS⁺09] Ahmed M. Fahim, Gunter Saake, Abdel-Badeeh M. Salem, Fawzy A. Torkey, and Mohamed A. Ramadan. An Enhanced Density Based Spatial Clustering of Applications with Noise. In *DMIN*, pages 517–523, 2009.
- [Gec11] Gecko. The Gecko Engine provided by the Mozilla Developer Network. <https://developer.mozilla.org/en/gecko>, 2011.
- [Goo11] GoogleImages. The Image Search Engine provided by Google. <http://www.google.com/imghp>, 2011.
- [GR95] Venkat N. Gudivada and Vijay V. Raghavan. Content-based image retrieval systems. *Computer*, 28:18–22, 1995.
- [HCW⁺07] Xiaofei He, Deng Cai, Ji-Rong Wen, Wei-Ying Ma, and Hong-Jiang Zhang. Clustering and searching www images using link and page layout analysis. *ACM Trans. Multimedia Comput. Commun. Appl.*, 3(2):10, 2007.
- [Hed11] Jonathan Hedley. jsoup: Java HTML Parser. <http://jsoup.org/>, 2011.
- [HHMS07] Gen Hattori, Keiichiro Hoashi, Kazunori Matsumoto, and Fumiaki Sugaya. Robust web page segmentation for mobile terminal using content-distances and page layout information. In *Proceedings of the 16th international conference on World Wide Web, WWW '07*, pages 361–370, New York, NY, USA, 2007. ACM.
- [HHW⁺00] Arnaud Le Hors, Philippe Le Hégarret, Lauren Wood, Gavin Nicol, Jonathan Robie, Mike Champion, ArborText, and Steve Byrne. Document Object Model (DOM) Level 2 Core Specification. <http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113/>, 2000.
- [HHW⁺04] Arnaud Le Hors, Philippe Le Hégarret, Lauren Wood, Gavin Nicol, Jonathan Robie Mike Champion, and Steve Byrne. Document Object Model (DOM) Level 3 Core Specification. <http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/>, 2004.
- [Hir75] D. S. Hirschberg. A linear space algorithm for computing maximal common subsequences. *Commun. ACM*, 18:341–343, June 1975.
- [Hug94] Kevin Hughes. Entering the world-wide web: a guide to cyberspace. *SIGWEB Newsl.*, 3:4–8, March 1994.

- [HYR08] Fang He, Nenghai Yu, and Xiaoguang Rui. Multi-progressive model for web image annotation. In *Proceeding of the 16th ACM international conference on Multimedia*, MM '08, pages 825–828, New York, NY, USA, 2008. ACM.
- [Jav11] JavaXPCOM. The Java based Version of o XPCOM. <https://developer.mozilla.org/en/javaxpcom>, 2011.
- [JC97] Jay J. Jiang and David W. Conrath. Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. *Proceeding of International Conference Research on Computational Linguistics*, pages 24–26, 1997.
- [JHTS08] Hai Jin, Ruhan He, Wenbing Tao, and Aobing Sun. VAST: Automatically Combining Keywords and Visual Features for Web Image Retrieval. In *10th International Conference on Advanced Communication Technology, 2008. ICACT 2008*, volume 3, pages 2188–2193, Feb 2008.
- [JKWA05] Yohan Jin, Latifur Khan, Lei Wang, and Mamoun Awad. Image annotations by combining multiple evidence & WordNet. In *Proceedings of the 13th annual ACM international conference on Multimedia*, MULTIMEDIA '05, pages 706–715, New York, NY, USA, 2005. ACM.
- [Kah97] B. Kahle. Archiving the internet. *Scientific American*, March 1997.
- [KHC05] Hung-Yu Kao, Jan-Ming Ho, and Ming-Syan Chen. WISDOM: Web Intrapage Informative Structure Mining Based on Document Object Model. *IEEE Trans. on Knowl. and Data Eng.*, 17:614–627, May 2005.
- [KN08] Christian Kohlschütter and Wolfgang Nejdl. A densitometric approach to web page segmentation. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 1173–1182, New York, NY, USA, 2008. ACM.
- [KZB04] M. L. Kherfi, D. Ziou, and A. Bernardi. Image Retrieval from the World Wide Web: Issues, Techniques, and Systems. *ACM Comput. Surv.*, 36:35–67, March 2004.
- [LBL98] H. Lie, B. Boss, and C. Lilley. The text/css Media Type (RFC 2318). <ftp://ftp.rfc-editor.org/in-notes/rfc2318.txt>, 1998.
- [LH02] Shian-Hua Lin and Jan-Ming Ho. Discovering Informative Content Blocks from Web Documents. In *In Proceedings of ACM SIGKDD'02*, pages 588–593, 2002.

- [Lin98] Dekang Lin. An Information-Theoretic Definition of Similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning, ICML '98*, pages 296–304, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.
- [Liu07] Bing Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*. Data-Centric Systems and Applications. Springer, 2007.
- [LLhPH02] Xiaoli Li, Bing Liu, Tong heng Phang, and Minqing Hu. Using Micro Information Units for Internet Search. In *Proc. of the ACM 11th International Conf. on Information and Knowledge Management (CIKM-02)*, 2002.
- [LLM⁺06] Jing Liu, Mingjing Li, Wei-Ying Ma, Qingshan Liu, and Hanqing Lu. An adaptive graph model for automatic image annotation. In *Proceedings of the 8th ACM international workshop on Multimedia information retrieval, MIR '06*, pages 61–70, New York, NY, USA, 2006. ACM.
- [Mac67] J. B. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.
- [MBF⁺90] George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. WordNet: An on-line lexical database. *International Journal of Lexicography*, 3:235–244, 1990.
- [Nik11] Vladimir Nikic. The Homepage of the HTMLCleaner Java API. <http://htmlcleaner.sourceforge.net/>, 2011.
- [OBMCP00] Michael Ortega-Binderberger, S. Mehrotra, K. Chakrabarti, and K. Porakaew. WebMARS: A Multimedia Search Engine For The World Wide Web. In *In Proceedings of the SPIE Electronic Imaging 2000: Internet Imaging*, San Jose, CA, 2000.
- [Ope10] OpenSource. Opennlp: <http://opennlp.sourceforge.net/>, 2010.
- [Osw06] Derrick Oswald. The Homepage of the HTMLParser Java API. <http://htmlparser.sourceforge.net/>, 2006.
- [Pan03] Lexin Pan. Image8: an image search engine for the internet. Honours year project report, School of Computing, National University of Singapore, Apr. 2003.

- [PBP03] Siddharth Patwardhan, Satanjeev Banerjee, and Ted Pedersen. Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the 4th international conference on Computational linguistics and intelligent text processing, CICLing'03*, pages 241–257, Berlin, Heidelberg, 2003. Springer-Verlag.
- [Por97] M. F. Porter. *An algorithm for suffix stripping*, pages 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.
- [Res95] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 1*, pages 448–453, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.
- [RHJ09] Dave Ragett, Arnaud Le Hors, and Ian Jacobs. HTML 4.01 Specification. <http://www.w3.org/TR/html401/>, 2009.
- [RLL⁺07] Xiaoguang Rui, Mingjing Li, Zhiwei Li, Wei-Ying Ma, and Nenghai Yu. Bipartite graph reinforcement model for web image annotation. In *Proceedings of the 15th international conference on Multimedia, MULTIMEDIA '07*, pages 585–594, New York, NY, USA, 2007. ACM.
- [RYWL07] Xiaoguang Rui, Nenghai Yu, Taifeng Wang, and Mingjing Li. A Search-Based Web Image Annotation Method. In *ICME'07*, pages 655–658, 2007.
- [SC97] John R. Smith and Shih-Fu Chang. Visually searching the web for content. *IEEE MultiMedia*, 4:12–20, July 1997.
- [Sch05] Ingo Schmitt. *Ähnlichkeitssuche in Multimedia-Datenbanken - Retrieval, Suchalgorithmen und Anfragebehandlung*. Oldenbourg, 2005.
- [SCS99] Stan Sclaroff, Marco La Cascia, and Saratendu Sethi. Unifying textual and visual cues for content-based image retrieval on the World Wide Web. *Computer Vision and Image Understanding*, 75(1-2):86–98, 1999.
- [Sha76] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.
- [Sib73] R. Sibson. SLINK: An Optimally Efficient Algorithm for the Single-Link Cluster Method. *The Computer Journal*, 16(1):30 – 34, 1973.
- [SJ88] Karen Sparck Jones. *A statistical Interpretation of Term Specificity and its Application in Retrieval*, pages 132–142. Taylor Graham Publishing, London, UK, UK, 1988.

- [SOT00] Heng Tao Shen, Beng Chin Ooi, and Kian-Lee Tan. Giving meanings to www images. In *Proceedings of the eighth ACM international conference on Multimedia*, MULTIMEDIA '00, pages 39–47, New York, NY, USA, 2000. ACM.
- [Spe99] HTML 4.01 Specification. Objects, Images, and Applets in HTML documents. <http://www.w3.org/TR/html401/struct/objects.html#h-13.2>, December 1999. Last access on March 14th, 2011.
- [Spi11] SpiegelONLINE. The official web site of the german news magazine. <http://www.spiegel.de>, 2011. Access on March 14th, 2011.
- [STC97] Stan Sclaroff, Leonid Taycher, and Marco La Cascia. Imagerover: A content-based image browser for the world wide web. In *Proceedings of the 1997 Workshop on Content-Based Access of Image and Video Libraries (CBAIVL '97)*, CAIVL '97, pages 2–, Washington, DC, USA, 1997. IEEE Computer Society.
- [STG⁺11] Adrian Sandor, Andy Tripp, Fabrizio Giustina, Gary L. Peskin, Sami Lempinen, Russell Gold, James Sanders, and Steffen Yount. The Homepage of the JTidy Java API. <http://jtidy.sourceforge.net/>, 2011.
- [Tri10] Gergina Trifonova. Entwicklung eines Tools zur manuellen Zuordnung von Kontextinformationen zu Webbildern, September 2010. Bachelor Thesis.
- [TSK05] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.
- [Uni11] Oxford University. The web page maintaining the British National Corpus (BNC). <http://www.natcorp.ox.ac.uk/>, 2011. Access on March 20th, 2011.
- [Vin09] G. Vineel. Web page DOM node characterization and its application to page segmentation. In *Internet Multimedia Services Architecture and Applications (IMSAA), 2009 IEEE International Conference on*, pages 1–6, 2009.
- [VL00] Nuno Vasconcelos and Andrew Lippman. Bayesian relevance feedback for content-based image retrieval. In *CBAIVL '00: Proceedings of the IEEE Workshop on Content-based Access of Image and Video Libraries*

- (*CBAIVL'00*), page 63, Washington, DC, USA, 2000. IEEE Computer Society.
- [WJZZ06] Changhu Wang, Feng Jing, Lei Zhang, and Hong-Jiang Zhang. Scalable search-based image annotation of personal images. In *Proceedings of the 8th ACM international workshop on Multimedia information retrieval, MIR '06*, pages 269–278, New York, NY, USA, 2006. ACM.
- [WZJM06] Xin-Jing Wang, Lei Zhang, Feng Jing, and Wei-Ying Ma. AnnoSearch: Image Auto-Annotation by Search. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2, CVPR '06*, pages 1483–1490, Washington, DC, USA, 2006. IEEE Computer Society.
- [XPC11] XPCOM. A cross platform component object model. <https://developer.mozilla.org/en/XPCOM>, 2011.
- [YCWM03] Shipeng Yu, Deng Cai, Ji-Rong Wen, and Wei-Ying Ma. Improving pseudo-relevance feedback in web information retrieval using web page segmentation. In *Proceedings of the 12th international conference on World Wide Web, WWW '03*, pages 11–18, New York, NY, USA, 2003. ACM.
- [YhTjW05] Tian Yong-hong, Huang Tie-jun, and Gao Wen. Exploiting multi-context analysis in semantic image classification. *Journal of Zhejiang University SCIENCE*, pages 1268–1283, 2005.
- [ZXJQH05] Hua Zhigang, Wang Xiang-Jun, Liu Qingshan, and Lu Hanqing. Semantic knowledge extraction and annotation for web images. In *MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia*, pages 467–470, New York, NY, USA, 2005. ACM.

LIST OF FIGURES

1.1	From low-level pixels to high-level semantics: the visual Semantic Gap.	2
2.1	Visual features of selected Web Contents	11
2.2	An Example HTML snippet a) HTML, b) browser and c) DOM representation.	13
2.3	Part of the hierarchy of DOM interfaces.	15
2.4	What is the perfect Clustering? Different Ways to group Points to Clusters	21
3.1	Spiegel-Online [Spi11] Excerpt: Browser (left) and DOM (right) Representation	32
3.2	Frame of n terms surrounding an image in a list of terms and images. .	33
3.3	The Paragraph Extractor applied on the Example Web Page	37
3.4	Different classes of images in browser and DOM representation [FHB09].	38
3.5	Yahoo! Auctions web page a) in browser view, b) as block representation, and c) as tree of visual content structure	41
3.6	Construction of Semantic Network [JHTS08]	46
4.1	The header of the CNN web site	50
4.2	Advertisement images and corresponding web context.	50
4.3	Links with web images to other CNN stories.	51
4.4	Image and corresponding context in the main part of web page: a) Long story article (TheCitizens), b) short story article (BBC) and c) detail view of image (FlickrR)	52
5.1	Overview of the different Modules of the Evaluation Framework	59
5.2	Web Document Preprocessing	61
5.3	Three different pages from the CNN News Page based on the same template	64
5.4	Average running times for different WICE methods over all collections	68
5.5	Golem-Collection Results: (left) precision (P) and recall (R), (right) F -score (F) and standard deviation of F -score ($\sigma(F)$)	69
5.6	Spiegel-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	71

5.7	Manual-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	72
6.1	Image-to-Article Mapping: A Schema of the main Concepts.	76
6.2	Two news articles from Spiegel ONLINE: (left) browser representation, (right) corresponding HTML source	78
6.3	A TABLE in a) source code, b) DOM tree and c) browser representation. The dotted lines are relationships that can not be detected in the one dimensional representations. The solid line is a relationship that does not exists in the browser representation.	82
6.4	An example of an HTML table in 2-DOM representation.	83
6.5	Precision, Recall and F-score Results for the BBC Collection	85
6.6	Precision, Recall and F-score Results for the Heise Collection	86
6.7	Precision, Recall and F-score Results for the Manual Collection	86
6.8	Precision, Recall and F-score Results for the Wikipedia Collection	87
6.9	Standard Deviation of F -scores of Article-based WICE on different Collections	88
6.10	Average Processing Times over all Collections	89
7.1	Preconditions for the DOM-distance function	94
7.2	A simplified example of a DOM tree. Weights are computed using formula in Equation 7.2 with $c=1$	95
7.3	The minimal geometric distance between two rectangles R and S	96
7.4	Computational times for the distance-matrices of different distance measures.	101
7.5	The start page of WIMS 2011 and possible blocks in which the page could be segmented. The numbers are the id's of the contents arranged in the order as they appear in HTML.	101
7.6	Visualized distance matrices for a) DOM distance, b) geometric distance and c) semantic distance.	102
7.7	Partitioning six Web Contents into two Clusters using Geometric Distance	104
7.8	Distribution of Web Contents over an one-dimensional Space	107
7.9	Distances of adjacent contents and the threshold values computed by weighted averaging with gaussian. The peaks at which the distance exceeds the threshold point to the corresponding separators in the browser output of the document.	109
7.10	Average Processing Times over all Collections	110
7.11	Golem Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	111

7.12	The Globe&Mail Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	112
7.13	Plotting of adjacent Web Content Distances (solid-line) and their Gaussian-Smoothed (dashed-line)	112
7.14	BBC Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	113
7.15	Manual-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	114
A.1	Golem-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	135
A.2	Heise-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	135
A.3	MSN-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	136
A.4	The Globe&Mail-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	136
A.5	Spiegel-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	137
A.6	Wikipedia-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	137
B.7	BBC-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	138
B.8	CNN-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	138
B.9	Manual-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	139
B.10	New York Times-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	139
B.11	Telegraph-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	140
B.12	Yahoo!-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)	140

LIST OF TABLES

2.1	Confusion Matrix of a binary Classification	24
5.1	Test collections with total number of documents and images	65
6.1	Logical classes of text contents appearing in most news articles	79
7.1	Average Dunn index for different Distance Measures and different Clustering Techniques	103
7.2	Average Rand statistic for different Distance Measures and different Clustering Techniques	105

APPENDIX

The experimental results obtained in the Evaluation studies were not fully presented during the evaluation discussions. Here we include them by reason of completeness.

A Evaluation Results for Simple-Structure-Document Collections

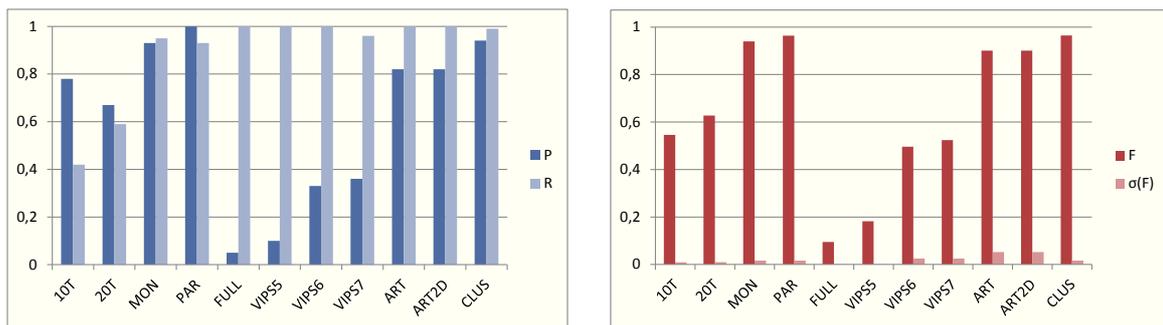


Figure A.1: Golem-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

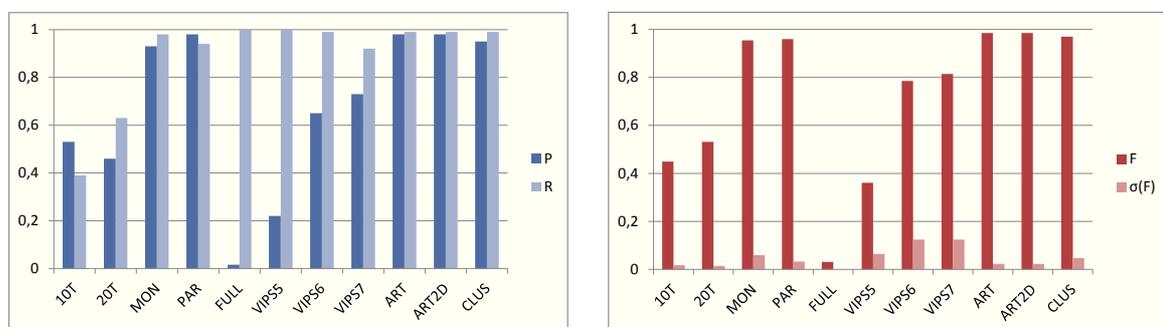


Figure A.2: Heise-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

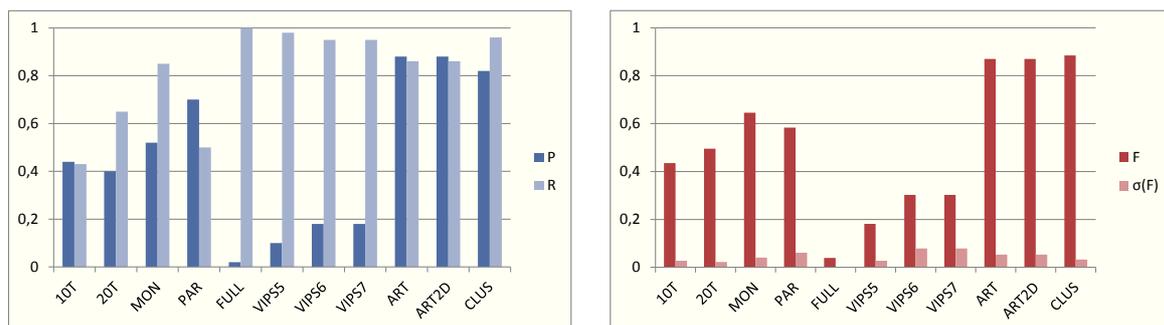


Figure A.3: MSN-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

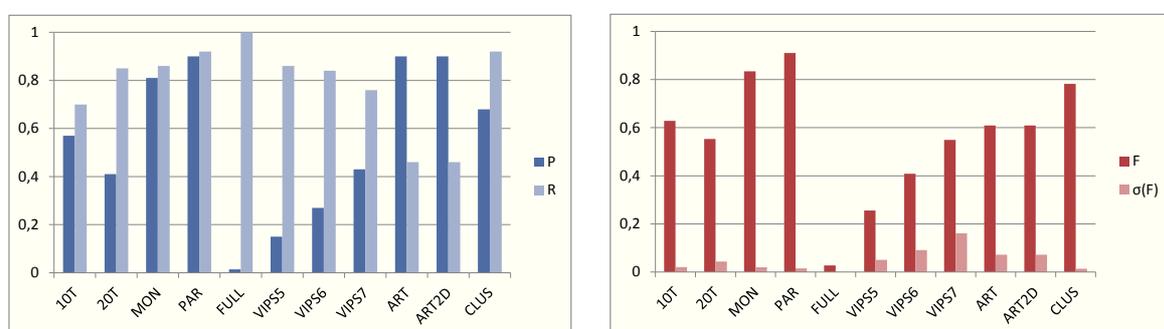


Figure A.4: The Globe&Mail-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

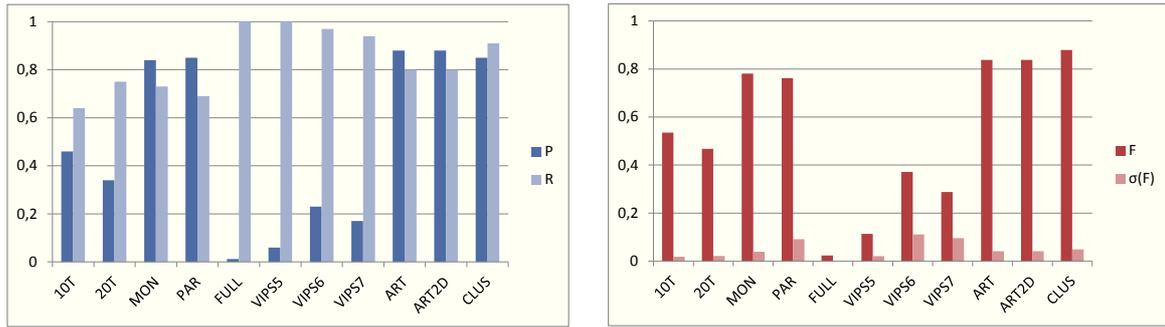


Figure A.5: Spiegel-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

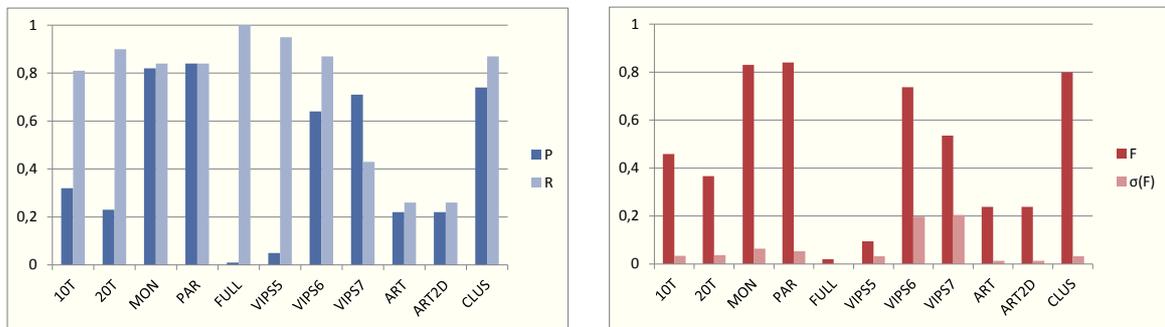


Figure A.6: Wikipedia-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

B Evaluation Results for Complex-Structure-Document Collections

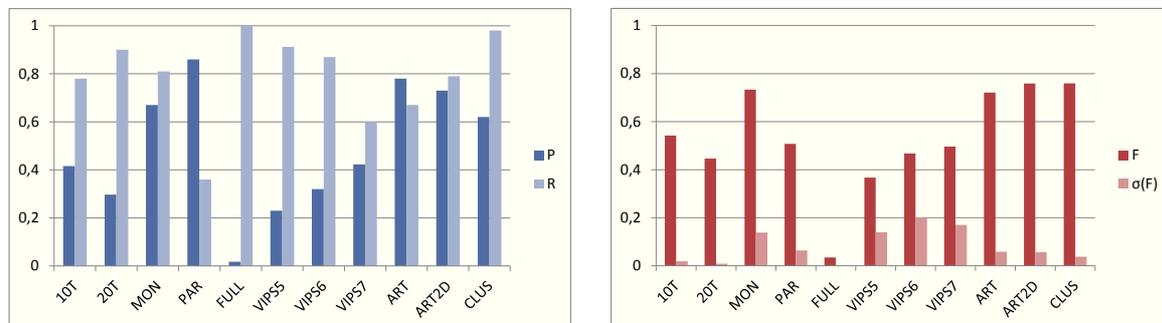


Figure B.7: BBC-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

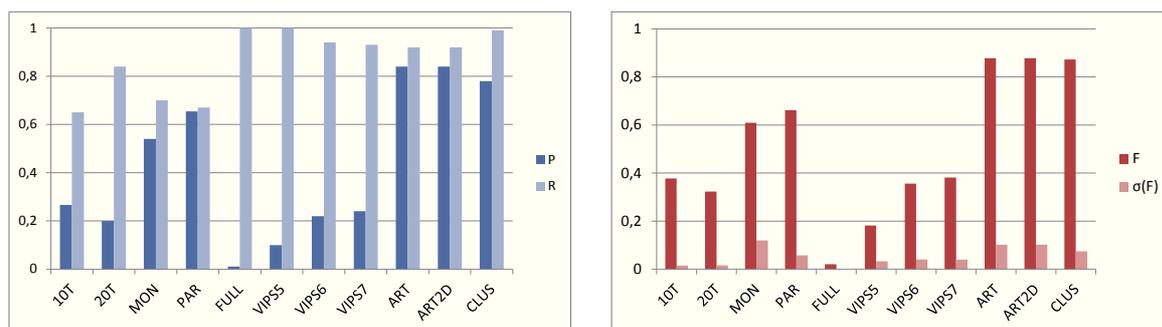


Figure B.8: CNN-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

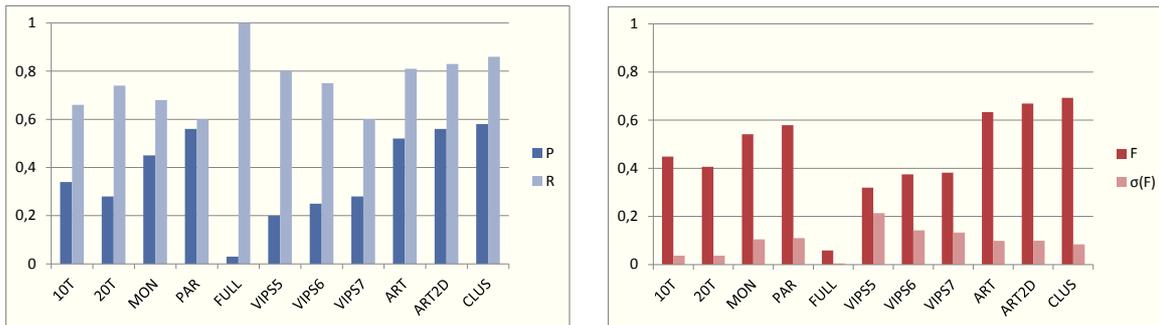


Figure B.9: Manual-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

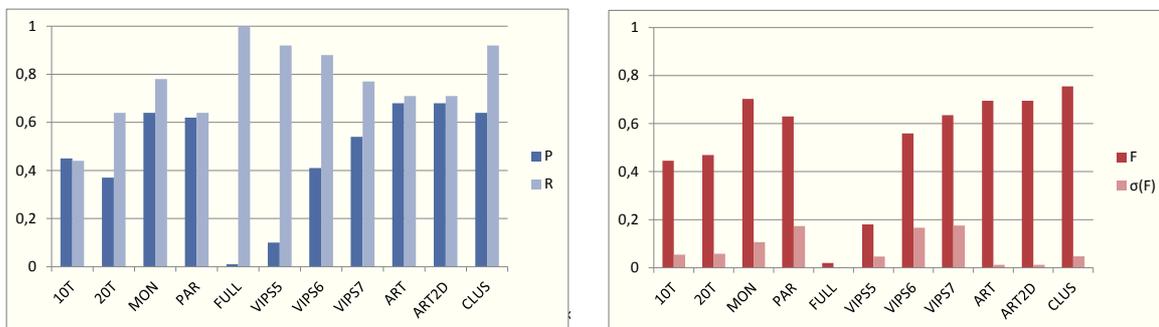


Figure B.10: New York Times-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

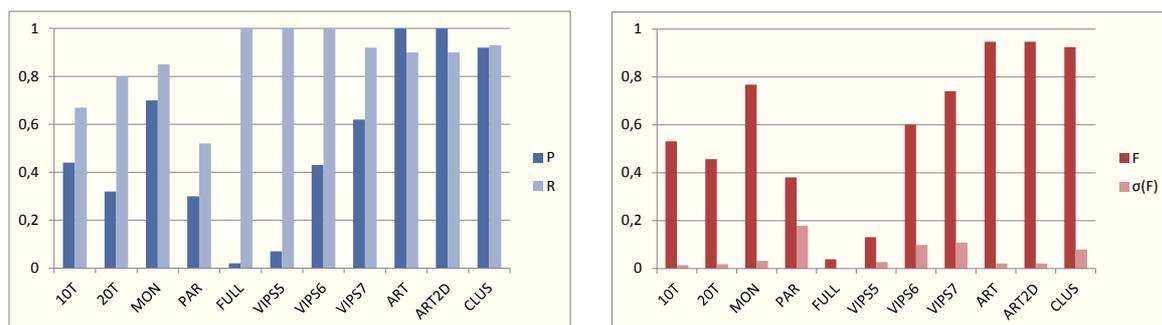


Figure B.11: Telegraph-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)

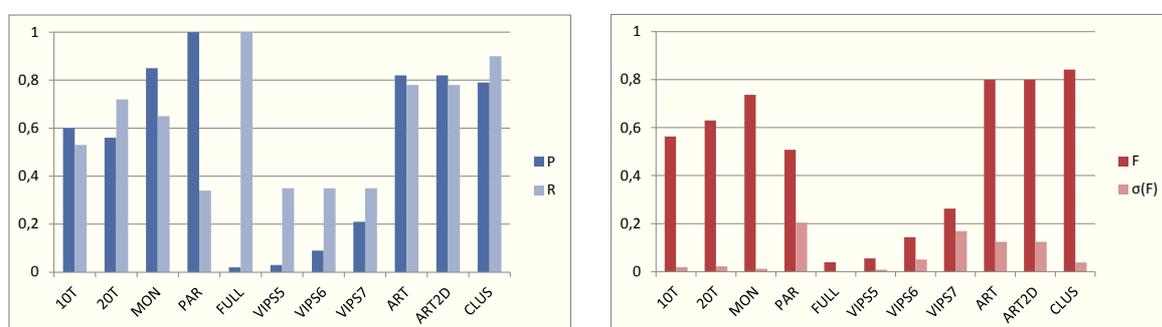


Figure B.12: Yahoo!-Collection Results: precision (P) and recall (R) (left), F -score (F) and standard deviation of F -score ($\sigma(F)$) (right)