

PUBLICATIONS OF
THE UNIVERSITY OF EASTERN FINLAND

*Dissertations in Forestry and
Natural Sciences*



NAJLAH GALI

SUMMARIZING THE CONTENT OF WEB PAGES

NAJLAH GALI

Summarizing the Content of Web pages

Publications of the University of Eastern Finland
Dissertations in Forestry and Natural Sciences
Number 259

Academic Dissertation

To be presented by permission of the Faculty of Science and Forestry for public examination in Louhela auditorium in Science Park Building at the University of Eastern Finland, Joensuu, on January 27, 2017, at 12 o'clock p.m.

School of Computing

Grano Oy
Jyväskylä, 2017
Editor: Research Dir. Pertti Pasanen

Distribution:
University of Eastern Finland Library / Sales of publications
P.O.Box 107, FI-80101 Joensuu, Finland
tel. +358-50-3058396
www.uef.fi/kirjasto

ISBN: 978-952-61-2407-0 (printed)
ISBN: 978-952-61-2408-7 (PDF)
ISSN: 1798-5668
ISSN: 1798-5668
ISSN: 1798-5676 (PDF)

Author's address: University of Eastern Finland
School of Computing
P.O. Box 111
80101 JOENSUU
FINLAND
Email: najlaa@cs.uef.fi

Supervisor: Professor Pasi Fränti, Ph.D
University of Eastern Finland
School of Computing
P.O. Box 111
80101 JOENSUU
FINLAND
Email: franti@cs.uef.fi

Reviewers: Professor Olfa Nasraoui, Ph.D
University of Louisville
Department of Computer Engineering & Computer Science
LOUISVILLE, KY 40292
USA
Email: olfa.nasraoui@louisville.edu

Professor Jyrki Nummenmaa, Ph.D
University of Tampere
School of Information Sciences
P.O. Box 33014
33100 TAMPERE
FINLAND
Email: jyrki.nummenmaa@cs.uta.fi

Opponent: Professor Jari Veijalainen, Ph.D
University of Jyväskylä
Department of Computer Science & Information Systems
P.O. Box 35
40014 JYVÄSKYLÄ
FINLAND
Email: veijalai@cs.jyu.fi

ABSTRACT

With the enormous amount of information available on the World Wide Web, locating necessary information efficiently is not a trivial matter. As web page summarization helps users to judge the relevance of information to what they are interested in more quickly and makes their web browsing easier, it saves their time. Although this summarization can offer a needed solution to information overload, automatically generating a summary is a challenge. In addition to the underlying structure embedded in the HTML language, a web page contains many irrelevant components that are unnecessary for fast content summarization. In this thesis, we develop new methods to extract a compact summary from any HTML web page. We present novel methods to extract a title, keywords, and a representative image. We compare the performance of these methods to the performance of existing state-of-the-art methods and show that the former outperform latter. The proposed methods are then integrated into a location-based application called Mopsi that is available to users through a website and mobile phone application. We compare different similarity measures for various data types and establish a novel setup for experiments, which allows for a systematic evaluation of the measures. Our results improve the current state-of-the-art methods and provide readily available solutions for web page summarization problems.

Universal Decimal Classification: 001.814, 004.439, 004.62, 004.912, 004.93

INSPEC Thesaurus: data mining; information analysis; text analysis; abstracting; pattern recognition; text detection; image recognition; data structures; string matching; content management; hypermedia markup languages; Web sites; mobile computing

Yleinen suomalainen asiasanasto: tiedonlouhinta; tekstinlouhinta; automaattinen sisällönkuvalu; tekstintunnistus; tiivistelmät; avainsanat; kuvat; HTML; WWW-sivustot; WWW-sivut; mobiilisovellukset

Preface

The work presented in this thesis was carried out in the Machine Learning Group at the School of Computing, University of Eastern Finland, Finland, during the years 2014 – 2016.

I would like to express my sincere gratitude to my supervisor Professor Pasi Fränti, who offered me the opportunity to work in his research group. Without his great enthusiasm and strong encouragement throughout the years, this thesis would never have been accomplished.

I wish to thank my colleague, Radu Marinescu-Istodor, for his endless discussions and guidance in the research. Without his help, this study would not have proceeded positively. I also would like to acknowledge my colleagues, whom I have had the pleasure to work with during these years, especially Andrei Tabarcea and Mohammad Rezaei. I sincerely thank Hana Vrzakova for her support and assistance during the writing of this thesis.

I would like to thank Oili Kohonen for her support and care. You have been more than a colleague to me and I feel lucky to have made a friend like you.

I am especially grateful to Professor Olfa Nasraoui and Professor Jyrki Nummenmaa, the reviewers of this thesis, for their feedback and valuable comments.

I greatly appreciate my family and all of my friends, who have given me strength during these years. Words cannot express how grateful I am to my parents, Mohammed and Mulkia, who during all these years away from home have been in my heart. These moments would never have come if I did not have their continuous love and support in the best and worst moments of my life.

Finally, I am forever indebted to my dear husband, Amer, for his endless love and support on my life and work and our children Omar and Dina, they are the love of my life.

Joensuu, January 1, 2017

Najlah Gali

LIST OF ABBREVIATIONS

ASCII	American Standard Code for Information Interchange
CRF	Conditional random field
CSS	Cascading Style Sheets
DOM	Document object model
GPS	Global Positioning System
HITS	Hyperlink-induced topic search
HTML	Hyper Text Markup Language
JS	JavaScript
K-NN	K-nearest neighbor
LCS	Longest common substring
MSE	Mean square error
OCR	Optical character recognition
PDA	Personal digital assistant
POS	Part-of-speech
SEO	Search engine optimization
SVM	Support vector machine
TF	Term frequency
TF-IDF	Term frequency-inverse document frequency
TTA	Title tag analyzer
URL	Uniform resource locator
VIPS	Vision-based segmentation
XML	Extensible Markup Language

LIST OF ORIGINAL PUBLICATIONS

This thesis presents a review of the author's work in the field of web content summarization and the following selection of the author's publications:

- [P1] N. Gali and P. Fränti, "Content-based title extraction from web page," *Int. Conf. on Web Information Systems and Technologies (WEBIST 2016)*, Rome, Italy, vol. 2, 204-210, April 2016.
- [P2] N. Gali, R. Mariescu-Istodor and P. Fränti, "Using linguistic features to automatically extract web page title," 2016, (submitted).
- [P3] N. Gali, R. Mariescu-Istodor and P. Fränti, "Similarity measures for title matching," *Int. Conf. on Pattern Recognition (ICPR 2016)*, Cancún, Mexico, 1549-1554, December 2016.
- [P4] M. Rezaei, N. Gali, and P. Fränti, "CIRank: a method for keyword extraction from web pages using clustering and distribution of nouns," *IEEE/WIC/ACM Int. Joint Conf. on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, 79-84, December 2015.
- [P5] N. Gali, A. Tabarcea, and P. Fränti, "Extracting representative image from web page," *Int. Conf. on Web Information Systems & Technologies (WEBIST'15)*, Lisbon, Portugal, 411-419, 2015.

Throughout the thesis, these papers are referred to as [P1]–[P5]. They are included at the end of this thesis by the permission of their copyright holders.

AUTHOR'S CONTRIBUTION

The ideas for papers [P1], [P2], and [P5] all originated from the author, and they were jointly refined via discussion with all co-authors. The idea for paper [P4] emerged from all authors jointly. Prof. Pasi Fränti originated the idea for paper [P3].

The author carried out the research and performed most of the experiments for all of the papers [P1]-[P5].

The author implemented the methods entirely for [P1] and partially for [P2]-[P5], while the co-authors contributed to the implementation and experimentation. The author wrote all of the papers, namely [P1]-[P5].

Contents

1 Introduction	1
2 HTML Web Page.....	7
2.1 THE STRUCTURE OF HTML WEB PAGE	9
2.2 THE DOCUMENT OBJECT MODEL TREE	11
3 Extracting Title	13
3.1 CONTENT SOURCE	14
3.2 CONTENT ANALYSIS	17
3.3 CANDIDATE TITLES	18
3.4 EXTRACTED FEATURES.....	20
3.5 RANKING CANDIDATE TITLES.....	21
3.6 SIMILARITY MEASURES FOR TITLE MATCHING	25
3.6.1 <i>String segmentation</i>	26
3.6.2 <i>String matching</i>	27
4 Extracting Keywords	35
4.1 CANDIDATE KEYWORD EXTRACTION	38
4.2 CANDIDATE KEYWORD SCORING	42
4.3 CLUSTER RANK	45
5 Extracting Images.....	51
6 Applications.....	61
6.1 INTRODUCTION TO MOPSI.....	61
6.2 LOCATION-BASED SEARCH AND RECOMMENDER..	63
6.3 THE INTERACTIVE TOOL FOR SERVICES.....	64

6.4 CLUSTERING65

7 Summary of Contributions69

8 Conclusions.....73

References75

Appendix: Original Publications

1 Introduction

With the rapid growth of the World Wide Web, the amount of information available online is becoming incredibly high and leading to information overload [69]. Users have access to a large number of information sources with the aid of search engines; however, finding relevant pages is challenging. A search engine such as Yahoo! often returns thousands of pages for a single query, of which 50% are less relevant to user desired information [47]. A typical user navigates through the top-ranked pages manually to find the relevant information. A *summary* saves a user's time and is an ideal solution for coping with information overload [119].

Summarization helps the user get a general overview of a web page's main content before deciding whether to read it more in-depth. Besides making web browsing and navigation easier, summarization also makes browsing faster as an entire web page does not need to be downloaded before viewing [2]. It can also improve the search engine indexing of web pages and thus provide more relevant search results to the end user. For instance, matching a user's query words with web page keywords yields a smaller and more relevant list of results than searching for the same query words in a full page. Summarization is also useful when the receiving device has limited storage capacity or bandwidth or a small screen that makes it difficult to view a web page's original content, as is true for personal digital assistants (PDAs) and cell phones [2].

The categorization of web pages is another domain in which summarization is useful. A summary shrinks a page's size and significantly reduces the number of features (i.e., words) that need to be considered. As a result, summarization overcomes the problem of high space dimensionality [59, 93]. Social networking services such as Facebook and Google+ and business directories such as Foursquare, TripAdvisor, and Yelp

are examples of sites that use a simple thumbnail and title as the summary of a web page.

Summarization produces a concise description of a web page, which is known as an *abstract* or *summary*. Two classes of summarization exist: extractive and abstractive [59]. In *extractive summarization*, a summary is formed solely from words, phrases, or sentences that are extracted from the page. In contrast, *abstractive summarization* requires natural language processing to interpret the information on the page and produces a summary that expresses the same meaning but more concisely (omitting unnecessary details) [6]. It also allows words and phrases that are not present in the page to be included. While useful, abstractive summarization has been far less popular than extractive summarization given that constructing the latter is easier [6]. In this thesis, we focus on extractive summarization.

Different types of summaries can be generated from a web page depending on the desired level of detail. The components can include a title, a set of keywords, a set of key phrases, short paragraphs, an image, a thumbnail of the web page snapshot, or a combination thereof. A *title* is a descriptive name given to a web page to distinguish it from other pages. It is the first piece of information about a page that a user reads and provides a quick insight into that page's content. *Keywords* and *keyphrases* are a few selected words and phrases that summarize the web page [83]. *Paragraphs* consist of several sentences that contain relevant information that is coherently organized. A *representative image* is an image that describes the page content in a visual form. A *web page thumbnail* is a scaled-down snapshot of a page as rendered in the web browser. In this thesis, we aim to summarize Hypertext Markup Language (HTML) web pages by their title [P1, P2], keywords [P4], and representative image [P5].

Extracting a summary from a web page is not straightforward [93]. A web page typically combines different kinds of data. In addition to the main content, it contains a large amount of additional information, such as functional and design elements, navigation bars, advertisements, and commercial banners (see Figure 1.1).

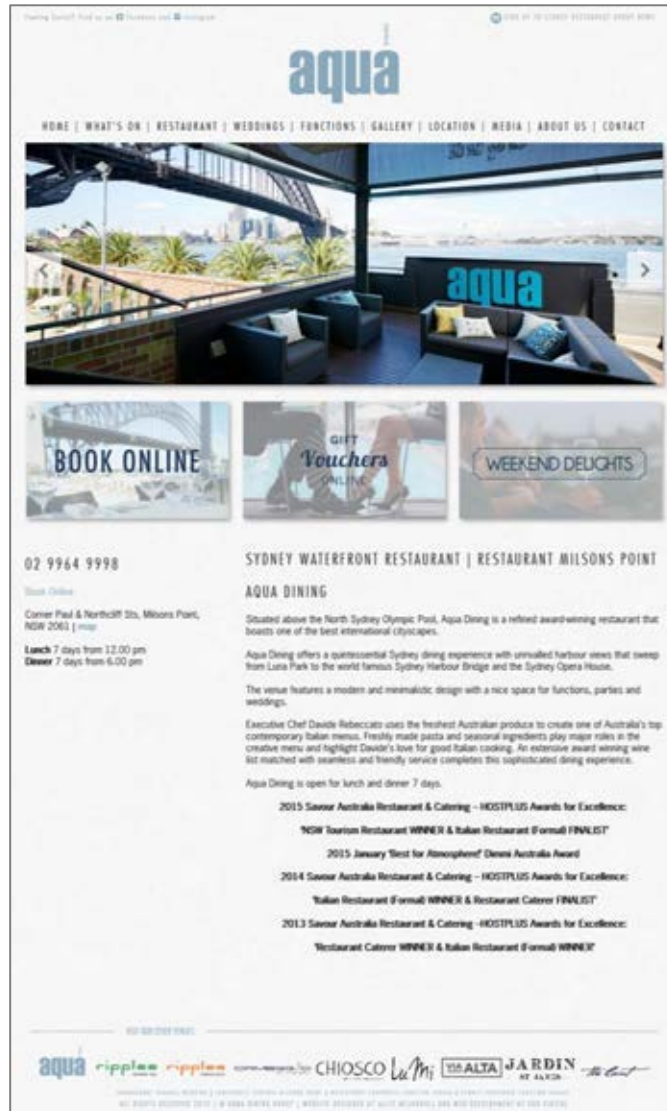


Figure 1.1: Example of a restaurant web page

It has been estimated that 40–50% of a web page's content is irrelevant to the end user [33]. Web pages also have their underlying embedded structure in HTML, which makes extractive summarization even more difficult [120].

Manually constructed summaries are subjective, labor intensive, and error prone; automatic summarization has emerged to address these challenges. Automatic summarization methods have mostly been developed for web pages that follow

a standard template that involves a title at the top followed by the article's main text (as commonly used inter alia by news and educational institution sites). It is noteworthy that the amount of irrelevant text is significantly lower on news pages compared to other types of pages (e.g., information pages) [33].

Extractive summarization has not been investigated widely in relation to service-based web pages (as needed for *entertainment*, *sport*, and *restaurants*) [P2]. One reason is that in this context, the main text is usually scattered all over the page. Another reason is that these types of web pages do not follow a certain template or common format. They are designed in a presentation-oriented fashion with significant layout variations that influence the way in which a user browses the page [111]. For example, in Figure 1.2 the titles are located in different places within the body of the pages, with no visual differentiation from other parts of the text. Furthermore, the titles in the top left corner are represented by logo images that cannot be easily processed as text (see Section 3.1).

In this thesis, our objective is to extract compact summaries from general HTML web pages and utilize them in a location-based application called Mopsi.¹ We also examine the performance of 21 similarity measures for matching title phrases and clustering web pages.

¹ <http://cs.uef.fi/mopsi>

Introduction

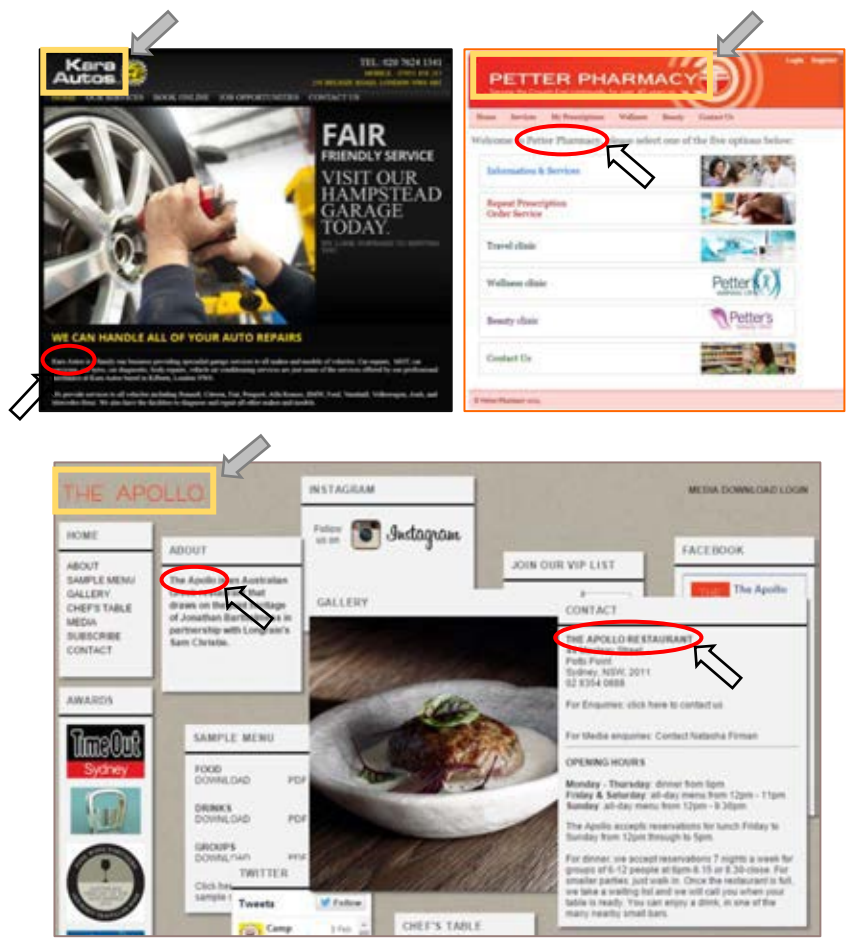


Figure 1.2: Different web page layouts. The yellow squares refer to logo images, while the red ovals refer to titles located in different part of the pages

2 HTML Web Page

A web page is a document on the World Wide Web; it is typically written in HTML and identified by a uniform resource locator (URL). As of May 31, 2016, about 4.57 billion web pages had been indexed by the Google and Bing search engines.² A common factor among web pages is that they often feature similar design elements (e.g., a logo, navigation menu, title, main content, footer, and additional content)³ despite discussing different topics (such as news, sports, food, entertainment, or shopping), see Figure 2.1.

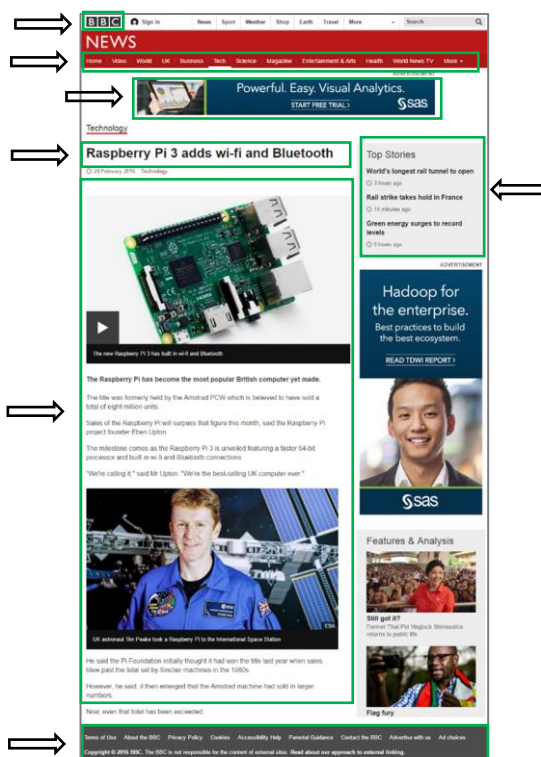


Figure 2.1: Example of a news page

² <http://www.worldwidewebsite.com/>

³ https://developer.mozilla.org/en-US/Learn/Common_questions/Common_web_layouts

These common web page elements can be described as follows:

- A *logo* is a graphic representation, stylized name, or symbol designed to identify the commercial enterprise, institution, or individual that owns a website. It is usually stored as an image, although it is sometimes stored as a styled text. Logos are generally placed in the top region of a web page, which is also known as the header.
- A *navigation menu* is a user interface within a web page that contains links to other parts of the website. It is typically displayed as a horizontal list of links in the header section, although it may also be placed vertically on the left side of the page (in this case it is called a *sidebar*). Some pages feature both a horizontal navigation menu on the top and a vertical sidebar on the left, each with different content.
- A *title* is a name, caption, or heading that describes the web page's content [P1]; for example, in Figure 2.1 the title is 'Raspberry Pi 3 adds wi-fi and Bluetooth.'
- The *main content* includes information unique to the current page, such as main text, audio, video, structured records (e.g., lists and tables), and relevant images. Images are used because they provide a good representation of content and can communicate information to people better than text [40]. Furthermore, between 80% and 90% of the information that is received by the brain is visual [46, 49]. Another reason is that people are naturally more attracted to images than to links or text. Images can exist on a web page for different purposes. They may be either directly relevant to the page's main content or included for advertising or functional reasons (for instance, navigational banners, icons, and images may serve as section headings).
- The *footer* can be seen on the bottom of the page. It usually contains hyperlinks, contact information, and copyrights.

- A web page can also have additional content, such as *related articles and links, advertisements, audio, video, and author and category information.*

2.1 THE STRUCTURE OF HTML WEB PAGE

Thus far we have investigated a web page's layout as it appears to users. In this section, we explore it from a technical perspective.

A web page is semi-structured. The data does not have a regular structure as in a relational database [37], although due to HTML tags it is also not completely unstructured. A web page is usually designed using three components: *HTML*, *Cascading Style Sheets (CSS)*, and *JavaScript (JS)* [91].

The *HTML* markup language is used to create a web page's structure. It defines elements called *tags* that surround plain text to constitute a page and add meaningful description to its content. For example, a headline for a news article is expected to be marked using `<h1>`, as it is the most important heading on the page.

A web page comprises two components. The first is the header, which contains tags that do not have any visual effect but add information about the page (e.g., comments, title, and metadata tags). The second part is the body, which consists of the data that is displayed by the browser. A web browser parses an HTML page by interpreting the tags in a sequence to display the web page. For example, a web browser presents all of the content after an opening tag `` as bold until it finds the closing tag ``. Listing 2.1 shows a portion of an HTML file that composes navigation menu items.

Listing 2.1: A portion of an HTML file composing navigation menu items.

```

----- Navigation menu -----
1      <div class="orb-nav-section" id="orb-nav-links">
2      <h2> BBC navigation</h2>
3      <ul>
4      <li><a href="http://www.bbc.com/news/">News</a></li>
5      <li><a href="/sport/">Sport</a></li>
6      <li><a href="/weather/">Weather</a></li>
7      <li><a href="http://shop.bbc.com/">Shop</a></li>
8      <li><a href="http://www.bbc.com/earth/">Earth</a></li>
9      <li><a href="http://www.bbc.com/travel/">Travel</a></li>
10     </ul>
11     </div>

```

A web page also often contains CSS and JS components. Components in CSS are used to define a web page's style, format, and layout; in essence, they describe how HTML elements should be displayed to the end user. An HTML element is the content from a start tag to an end tag, such as `<p>Web data</p>`.⁴ The CSS was developed to separate a web page's content from its styling information (e.g., layout, colors, and fonts) [15]. It reduces the complexity and repetition of the styling instructions in every tag and allows several web pages to share the same styling file (.css).

In contrast, JS is a programming language used to make web pages interactive. For example, it can validate an email address input to a form field by a user. This language has several advantages, such as reducing server interaction by validating the input data before sending it to the server. It can also be used to manipulate a web page's elements when a user clicks on them.

⁴ http://www.w3schools.com/HTML/HTML_elements.asp

2.2 THE DOCUMENT OBJECT MODEL TREE

An HTML page can be represented using a tree structure, referred to as *document object model (DOM) tree*,⁵ in which each of the page's HTML elements is a branch or leaf node in the tree. The nodes in a DOM tree have a hierarchical relationship to each other, which can be described using parent, child, and sibling. The top node (`<HTML>`) is the root of the tree and the nodes with no children are the leaves that contain the actual text.

A DOM tree allows scripts and programs to dynamically access and update a web page's elements, including its content, structure, and style. Figure 2.2 illustrates a portion of a DOM tree and the relationship between the HTML elements.

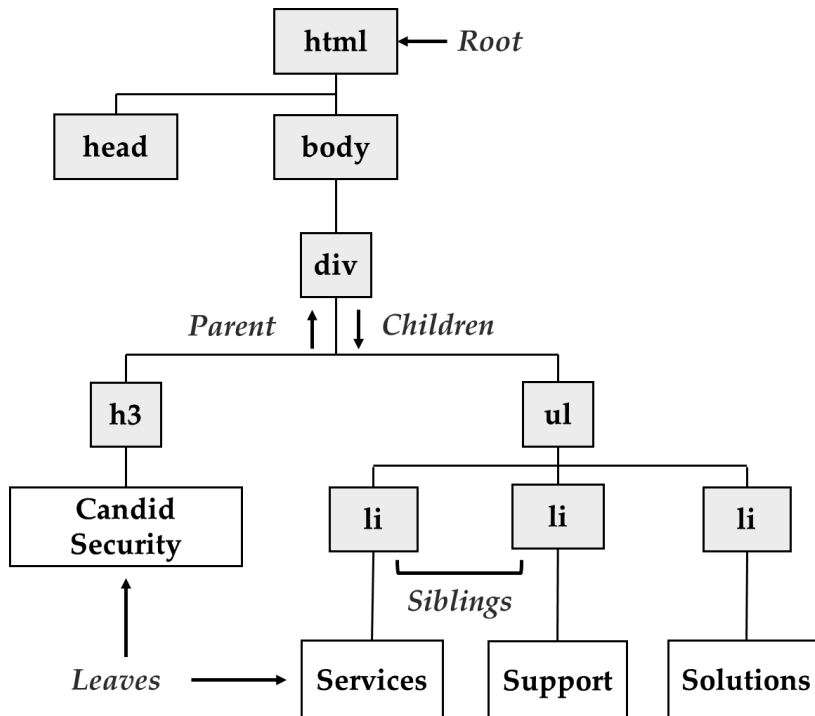


Figure 2.2: Example of a DOM tree and the relationships between HTML elements

⁵ www.w3.org/DOM

3 *Extracting Title*

When users read about a web page in different applications (such as the results page of a search engine and a location-based search engine), the title is the first piece of information they see. It can be any word or phrase that distinguishes the page's content. A title needs to be descriptive, concise, not too long, and grammatically correct [45]. Having a correct title improves a web page's retrieval, as reported in [115]. Titles also help with the indexing of web pages and thus improve search engines' ability to provide more relevant search results [66]. Moreover, they are useful in several web page applications, such as browsing and filtering [41].

Existing research concerning title extraction focuses mainly on extracting a title from the body of a web page that follows a standard format (such as news pages). It is assumed that a title is always located in the top region of a page and has visual prominence. For example, Hu et al. [45] and Xue et al. [115] explicitly state that a title must be in the top area of the page. Furthermore, Fan et al. [24] hypothesize that a title is located in the upper part of the main text. Changuel et al. [16] implicitly assume that a title appears in the top portion of a page and as a result extract only the first 20 text nodes from the DOM tree. All of these assumptions are often correct, but a title can generally appear anywhere in a page without visual distinction from other headlines – especially when a logo image is used to present the main title (see Figure 1.2). Our methods in [P1] and [P2] are independent of the visual features and the structure of the web page.

Another assumption often made is that the title in a body is a separate line of text (i.e., it has its own text node in the DOM tree). This is usually the case, although the title can generally also appear as a part of other phrases in a DOM tree's text nodes. For example, `<h1>Welcome to Petter Pharmacy, please`

select one of the five options below: `</h1>`. According to our experiments, about 68% of the title nodes also contain additional information, which motivates the segmentation approaches used in [P1] and [P2]. In [P1], we segment the content of the text node by delimiters such as punctuations and white space, while in [P2] we use natural language processing. Figure 3.1 shows typical steps for title extraction (left) and possible approaches to each step (right); the modules that are covered in [P1] and [P2] are highlighted in blue.

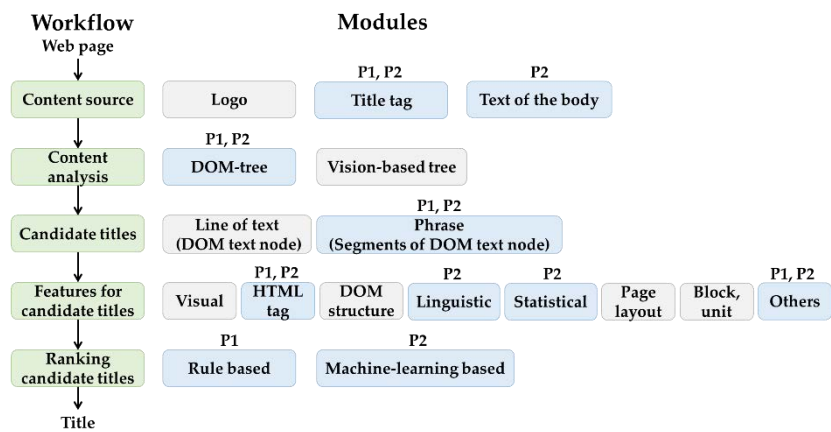


Figure 3.1: Typical steps for title extraction

3.1 CONTENT SOURCE

The title of a web page is usually found in one or more of three places: the *title tag* (i.e., between `<title>` and `</title>`), the *text of the body*, and the *logo image* (see Figure 3.2). According to our experiments with the Titler data set⁶ (which consists of 1002 websites [P2]), the occurrence of the title in these three places is as follows:

- Title tag (91%)
- Text of the body (97%)
- Logo (89%)

⁶ <http://cs.uef.fi/mopsi/titler>

<title>National Museum of Scotland</title>

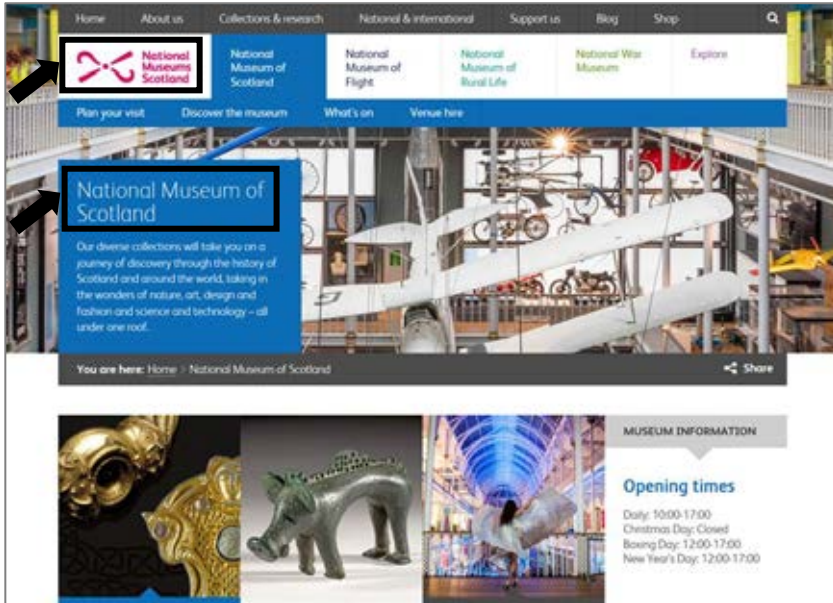


Figure 3.2: Sources for a title

The *title tag* is the obvious source, and a page's author is expected to fill it with a proper title. However, people often do not complete this tag carefully as it does not have a visual impact on the page [P2]. A title tag often contains additional text, such as the name of the hosting website, information about the place offering services, a slogan, and contact details. Nevertheless, about 91% of the 1002 websites in our sample include the correct title in their title tag (see Table 3.1). A similar observation was reported in [78]. The title tag is therefore a potential source for candidate title extraction, although existing methods rarely use it. In [P1], we consider the content of the title and meta tags as the only source for candidate title extraction.

The *body text* of a web page is a second source for a title. It has been given more focus by researchers given that a title in the body is visible to users and is thus expected to be written more carefully than the title tag [16, 45, 50, 115, 107, P2]. However, extracting a title from the body of the web page is not an easy task, as roughly half of a page's content is irrelevant text [33].

This irrelevant text (e.g., advertisements) is often given even more visual emphasis than the main headlines, which makes the title extraction task even more challenging. Furthermore, no standard location exists in relation to title placement. In [P2], we extract candidate titles from both the web page’s body and title tag.

Table 3.1: The most typical problems related to title tag and the frequency they appear (according to our experiments).

	Proportion (%)	Example	Annotated title
Correct	29	<title> Hesburger </title>	Hesburger
Long description	62	<title>Brook's Diner 24 Hampden Square, Southgate, London N14 5JR 020 8368 7201 eat@brooks diner.com Like us on Facebook — Home</title>	Brook's Diner
Incorrect	6	<title>Hot Tubs, hot tub hire, swimming pools, Bristol, Gloucester</title>	Rio Pool
Vague	2.4	<title>home</title> <title>index</title> <title> </title>	Hellard Bros Ltd.
Short description	0.5	<title> Toby's Estate</title>	Toby's Estate Coffee
Empty	0.2	<title> </title>	Zavino Hospitality Group

The third source for a title is the *logo image*. However, extracting a title from this image would be very challenging. One reason is that the logo image must first be identified. Another reason is that the standard optical character recognition (OCR) approach would not generally work given that the image’s content is highly complex. We are not aware of any technique that attempts this approach. It should technically be possible, but as shown by the examples in Figure 3.3, such a technique would need to handle a wide variety of complex text

fonts that involve shadowing effects, textures, and other artistic features.



Figure 3.3: Examples of web page logos

3.2 CONTENT ANALYSIS

Most title extraction methods use either DOM tree representation or combine the DOM structure with a page's visual cues in a *vision-based tree*. A vision-based tree is built using the vision-based page segmentation algorithm (VIPS) introduced by Cai et al. [14]. The VIPS first extracts visual blocks from the DOM tree. Each node in the DOM tree can correspond to a block. Extremely large nodes such as `<table>` and `<p>` are further divided and replaced by their children based on a set of heuristic rules (e.g., HTML tags, background color, and line breaks). The process continues recursively until no further division is possible. The VIPS then finds visual separators, which are the vertical and horizontal lines between extracted blocks. Finally, it constructs a tree based on the visual properties of each block and the separator lines between them (see Figure 3.4). It is not necessary that nodes in the vision-based tree correspond to the node in the DOM-tree. The former provides a visual partitioning of the page where the blocks are grouped visually, while the latter describes the parent-child relationship between the tree nodes.

The VIPS needs to refer to all styling information (including external sheets) to locate each block's proper place in the tree. If the web page lacks rich visual properties, the hierarchies are incorrectly constructed. A wrong structure can also result from

the algorithm not detecting separators represented by thin images.

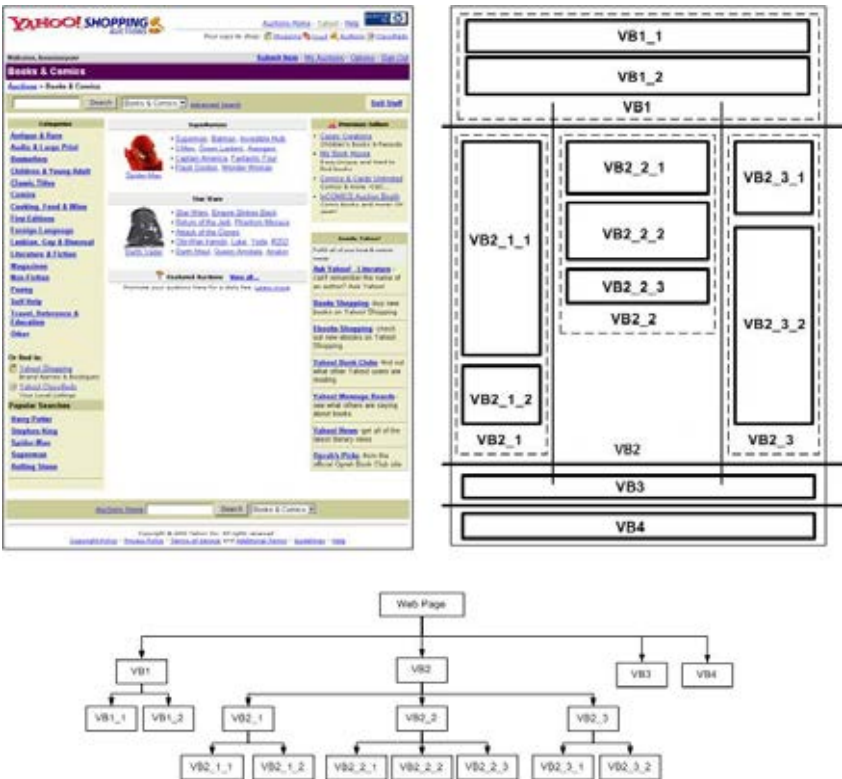


Figure 3.4: The layout structure and vision-based tree of an example page [14]

3.3 CANDIDATE TITLES

In both tree representations, existing methods use the entire text of the leaf nodes as candidate titles. We utilize the DOM tree approach in [P1] and [P2], where we divide the text within the DOM node into segments using delimiters [P1] and part-of-speech (POS) patterns [P2], respectively.

In [P1], we parse the web page into DOM tree and extract specified HTML tags (i.e., *title*, *meta*, *h1-h6*) using *XPath*. *XPath*⁷

⁷ <http://www.w3.org/TR/xpath20>

is a query language for addressing parts of an Extensible Markup Language (XML) document and extracting their text values. The content of the title tag and title meta tag is segmented using a set of predefined delimiters, as illustrated in Table 3.2. The distinct segments are then used as candidate titles.

Table 3.2: Pre-defined delimiter patterns.

space – space	space / space	space . space	space /
space : space	, space	space -	space <
: space	space :	space	-
space >	space «	space »	
? ,	- ,	space ::	

In [P2], we introduce a novel linguistic POS model for English titles. When building the POS model, we first add POS labels (or POS tags) to each title in the Titler corpus⁸ using Stanford Tagger⁹ [99]. We then generate 151 POS tag patterns with pattern's length varying from one to six by analyzing the POS tags that appeared among the ground truth titles. A *POS tag pattern* is a sequence of POS tags (e.g., <DT><JJ><NN>), where DT stands for determiner, JJ stands for adjective, and NN stands for noun. Using the predefined patterns, we extract all candidate phrases or sequence of words that match any of the POS tag patterns.

Instead of working with the entire texts of DOM nodes, we consider only those phrases that match any of the specified grammatical patterns. For example, given the text segment 'A warm welcome to Chinese Cricket Club,' we extract only the following (where NNP stands for a proper noun):

- Chinese NNP
- Cricket NNP
- Club NNP
- Chinese Cricket NNP NNP
- Cricket Club NNP NNP
- Chinese Cricket Club NNP NNP NNP

⁸ <http://cs.uef.fi/mopsi/titler>

⁹ <http://nlp.stanford.edu/software/tagger.shtml>

Because these structures match the patterns $\langle NNP \rangle$, $\langle NNP \rangle \langle NNP \rangle$, and $\langle NNP \rangle \langle NNP \rangle \langle NNP \rangle$, they are all valid syntactic structures for titles. However, we omit 'A_DT warm_JJ welcome_VB to_TO' (where VB stands for verb), as this segment does not fit into any pattern either entirely or partially.

3.4 EXTRACTED FEATURES

Researchers have extracted a wide range of features from either DOM or vision-based trees [115]. Those found in the literature are listed below.

Features from DOM tree:

- *Visual*: font weight, font family, font color [16, 45, 115]; font style, background color [45, 115]; alignment [24, 45, 115]; and font size [16, 24, 45, 115];
- *HTML tag*: bold, strong, emphasized text, paragraph, span, division [16]; image, horizontal ruler, line break, directory list [45, 115]; underline, list, anchor [16, 45, 115]; meta [P2]; position in tags [P1]; title [16, 24, 78, P2]; and heading level (h1- h6) [16, 24, 45, 115, P1, P2];
- *DOM structure*: number of sibling nodes in the DOM tree [45, 115]; relation with the root, parent, sibling, next, and previous nodes in terms of visual format [16, 45, 115];
- *Positional information*: position of the text unit from the beginning of the page's body and width of the text unit with respect to the width of the page [45];
- *Linguistic*: length of text, negative words, positive words [45, 115]; position in text [65]; syntactic structure, letter capitalization, and phrase length [P2]; and
- *Statistical*: term frequency [78]; term frequency-inverse document frequency [65, 78]; capitalization frequency, and independent appearance [P2].

Features from vision-based tree:

- *Page layout*: height, width, and position relative to the top left corner [115];
- *Block*: type, height, width, position [107, 115]; and front screen position [107];
- *Unit position*: from the top and left side of the page and from the top and left side of the block [115]; and
- *Content*: number of words in a block [107].

Other features:

- Web page URL [P1, P2].

The majority of these features are based on formatting, whereas the features we consider in [P1] and [P2] are independent of a page's design.

3.5 RANKING CANDIDATE TITLES

Ranking techniques can be divided into two broad classes: rule based and machine learning based [115].

Rule-based techniques use a set of predefined heuristic rules to score candidate titles. These rules are derived from the content of the DOM tree [24, 78, P1], the link structure between web pages [50], and the text [65]. The key advantage of the rule-based technique is that it does not require training data. Moreover, the technique is easy for humans to interpret and improve, as the weighting procedure and scoring formulas are explicit. However, heuristic methods often require determining thresholds and weights for feature parameters, which are not always straightforward to calculate. For example, if the number of features is $n = 9$ and each feature is assigned a value $m = 0$ to 5, it takes $O(m^n)$ time to test all weight combinations. In this example, testing would take about four months if each attempt took 1 second.

In the *TitleTagAnalyzer (TTA)* method [P1], we use a heuristic technique and apply three criteria. Two of these criteria are new, namely the position of segments in the URL (i.e., host, path, and document name) and the position of segments in the title and meta tags (i.e., first, middle, and last). The third criterion is the popularity among heading tags (see Figure 3.5). We use the heading rule, as the headlines in a web page’s body are usually emphasized by heading tags.

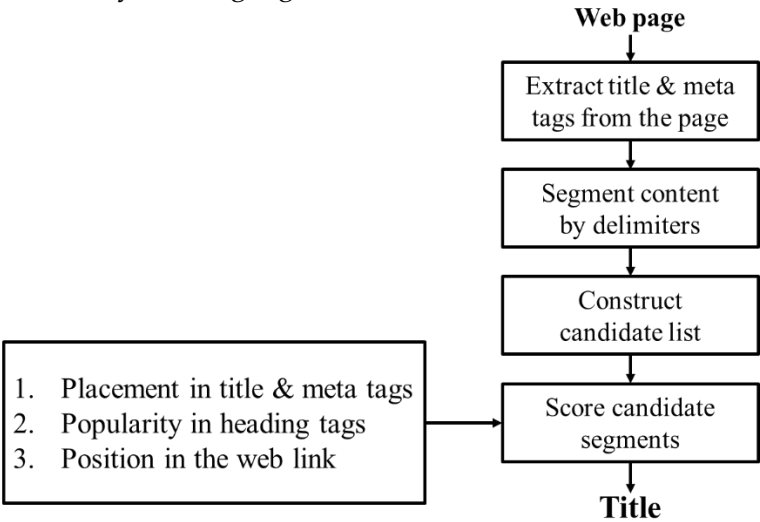


Figure 3.5: The workflow for TTA

The rationale for using the URL is that words in a web page link are precise and usually relevant to that page’s content. We assume a segment that appears in the document name (*index.html*) is more related to the content of a page than one that appears in the host (*www.example.com*). The same observation was also made by [56]. For example, consider the word ‘Kaspersky’ contained in the host of one link and document name of another. In the first case, we understand that the web page is located on Kaspersky’s web server but can relate to any topic. In the second case, the document is named ‘Kaspersky’ and is likely to discuss the company itself. We also assume that a segment with higher similarity to the words in the link is more important. For instance, consider the document name in the web link ‘Riga café’ and suppose that ‘Riga,’ ‘café,’ and ‘Riga Café’

are candidate titles. ‘Riga Café’ is judged to be more important, as it perfectly matches the document name.

We consider position in the tag, because according to a survey on search engine ranking factors undertaken in 2015 by MOZ,¹⁰ the key segment’s position in the title tag helps search engine optimization (SEO). The closer to the beginning of the tag a segment is, the more useful it is for ranking. It is also preferable to have the brand name at the end of the tag; for example, ‘<title> *Machine learning* | UEF </title>.’ Swapping the location of the key segment and the brand depends on the popularity of the brand in the target market. Given these factors, people are expected to follow the position rule when creating a title tag.

We tested all linear combinations for weighting the criteria, and the results indicate the URL is the most significant criterion. Experimental results in [P1] and [P2] show that our method provides 0.84 average similarity with the ground truth and outperforms the comparative methods used in [16, 78].

In contrast, *machine learning-based techniques* involve two steps: training and testing. In training, the goal is to learn a set of rules that maps inputs to outputs, so that the rules generalize beyond the training data [P2]. In testing, the generated classifier receives unseen data as input and predicts output values. Proper training of the model is the key to generalizing the classifier beyond the training data [77].

Researchers have considered several machine learning algorithms for title classification such as perceptron [64], decision tree (C4.5) [90], random forest [10], support vector machine (SVM) [51], and conditional random fields (CRF) [60]. While SVM has shown to be an effective classifier for the title extraction task, it has not been compared against simpler algorithms such as Naive Bayes [22], k-nearest neighbor (k-NN) [19], and clustering [29], all of which we investigate in [P2].

In [P2] we propose *Titler*, which uses a machine-learning technique (Figure 3.6) and is based on the following steps:

¹⁰ <https://moz.com/learn/seo/title-tag>

candidate phrase extraction (see Section 3.3), feature generation (see Section 3.4), phrase classification, and title selection.

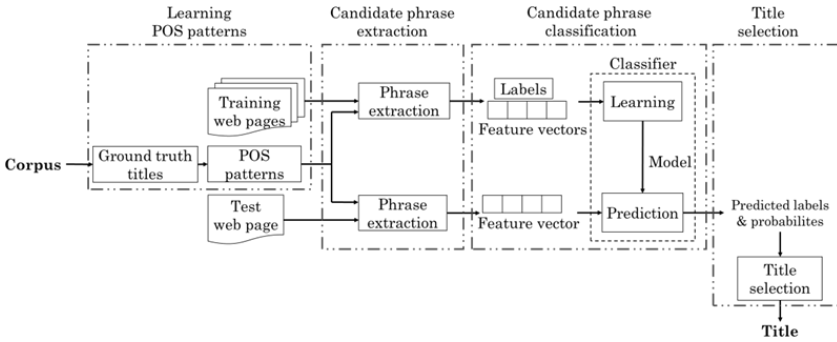


Figure 3.6: The workflow for Titler

Once the features for all candidate phrases are extracted, we classify the phrases as either *title* or *not-title*. We consider four alternative classifiers: Naive Bayes, clustering, k-NN, and SVM. Naive Bayes is a probabilistic model in which conditional probabilities are calculated from the training data and classification is done by taking the class with the highest probability given the features; it is easy to implement and usually yields good results [121]. The clustering model is a centroid-based classifier in which the model is trained by minimizing mean square error (MSE). A class label is assigned to each cluster based on the majority rule, and classification is done simply by mapping the input feature vector to its nearest centroid. The k-NN classifier is even simpler; it finds its k-nearest vectors in the training data and selects the class label using the majority rule. Finally, SVM is a binary classifier that attempts to find the best separation line between two classes using the training data. Classification is then performed by mapping the input vector into the feature space and selecting the class label of the side into which the vector falls. We choose the candidate phrase with the highest probability to reside in the title class as the title.

The results of our experiments using Titler corpus and Mopsi services data sets¹¹ show that our methods outperform the comparative methods by a large margin (see Tables 3.2 and 3.3).

Table 3.2: Comparative results: Titler corpus data.

Method	Rouge-1			Jaccard	Dice
	Precision	Recall	F-score		
Baseline	0.41	0.89	0.52	0.50	0.58
Google (2016)	0.48	0.89	0.58	0.56	0.64
TitleFinder [78]	0.43	0.61	0.47	0.43	0.50
Styling [16]	0.36	0.41	0.35	0.38	0.43
TTA [P1]	0.73	0.80	0.75	0.75	0.78
Titler (BAYES)	0.46	0.40	0.42	0.64	0.70
Titler (CLUS) [P2]	0.87	0.81	0.82	0.82	0.86
Titler (KNN)	0.87	0.82	0.83	0.84	0.87
Titler (SVM)	0.88	0.84	0.85	0.85	0.88

Table 3.3: Comparative results: Mopsi services data.

Method	Rouge-1			Jaccard	Dice
	Precision	Recall	F-score		
Baseline	0.33	0.71	0.41	0.44	0.54
Google (2016)	0.34	0.74	0.43	0.46	0.56
TitleFinder [78]	0.35	0.47	0.37	0.37	0.43
Styling [16]	0.14	0.21	0.15	0.22	0.28
TTA [P1]	0.52	0.59	0.52	0.54	0.62
Titler (k-NN) [P2]	0.59	0.56	0.55	0.59	0.66

In addition, we show that although the SVM model achieved the highest F-score (0.85), its differences to the k-NN (0.83) and clustering (0.82) models are insignificant. Both k-NN and clustering are simpler to implement and easier to understand in comparison to the theory that underlies SVM.

3.6 SIMILARITY MEASURES FOR TITLE MATCHING

Although several methods exist for extracting a title, little attention has been given to how to match title phrases. Titles (or

¹¹ <http://cs.uef.fi/mopsi/MopsiSet/>

strings) can be semantically or syntactically similar. Strings are *semantically equal* if they have exactly the same meaning (such as ‘car’ and ‘auto’) and *syntactically similar* if they have exactly the same character sequence. Semantic similarity can be measured using information from large corpora (i.e., be corpus based) [75], information from semantic networks such as WordNet (i.e., be knowledge based) [13], or a combination of both [75]. On the other hand, syntactic similarity is determined by measures operating on the sequences of strings and strings character compositions.

Given two titles, a useful similarity measure should be able to determine whether titles represent the same entity. Measures are used to evaluate the accuracy of the extracted title in [P1] and [P2]. In this section, we focus on syntactic similarity.

3.6.1 String segmentation

A *string* can be viewed as a stream of characters or as segments. Two approaches to segmenting strings exist: overlapping and non-overlapping. *Overlapping* divides a string into substrings of length q (q -grams); for example, substrings of length 2 are called *bigrams* and length 3 are called *trigrams*. The rationale behind the q -grams is that the sequence of the characters is more important than the characters alone. The q -grams for a string s are obtained by sliding a window of length q over the string (see Table 3.4). In contrast, *non-overlapping* is known as *tokenization*. It breaks a string into words and symbols (called tokens) using whitespace, line breaks, and punctuation characters (see Table 3.4). Similarity measures can operate at character sequence and any of these two levels of segmentation. In the following, we refer to these levels as being at the character level (when no segmentation is applied), the q -grams level, or the token level.

Table 3.4: The segmentation of strings.

Segmentation method	Output
None (character sequence)	The club at the Ivy
Overlapping (3-grams)	'the,' 'hec,' 'ecl,' 'clu,' 'lub,' 'uba,' 'bat,' 'att,' 'tth,' 'the,' 'hei,' 'eiv,' 'ivy'
Non-overlapping (tokens)	'The,' 'club,' 'at,' 'the,' 'Ivy'

3.6.2 String matching

Strings can be matched at the character, q-grams, or token level (see Table 3.5). The character and token matching are explored further below; q-grams matching is not addressed as it uses the same techniques as token matching.

Character level

On a *character level*, matching is done by transformation or by using the longest common substring (LCS). Transformation can be achieved in a number of ways. One example is *edit distance*, which is the minimum number of edit operations needed to transform a string s to a string t ; the edit operations include insertion, deletion, and substitution. The most common approach to computing the edit distance is dynamic programming. Variations of the edit distance have been proposed depending on the number, type, and cost of the operations, including *Levenshtein* [61], *Damerau-Levenshtein* [20], *Needleman-Wunsch* [80], *Smith-Waterman* [94], and *Smith-Waterman-Gotoh* [35].

Other examples of transformation are Hamming distance, Jaro distance, and Jaro-Winkler distance. *Hamming distance* allows only substitutions, and the length of the strings must be equal. *Jaro distance* [48] and its variant the *Jaro-Winkler distance* [112] are based on the number of matching and transposed characters. Characters are matched if they are the same and located no farther than $\lceil \max(|s|, |t|)/2 \rceil - 1$ within the string and transposed if they are the same but in reverse order (e.g., a-u, u-a). The Winkler distance also uses the length of the longest common prefix between the two strings up to four characters (see Table 3.5).

In contrast, the *LCS* [30] calculates the longest contiguous sequence of characters that co-occur in two strings. The result is normalized by dividing this sequence by the length of the longer string.

Character-level matching is useful for matching strings that contain only a few typographical errors, but it cannot detect the ordering of entire words. For example, it fails to capture the

similarity between 'Café Manta' and 'Manta café.' Token-level matching aims to compensate for this problem.

Token matching

Methods for token matching involve two challenges: which tokens to match and how to perform the matching.

Token matching uses three possible representations: array, set of tokens, or bag-of-tokens. Each representation requires different processing and offers varying performance in measuring similarity. All three are discussed below.

An *array* is constructed from all tokens in a string; as such, duplicates are allowed. Matching is performed by searching for a match for each token in s , in the list of t . If more than one match exists, the token is counted only once. This type of matching is asymmetrical; for example, in Figure 3.7a (left), three tokens in s match two tokens in t . However, only two tokens are matched when s and t swap content, as illustrated in Figure 3.7a (right). String similarity is obtained by dividing the results by the length of the longest string, for example. To make the measure symmetric, the similarity results in both directions are usually combined; see the Rouge-N equation [62] in Table 3.5.

A *set of tokens* is a collection of distinct tokens in a string. Similarity is obtained by counting the number of tokens that two strings have in common and then dividing that number by the number of tokens in the longest string (for the *matching coefficient*), the shortest string (for the *overlap coefficient*), the total number of unique tokens in both strings (for the *Jaccard index*), or the total number of all tokens in both strings (for the *Dice coefficient*) [11]; see Figure 3.7b.

A *bag-of-tokens* combines the distinct tokens in two strings. A *feature vector* for each string is then generated using presence, frequency, or the term frequency-inverse document frequency of each token in the corresponding string. *Presence* is the occurrence of the tokens within a string (1 = occurrence; 0 = otherwise). *Term frequency* ($TF_{w,s}$) counts the occurrences of a token w in s ; it favors tokens that appear frequently in a string. *Term frequency-inverse document frequency* ($TF-IDF$) is the product

of two statistics: the token frequency and its inverse document frequency (IDF_w). The latter is the total number of strings (str) divided by the number of strings that contain w . Metrics such as *Euclidean distance* [68], *Manhattan distance* [68], and *cosine* [17] are then applied to compute the similarity between the feature vectors (see Figure 3.7c).

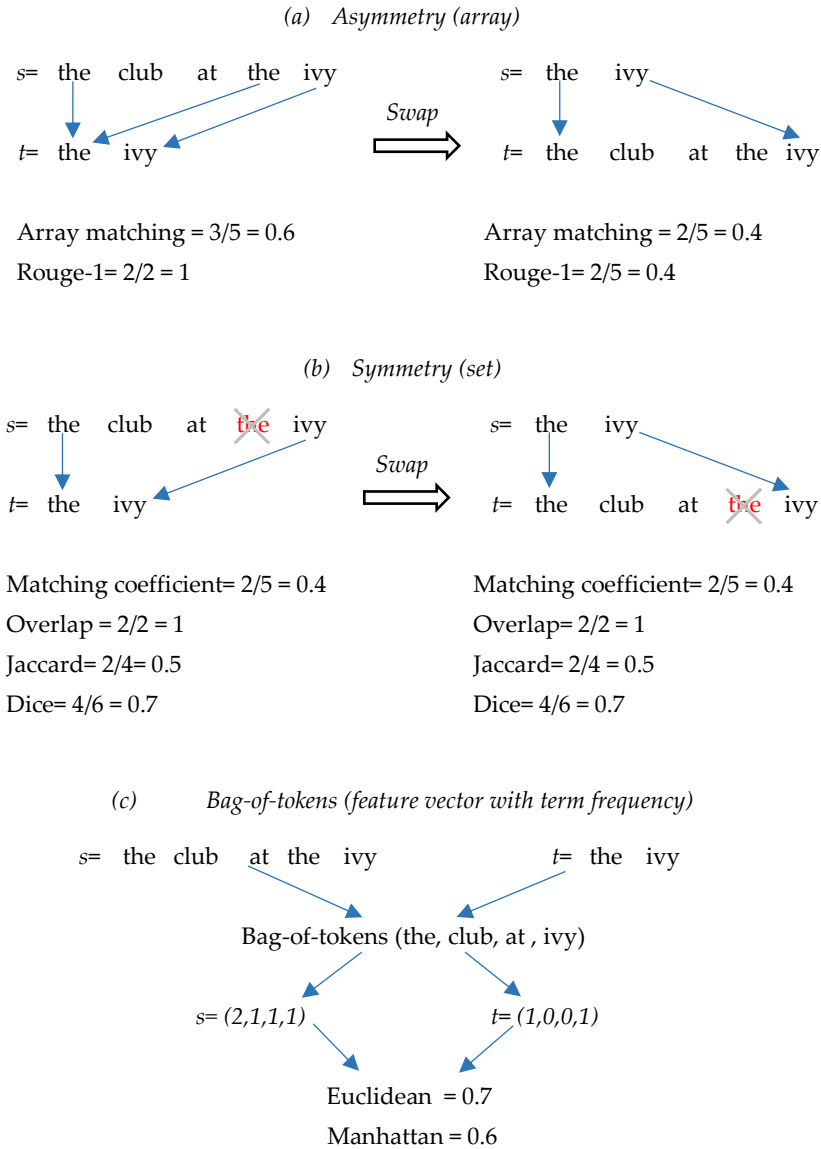


Figure 3.7: Examples of string exact matching at the token level

Token similarity

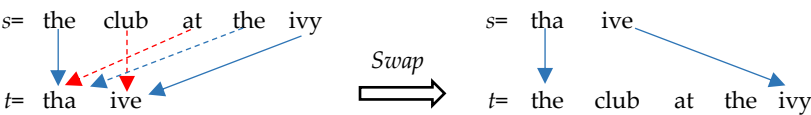
Two alternatives for the token matching exist: exact and approximate. *Exact matching* provides a simple binary result: 1= if the tokens are exactly the same; 0= otherwise. *Approximate matching* computes the similarity between tokens using a character or q-grams measure and estimates the similarity within the range [0, 1]. Approximate matching can also be used for comparing tokens that are similar but not exactly identical (e.g., ‘color’ and ‘colour’).

In Figure 3.7, similarity is computed using exact matching. In Figure 3.8, we add artifacts to the string ‘the ivy’ and use Smith-Waterman-Gotoh as the secondary (character-level) measure. We apply a similarity threshold of 0.7 and take the matching coefficient (MC) and Monge-Elkan (ME) as examples. Monge-Elkan [79] computes the similarity between two strings by taking the average score of the best matching tokens.

Matching between all combinations of tokens using Smith-Waterman-Gotoh

Sm-Wa-Go (the, tha)	= 0.9	Sm-Wa-Go (at, ive)	= 0.3
Sm-Wa-Go (the, ive)	= 0.3	Sm-Wa-Go (the, tha)	= 0.9
Sm-Wa-Go (club, tha)	= 0.2	Sm-Wa-Go (the, ive)	= 0.3
Sm-Wa-Go (club, ive)	= 0.4	Sm-Wa-Go (ivy, tha)	= 0.2
Sm-Wa-Go (at, tha)	= 0.5	Sm-Wa-Go (ivy, ive)	= 0.7

String matching



MC (exact) = 0/5 = 0	MC (exact) = 0/5 = 0
MC (approx.) = 2/5 = 0.4	MC (approx.) = 2/5 = 0.4
ME = 3.4/5 = 0.7	ME = 1.6/2 = 0.8

Figure 3.8: Example of string approximate matching

Other examples of approximate matching measures include Soft-TFIDF [18], Soft-Jaccard, and Soft-Dice [74], which respectively combine Jaro-Winkler with TF-IDF, Jaccard, and

Dice. In Table 3.5 we review 21 existing measures that are classified according to the segmentation level, matching technique, and similarity method.

Table 3.5: Overview of 21 similarity measures. Components include segmentation level, matching technique, and similarity method.

The symbols s and t refer to the input strings. Symbol $|s|$ can refer to the length of the string in characters, size, or the length of the string in tokens, depending on the measure type. In Jaro, m is the number of matching characters and x is half the number of the transposed characters. Symbol p in Jaro-Winkler is a scaling factor fixed to 0.1, and l is the length of the common prefix up to four characters between strings. In cosine, symbol $|\Sigma|$ is the dimensionality of the feature vector and symbols s_i and t_i are vector components. In Monge-Elkan, $sim(s_i, t_j)$ is the Smith-Waterman-Gotoh between two tokens. Symbol w in TF-IDF refers to a token, $|str|$ is the number of strings to be compared, and z is the number of strings that contain w . In Soft-TFIDF, $sim(w, v)$ is the Jaro-Winkler between two tokens.

Name	Formula	Components
Levenshtein	$sim_{lev}(s, t) = 1 - \frac{Lev(s, t)}{\max(s , t)}$	Character; Transformation; Approx.
Damerau-Levenshtein	$sim_{dam}(s, t) = 1 - \frac{Dam(s, t)}{\max(s , t)}$	
Needleman-Wunsch	$sim_{nw}(s, t) = 1 - \frac{Ne - Wu(s, t)}{2 \times \max(s , t)}$	
Smith-Waterman	$sim_{sw}(s, t) = \frac{SW(s, t)}{\min(s , t)}$	
Smith-Waterman-Gotoh	$sim_{swg}(s, t) = \frac{SWG(s, t)}{\min(s , t)}$	
Hamming distance	$sim_H(s, t) = 1 - \frac{H(s, t)}{\max(s , t)}$	
Jaro	$J(s, t) = \frac{1}{3} \times \left(\frac{m}{ s } + \frac{m}{ t } + \frac{m - x}{m} \right)$	
Jaro-Winkler	$JW(s, t) = J(s, t) + (l \times p(1 - J(s, t)))$	Character; Longest substring; Approx.
LCS	$LCS(s, t) = \frac{LCS(s, t)}{\max(s , t)}$	
Trigrams	$Trigram(s, t) = \frac{ trigr(s) \cap trigr(t) }{\text{average}(trigr(s) , trigr(t))}$	
Matching	$MC(s, t) = \frac{ s \cap t }{\max(s , t)}$	

coefficient		Set; Exact, approx.
Overlap coefficient	$OC(s, t) = \frac{ s \cap t }{\min(s , t)}$	
Jaccard	$Jac(s, t)_{token} = \frac{ s \cap t }{ s \cup t }$	
Dice	$Dice(s, t)_{token} = \frac{2 \times s \cap t }{ s + t }$	
Rouge-N	$F(s, t) = \left(\alpha \times \left(\frac{1}{p} \right) + (1 - \alpha) \times \left(\frac{1}{r} \right) \right)^{-1}$ $\alpha = 0.5, \quad p = \frac{ s \cap t }{ s }, \quad r = \frac{ s \cap t }{ t }$	Q-grams, tokens; Array; Exact.
Cosine	$cos(s, t) = \frac{\sum_{i=1}^{ S } s_i t_i}{\sqrt{\sum_{i=1}^{ S } (s_i)^2} \sqrt{\sum_{i=1}^{ T } (t_i)^2}}$	Tokens; Feature vector; Exact, approx.
Euclidean	$1 - \frac{\sqrt{\sum_{i=1}^n (s_i - t_i)^2}}{\sqrt{ s ^2 + t ^2}}, \quad n = s \cup t $	
Manhattan	$1 - \frac{\sum_{i=1}^n s_i - t_i }{ s + t }, \quad n = s \cup t $	
Monge-Elkan	$ME(s, t) = \frac{1}{K} \sum_{i=1}^K \max_{j=1 \text{ to } L} sim(s_i, t_j)$ $K = s , L = t $	Tokens; Array; Approx.
TF-IDF	$sim_{tfidf}(s, t) = \sum_{w \in s \cap t} v(w, s) \times v(w, t)$ $v(w, s) = \frac{\dot{v}(w, s)}{\sqrt{\sum_{w \in s} \dot{v}(w, s)^2}}$ $\dot{v}(w, s) = \log(TF_{w,s} + 1) \times \log(IDF_w)$ $TF_{w,s} = N_{w,s}, IDF_w = \log \frac{ str }{ \{z \in str w \in z\} }$	Tokens; Feature vector; Exact.
Soft-TFIDF	$SoftTFIDF(s, t) = \sum_{w \in CLOSE(\theta, s, t)} v(w, s) \times v(w, t) \times d(w, t)$ $CLOSE(\theta, s, t) = \{w \in s \exists v \in t: sim(w, v) > \theta\}$ $\theta \geq 0.9, d(w, t) = \max_{v \in t} sim(w, v)$	Tokens; Feature vector; Approx.

We conducted six experiments using the Titler data set [P2] and Mopsi photo collection¹² [P3]:

- Text manipulation (character change, token change, token swap);
- Correlation to human judgments;
- Correlation to distance; and
- Clustering.

¹² <http://cs.uef.fi/mopsi/tools/photoclusters.php>

The results reveal that no measure outperforms others in all experiments. Character-level measures correlate best with human judgments and perform well for matching nearby photos, but they are outperformed by the token-level and soft measures in finding good clusters (see Table 3.6). Soft-TFIDF manages to combine the best properties of the character- and token-level measures. Since this measure works well with real data (e.g., Mopsi photos) and correlates with human judgments, it is a suitable candidate for our experiments.

*Table 3.6: Summary of the results of six experiments: *excellent*, *good*, and *poor*.*

	Text manipulation			Titler	Mopsi photo	
	Char. change	Token change	Token swap	Corr. to human	Corr. to distance	Clustering
Character-level						
Levenshtein	0.36	0.07	0.39	0.59	76	0.11
Damerau- Leven.	0.02	0.07	0.39	0.59	76	0.11
Needleman- Wu.	0.02	0.28	0.62	0.56	77	0.11
Smith-Waterman	0.02	0.05	0.51	0.25	60	0.04
Smith-Wa- Goto.	0.28	0.04	0.57	0.25	63	0.07
Hamming	0.20	0.05	0.14	-	61	0.19
Jaro	0.23	0.16	0.58	0.45	71	0.16
Jaro-Winkler	0.13	0.22	0.60	0.39	71	0.14
LCS	0.36	0.07	0.43	0.56	66	0.11
Q-grams						
Bi-Jaccard	0.15	0.10	1.00	0.52	68	0.09
Bi-Dice	0.37	0.01	1.00	0.47	68	0.11
Trigrams	0.35	0.02	0.75	0.58	69	0.05
Token-level						
Matching	0.40	0.00	1.00	0.52	65	0.12
Overlap	0.36	0.00	1.00	0.21	64	0.08
Jaccard	0.26	0.08	1.00	0.53	64	0.07
Dice	0.16	0.00	1.00	0.45	64	0.06
Rouge-1	0.11	0.09	1.00	0.47	66	0.09
Cosine	0.31	0.00	1.00	0.44	64	0.10
TF-IDF	0.35	0.08	1.00	0.47	64	0.09
Euclidean	0.37	0.23	1.00	0.51	84	0.08
Manhattan	0.36	0.00	1.00	0.45	64	0.06
Soft measures						
Monge-Elkan	0.24	0.19	1.00	0.50	60	0.08
Soft-TFIDF	0.15	0.08	1.00	0.51	65	0.07

4 *Extracting Keywords*

Keywords, which are the smallest units to express the meaning of a text, summarize a web page's content in a few select words [83]. They are used in several tasks, such as topic detection, text summarization, and web page indexing. In topic detection, for instance, each page in a collection of web pages is represented by its keywords. The keywords are then clustered using algorithms such as k-means, where centroids describe the topics of the clusters [109]. Another approach is to cluster pages by matching keywords and the top-ranked keywords define the topic of the cluster [38]. According to [55], using keywords to cluster pages produces better results, as it prevents misclassification due to irrelevant data.

Web page indexing is another example of a task in which keywords play a major role. Indexing is the process of converting a page into a set of words called the index language [54]. Not all words serve as useful indices for a web page. For instance, some only have a grammatical role, such as conjunctions and particles (i.e., stop words); others (e.g., advertisement words) are irrelevant to a page's content. Keywords that are indeed related to content should be considered in the index. For example, in a news website that covers several topics (such as sports, politics, society, and information technology), the keyword 'software' may be crucial for categorizing information technology. However, in the case of a website that covers specific areas within information technology, the keyword 'software' is too general. In addition to assisting in indexing, keywords support search engines' work in relation to information retrieval by preventing non-relevant pages from being retrieved [54].

Scientific publications contain a list of keywords that have explicitly been assigned by their authors. However, most other documents have no keywords assigned to them [118]. In web

pages, the keywords meta tag allows authors to include keywords in their pages. However, this tag is often abused to bias search engines,¹³ as people usually fill it with a long list of keywords that are barely relevant to a page's content to increase the chance that their page will rank high on the search results page. This is demonstrated in the following example.¹⁴

```
<meta name="keywords" content="lodge tapas, lodge lane
cove, genuine tapas restaurant, delicious freshly cooked
tapas, delicious taste, vegetarian and gluten free
recipes, fully licensed bar, fully licensed restaurant,
great selection of beers, wines, cocktails, licensed
tapas bar at lane cove, restaurant burns bay road, lane
cove, nsw, taste lots of different dishes, family favorite
restaurant, tapas bar in Sydney, tapas restaurants Sydney,
lodge tapas lane cove, fun, friendly place for breakfast,
lunch, dinner, parties, anniversaries, place to celebrate
occasions, authentic worldwide cuisine, delicious
Australian food at lane cove, lane coves favorite places
for people, join us for authentic tapas cuisine,
experience great freshly made tapas, we cater for
functions, design your own menu for function, customize
functions menu, tapas bar lane cove, nsw, Spanish
restaurant at lane cove, nsw, tapas restaurants Sydney,
Australian restaurant, quality food and bar at cove lane
nsw, mid-week specials, enjoy our happy hours, beer, wine
& cocktails, lunch special menu"/>
```

Manually assigning keywords is a labor-intensive, time-consuming, and error-prone task. As a result, three automatic techniques have emerged: keyword extraction, keyword assignment, and keyword indexing [52].

In *keyword extraction*, words are extracted from the body of the page, filtered, and ranked according to statistical and linguistic properties (e.g., term frequency, semantic relatedness to other words in the web page, position, phrase length, and word co-occurrence). The ranking function is either *supervised* or *unsupervised*. In supervised ranking, keyword extraction is viewed as a classification task in which a word on a page is either a keyword or not [114, 118]. In unsupervised ranking,

¹³ <https://webmasters.googleblog.com/2009/09/google-does-not-use-keywords-meta-tag.HTML>

¹⁴ <http://lodgetapas.com.au/>

candidate words are sorted and ranked using techniques such as simple statistics (e.g., TF, TF-IDF) [95], clustering [36, 63, 86, P4], and graph-based ranking [8, 76].

In *keyword assignment*, keywords are not limited to the words that appear on a page; they are instead chosen from controlled vocabularies such as thesauri, predefined taxonomies, or controlled index term sets (e.g., Medical Subject Headings) [52]. Keyword annotation is treated here as a multi-label text classification task for which the keywords from relevant vocabularies are treated as classes. Given a keyword in the controlled vocabulary, machine-learning algorithms such as SVMs, Naive Bayes, and decision trees are used to learn a model for it from a set of training web pages that are manually labeled. The web pages are represented by feature vectors with binary elements corresponding to the presence and absence of the words in a document. The learned models are then applied to the unseen pages for classification, which generates a set of classes (i.e., keywords) for each web page [21]. Keyword assignment is not restricted to the words appearing on a web page; however, it depends greatly on the characteristics of the target domain (such as the domain's vocabulary). It may therefore suffer from a lack of high-quality or -quantity training data [32]. In some cases, no matching word may even be available for a web page's content. A final challenge related to keyword assignment is that the controlled vocabularies also need to be constructed and maintained.

An approach that falls between the above two techniques is *keyword indexing*, which was proposed in [71, 72]. In this approach, candidate keywords have to satisfy two criteria: those keywords (or their synonyms) must occur in a page's text and should be names commonly used to represent concepts in a domain-specific thesaurus. For example, as the preferred name for 'computer programme' is 'software' according to the Multilingual Thesaurus of the European Union, the appropriate keyword would be 'software'. In contrast to keyword assignment, keyword indexing does not require a set of training data for each label. However, it assumes that a comprehensive

thesaurus is available for a specific domain, which is not always the case.

Based on the relative strengths and weakness of these three techniques for automatically extracting keywords, the rest of this chapter focuses solely on keyword extraction. Existing methods for keyword extraction often undergoes two steps: *candidate keyword extraction* and *scoring* [108].

4.1 CANDIDATE KEYWORD EXTRACTION

Standard text documents are often presented in one layout, with the title, abstract, and main contents presented sequentially. In contrast, text is scattered all over on a web page and the formatting differs, which makes it more difficult to analyze the page's content [P4]. Moreover, web pages contain irrelevant text, such as advertisements, navigation menus, and hyperlinks. The amount of this information can be extensive compared to the main text, which makes the task of keyword extraction more challenging.

Candidate keyword extraction usually involves several steps, such as text extraction, cleaning, segmentation, POS tagging, normalization, filtering, and other heuristics [108]. Table 4.1 summarizes which of these steps are used by the existing methods found in the literature (and in what order). Thereafter the steps are discussed in greater detail.

The *text extraction* step has not been explored widely in the literature. Previous studies have not explicitly mentioned the techniques used to extract useful text from a web page. Furthermore, web pages converted from the original published format to plain text are input to keyword extraction methods [36, 70]. Our challenge in [P4] was to process text directly from a web page in the presence of irrelevant data without using intermediate main content extractors, as such extractors are usually domain dependent [58].

Table 4.1: Typical steps for candidate keyword extraction. The acronyms used are as follows: TXT= text extraction; CL= cleaning; TOKN= tokenization; SPLT= splitting text by punctuations marks or stop words; POS= POS tagging; NORM= normalization; FILT= filtering by n-grams with predefined rules or by POS patterns; OTHER= other heuristics (e.g., bold, big, image alt, and title text).

Method	Processing sequence
Aquino et al. [5]	SPLT, FILT
Bracewell et al. [9]	TOKN, POS, NORM, FILT
Dostal and Jazek [23]	TOKN, CL, POS, FILT
Humphreys [43]	TXT, OTHER, CL
Mihalcea and Tarau [76]	TOKN, POS, FILT
Rose et al. [92]	CL, SPLT, TOKN
Zhang et al. [118]	SPLT, TOKN, POS, FILT, NORM
Zhang et al. [120]	TXT, TOKN, NORM, FILT, CL
[P4]	TXT, CL, TOKN, POS, FILT, NORM

Text cleaning involves identifying and possibly removing words and characters that carry little meaning to a web page's text. Less relevant words and characters include stop words such as particles, prepositions, conjunctions, pronouns, and special verbs (e.g., can, may, should, would); symbols; and punctuation marks. The method in [87] applies heuristic rules to remove tables and figures from the text.

In **[P4]** text extraction and cleaning are processed as follows: we start by downloading the HTML source of a web page and parsing it as a DOM tree. We then remove script and styling tags, as their content is mainly used for styling. Thereafter we extract the text nodes by XPath and clean symbols such as &, £, and \$ and digits such as 1, 2, and 3 from the text. We subsequently compute the length (i.e., number of tokens) of each text node: if it is less than 6 followed by a text node of the same length or less, the text of the preceding node is deleted. This step ensures that most of the navigation menu elements, formatting objects, and functional words are not considered as a part of the text we extract. For example, in Figure 4.3, the token 'home' is followed by 'treatment menu'; as a result, 'home' is deleted. However, 'Our Journey' is followed by a longer text, namely 'we want to make Forme Spa & Wellbeing your happy place...'; as such, 'Our Journey' remains.



Figure 4.3: Filtering irrelevant text from a sample page¹⁵

Text segmentation divides a text into either tokens (using white space) or segments (using punctuation marks and stop words that were identified in the cleaning step).

Normalization aims at converting a text into a more convenient format to enable efficient filtering. For instance, 'apple,' 'Apple,' and 'apples' should be identical after normalization. This process includes conversion to lowercase, stemming, and lemmatization. In *stemming*, inflected words are reduced to their common stem. The *stem* is the part of a word that is left after removing its prefixes or suffixes; for example, 'care,' 'careful,' and 'cares' are all reduced to 'care.' However, stemming might fail and return words with no meaning [39]. For example, the stem of 'introduce,' 'introduces,' and 'introduced' is 'introduc,' which does not mean anything on its own. On the other hand, *lemmatization* aims at removing a word's inflectional endings and returns base forms as found in the dictionary (which is also known as a lemma) utilizing vocabulary and morphological analyses of words [4]. It is useful when counting the frequency of words in a web page. For example, in lemmatization, the plural 'mice' can be transformed to the singular 'mouse,' even though the stem of 'mice' is 'mice.' In [P4], we measure the semantic similarity between nouns

¹⁵ <http://www.formespa.co.nz/site/webpages/general/our-journey>

using WordNet [73]; we choose lemmatization instead of stemming to avoid meaningless words being return by the stemmer.

Filtration leaves only candidate keywords. Two common techniques are *n*-grams with heuristic rules and predefined POS patterns. The *n*-grams are all possible sequential combinations of words up to length *n* [108]. For instance, the three-word text segment ‘image processing unit’ produces six combinations: ‘image processing unit,’ ‘image processing,’ ‘image,’ ‘processing unit,’ ‘processing,’ and ‘unit.’ Heuristic rules are then applied to choose the candidate keywords; for example, the frequency of candidates in a text must be above a predefined threshold [81].

Finally, *POS patterns* are manually constructed based on an analysis of the manual keywords present in the training data under study. Words and sequences of words that match any of the constructed patterns are extracted as candidates; a common pattern is an adjective followed by nouns [117]. According to [42], a majority of the keywords manually assigned by humans are either nouns or noun phrases. We thus use the Stanford POS tagger to extract nouns as potential candidate keywords in [P4].

Although we use the POS patterns technique to extract titles as described in Section 3.3, the patterns generated are different from the patterns created for keywords. In title extraction, patterns can also contain items other than nouns or adjectives. For example, ‘<VB><CC><VB>’ is a pattern allowed for a title such as *Slice and Dice* (where CC stands for coordinating conjunction). Stop words are also included in the pattern to produce phrases that are grammatically correct and understandable to humans. In keyword extraction, priority is given to words that carry significant meaning; as a result, keyword patterns mainly involve nouns.

4.2 CANDIDATE KEYWORD SCORING

The method for scoring candidate keywords can be divided into two classes: *supervised* and *unsupervised*. An overview of these scoring classes is presented in Figure 4.4.

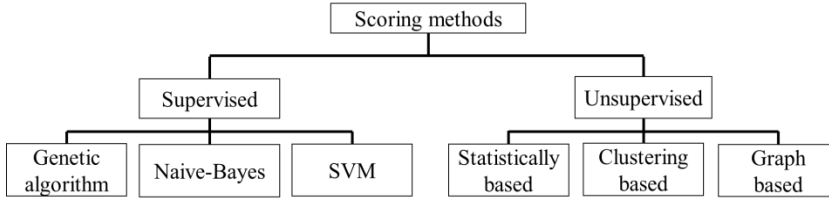


Figure 4.4: Scoring methods for keyword extraction

Supervised methods employ classifiers such as Naive Bayes and SVM and features such as TF-IDF, POS tag, first appearance in the page, and word position to classify words as either keywords or not [27, 101, 114]. They are lacking in two ways: first, they require training data with manually annotated keywords (which is not always available, especially for web pages) [86]; second, they are biased toward the domain in which they are trained.

Unsupervised methods can be classified into three groups: statistically based, graph based, and clustering based [108]. These methods are further discussed in the following paragraphs.

Statistical methods assign scores to candidate keywords by applying different measures to the input text. Popular measures include term frequency, TF-IDF (see Section 3.6.2), and word co-occurrence.

Word co-occurrence can be determined within either a window of N words [76] or a set of candidate phrases [92]. For instance, in [92], the text is first segmented into candidate phrases using stop words and delimiters. A word's frequency ($TF(w)$) and its co-occurrence frequencies with other words in the list including itself ($degree(w)$) are then computed. The following three candidate phrases can be used as an example:

- Minimal set
- Minimal generating sets
- Minimal support set

The unique words in these phrases are ‘minimal,’ ‘set,’ ‘generating,’ ‘sets,’ and ‘support.’ The frequency of the word ‘minimal’ and its co-occurrence with other words are as follows:

	minimal	set	sets	generating	support
minimal	3	2	1	1	1

Therefore, $TF(\text{minimal}) = 3$ and $degree(\text{minimal}) = 8$. The score of a word $w \in W$ is the sum of the ratio of degree to frequency of each word:

$$Score(w) = \sum_{w \in W} \frac{degree(w)}{TF(w)} \quad (4.1)$$

Relying only on statistical information is insufficient for selecting important words, as it lacks the ability to identify semantically related words such as synonyms. Synonyms are words that carry exactly or nearly the same meaning but do not share the same character sequence; for example, ‘toy’ and ‘doll’.

Graph-based methods represent text as a graph with words as the vertices and edges as the relationship between the words. The relationship can be term frequency, word co-occurrence, or semantic similarity between words. The vertices are then ranked according to different algorithms, such as Google’s PageRank [12] and hyperlink-induced topic search (HITS) [57]. For example, TextRank [76] uses word co-occurrence and PageRank by implementing the idea of voting. A link pointing from vertex i to a vertex j means that V_i casts a vote for V_j . The importance of V_j is then measured by the number of votes it gets. Furthermore, the importance of the vote itself is also considered. The more important V_i is, the more important the vote it generates. The number of votes a vertex earns is local information, while the

importance of the voters is global information, which is recursively drawn from the entire graph.

Clustering-based methods aim at grouping words that are either syntactically or semantically related. For example, in [9], two noun phrases are grouped in one cluster if they have a word in common (e.g., the noun phrases ‘stem cell’ and ‘stem cell research’ are clustered together because they both use ‘stem cell’). In [63], the semantic relatedness between words is utilized in clustering. Three algorithms are considered: hierarchical, spectral, and affinity propagation.

There are two different approaches to measuring the importance of words. In one approach [9], clusters are first ranked according to the size or sum of the average frequencies of the words within each cluster. The words with the shortest length or highest frequencies are then selected from the top clusters. In another approach [63], keywords are chosen from each cluster; for instance, the centroids of the clusters are considered keywords.

Previous scoring methods may fail when applied immediately to web page content due to the diverse content and semi-structured nature of web pages in general. For example, in [P4], we applied the graph-based method (i.e., TextRank) on several web pages and found that the extracted keywords are less relevant to the pages’ topics compared to the keywords extracted by term frequency. Term frequency performs better after a pre-processing step such as the removal of stop words. One reason is the heterogeneous structure of the web pages we study. Second, in most web pages, important words are emphasized by repetition. Third, human tends to select words that appear several times on a page. However, some web pages use synonyms to emphasize meaning, and in such cases term frequency fails. In [108], experimental results show that statistically based methods such as TF-IDF and word co-occurrence perform better than graph-based methods, with different preprocessing steps.

Our method in [P4] benefits from using the semi-structured environment, i.e., the location of words in the DOM tree, to

judge the importance of words. Our solution combines several techniques to ensure that a web page's topic is well covered and we are not focusing on only a specific domain. These techniques include the following:

- POS tagging to extract nouns;
- Clustering to group semantically related nouns together;
- Distributing nouns over the DOM text nodes to ensure a high coverage of the web page's topic; and
- Using term frequency to select relevant keywords within a cluster.

4.3 CLUSTER RANK

In [P4], we introduce cluster rank (*Clrank*) to extract keywords (see Figure 4.5). After the pre-processing step discussed in Section 4.1 (see Table 4.1), we cluster the nouns according to their semantic relatedness in WordNet [73]. Nouns that do not exist in WordNet or match any word from a predefined list of stop words (e.g., 'yesterday') are omitted from the list of candidates. We use hierarchical clustering because the number of clusters can be controlled by simple thresholding. The clustering ends when the similarity between the next two clusters to be merged is less than a predefined threshold.

We score the clusters by counting the number of DOM text nodes in which the candidate nouns from a cluster appear. The counts of unique nouns are summed up to obtain each cluster's score. For example, suppose that one cluster contains 15 nouns from only one text node and another cluster contains 10 nouns from five different nodes. The smaller cluster is more important because its content is distributed wider within the page. In the event that two clusters get the same score (i.e., a tie case), the larger cluster is ranked first.

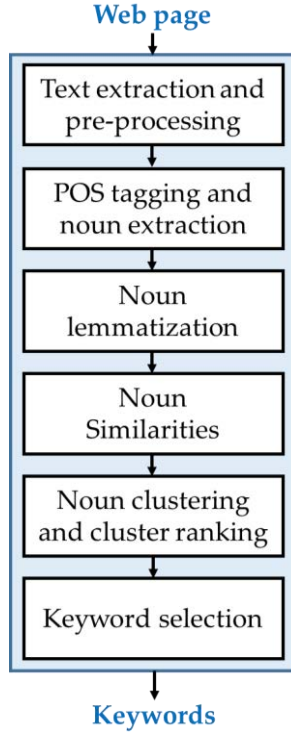


Figure 4.5: Workflow for Clrank

We also eliminate clusters of a very small size using the following rule, as they typically contain advertisement words (e.g., 'reward,' 'credit,' and 'bonus'):

$$Size < 0.2 \times maxClusterSize \quad (4.2)$$

We select the most frequent nouns from each cluster and in the case of a tie consider the average similarity to all other nouns in the same cluster as a secondary criterion. This favors nouns that are more central in the cluster.

The number of keywords selected from each cluster depends on the noun's frequency. The next keyword from the same cluster is selected only if its frequency meets the following two conditions:

$$\begin{aligned} Frequency &> 0.2 \times maxFrequency \\ Frequency &> 3 \end{aligned} \quad (4.3)$$

Where *maxFrequency* is the maximum frequency found in the same cluster. The thresholds are chosen empirically. Similarly to [83], the maximum number of keywords selected from each web page is limited to 10.

For example, in Figure 4.6, 'Spa' appears 38 times in total in 23 out of 37 different DOM text nodes while 'building' appears in one node; the score of this cluster is therefore 24. In Figure 4.6, only the first three clusters are considered, as the fourth cluster no longer meets the size criteria. The most important keywords of 'spa,' 'treatment,' and 'Auckland' are selected first from the significant clusters. The noun 'massage' is then selected as a secondary choice, since the maximum number of keywords (10) has not yet been reached and its frequency is above the threshold (3).

In Figure 4.6, we observe that the method finds three correct keywords (43%), misses four (57%), and selects one keyword that is not considered as a correct choice by a human.

Our empirical analysis of web pages from five different domains (namely education, news, tourism, beauty and fitness, and food and drink) shows that complete linkage works better than average linkage mainly due to the semantic similarities between the nouns. In the average linkage, different nouns can be incorrectly grouped in the same cluster. In the complete linkage, the similarity between every two nouns in the cluster is above the threshold.

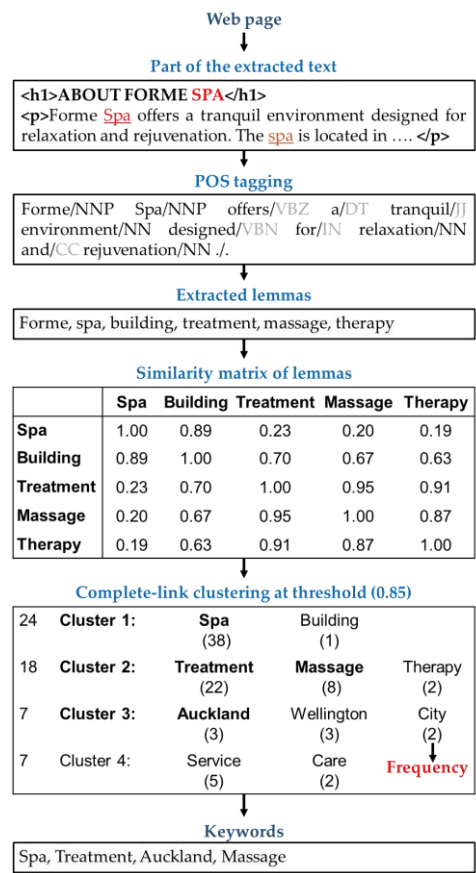


Figure 4.6: Corresponding results for Figure 4.5

Our findings suggest that the distribution of nouns is a more informative feature than term frequency. When all nouns are grouped into one cluster (merge threshold = 0), keywords are selected based merely on their frequency in the web page (F-score = 0.42). However, when every noun forms its own cluster (merge threshold = 1), the scoring depends on the nouns' distribution over the page and thus improves (F-score = 0.46). The best F-score is recorded at threshold 0.95, where highly similar nouns such as 'treatment' and 'massage' are grouped together. In Figure 4.7, we also observe that CIRank outperforms the F-score of term frequency and the state-of-the-art (i.e., TextRank) by 6 percentage points and 10 percentage points, respectively.

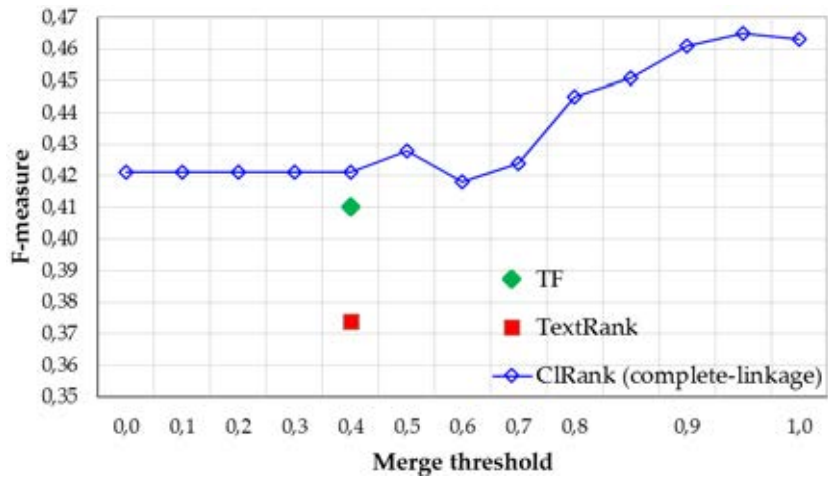


Figure 4.7: Comparative results for term frequency (TF), TextRank, and CLRANK

5 Extracting Images

After text, the *image* is the most popular media type on the web [3]. Images are used in web pages due to their ability to communicate information instantly; indeed, the human brain can interpret images faster than it can texts. A recent study by neuroscientists Potter et al. [88] from the Massachusetts Institute of Technology has shown that the human brain can process an entire image in 13 milliseconds. Moreover, images grab users' attention and help them to engage with content, which affects how long they stay on a web page. They also help users to remember visited pages' content.

A web page typically contains many images, a majority of which are not highly relevant to the main content. Images on web pages are used for different purposes (see Figure 5.1), including formatting objects, advertisements, navigational banners, and section headings [7].

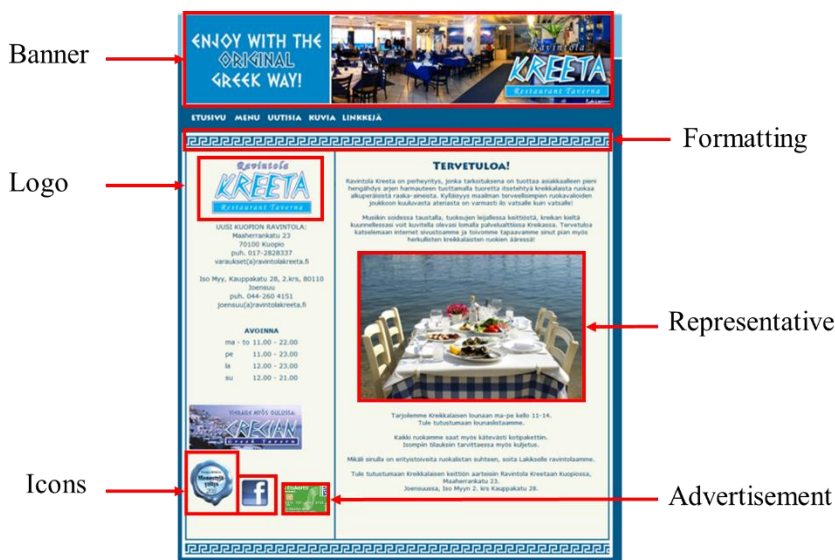


Figure 5.1: Types of images in an example web page

In this work, we are interested in images that clearly describe objects or scenery related to a page's main content. We refer to these images as *representative images*, and a web page may contain many of them [P5].

A representative image is useful in different applications, such as when bandwidth limitation restricts the total number of images that can be retrieved or when building a visual category in which a single image must represent an entire category of documents and their associated content [40]. In the results page of a search engine, showing an image along with an existing textual snippet enriches the contextual description [98]. When text snippets are long, users find them time-consuming to read; when they are too short, however, they do not convey enough information about a web page's content. A representative image is helpful in both situations, since it allows users to rapidly determine the relevance of search results for a broad query. For instance, Google News¹⁶ presents an image alongside a textual snippet, which lets users quickly identify the desired web page (see Figure 5.2).

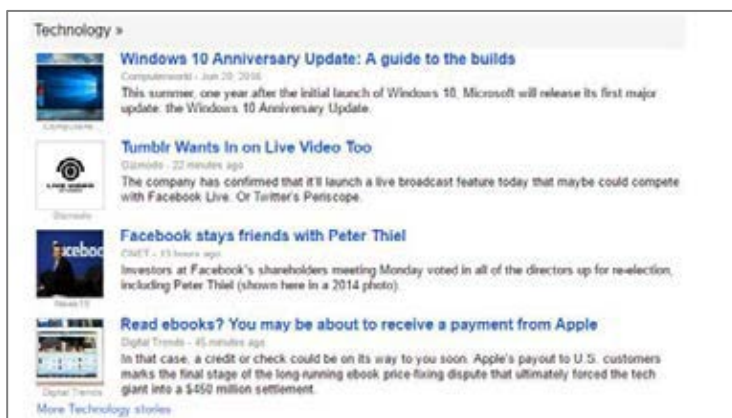


Figure 5.2: Textual snippet with representative images

Representative images are important for location-based applications such as Mopsi, where simple thumbnails with titles are displayed to users. They are also used by social networking

¹⁶ <https://news.google.com/?ar=1466503936>

platforms such as Facebook and Google+ when users share web page links on their wall.

Several researchers have been working on tasks such as extracting relevant images from specific parts of a web page like article block [1, 53], identifying and eliminating advertisement images [85], finding images relevant to a user's query [100], extracting multiple useful images from a web page [25] or collection of web pages [84], and representing a web page by a scaled-down snapshot as rendered in a browser [113]. Our goal is to select a single image that best represents a web page's entire content. Similar applications can be found for Facebook wall sharing and Google+ sharing preview snippets (which summarize a post made to Google+) [34]. According to our experiments in [P5], however, neither works very well. The methods' explicit frameworks have not been published in any scientific forum, although the Google+ algorithm is described in technical documents and both are used in real applications.

Web image (WebIma)

In [P5], we propose a method called WebIma for image extraction. It involves four steps: image identification, feature extraction, categorization, and ranking (see Figure 5.3).

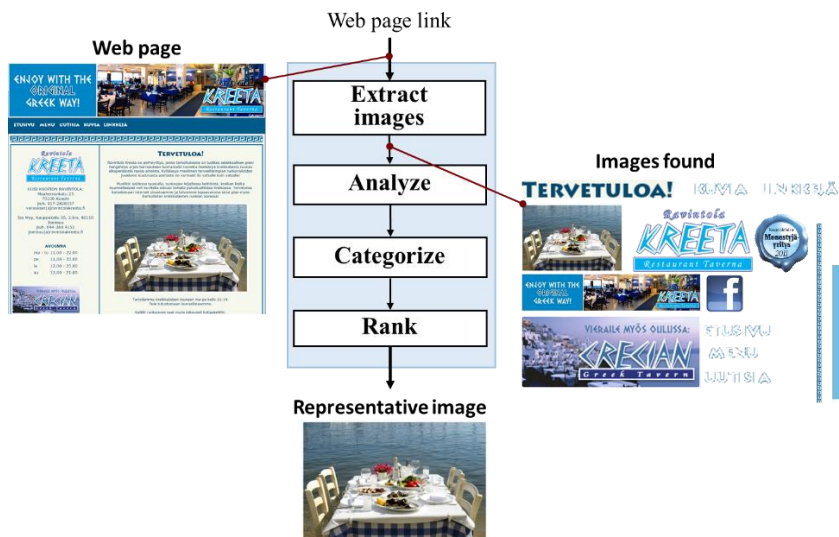


Figure 5.3: Workflow of WebIma

In the *identification step*, we find links to images by locating `` tag, `<link>` tag of type *text/css* or *rel = stylesheet*, and `<script>` tag. Once the images are identified, we move on to the *feature extraction step*, which entails extracting a list of features for each image and its corresponding HTML element: *src*, *alt*, *title*, *from*, *format*, *width*, *height*, *size*, and *aspect ratio* (see Figure 5.4).


	
src	<code>http://www.ravintolakreeta.fi/images/banner.jpg</code>
alt	--
title	--
from	<code>css</code>
format	<code>jpg</code>
width	<code>945</code>
height	<code>202</code>
size	<code>190,890 px</code>
aspect ratio	<code>4.67</code>
parent tag	<code><div></code>
class	<code>header</code>

Figure 5.4: Example of a banner image and its features

In the subsequent *categorization step*, we classify the images into five categories according to their functional role on the web page. In order of priority, the categories are representative, logo, banner, advertisement, and formatting.

- The *representative category* comprises images that are directly associated with a web page’s main content (see Figure 5.5) as well as images that do not fit into other categories.



Figure 5.5: Examples of representative images

- The *logo category* includes images that provide branding information about a web page (see Figure 5.6). The criterion for being categorized as a logo is that at least one of the HTML tag attributes (i.e., image link, detected classes, element IDs, or parent element IDs) contains the keyword 'logo.'



Figure 5.6: Examples of logos

- The *banner category* comprises images that are placed on the top, side, or bottom of a web page such as headers and footers (see Figure 5.7). The criterion for being in the banner category is that at least one HTML tag attribute contains one of the keywords 'banner,' 'header,' 'footer,' or 'button' or that the aspect ratio of the image is higher than a threshold 1.8, which was experimentally obtained using a small training set of 50 web pages.



Figure 5.7: Examples of banners

- The *advertisement category* contains images promoting products or services from other websites (see Figure 5.8). The criterion for this category is that at least one of the HTML tag attributes contains any of the advertising keywords 'free,' 'now,' 'buy,' 'join,' 'adserver,' 'click,' 'affiliate,' 'adv,' 'hits,' or 'counter.'



Figure 5.8: Examples of advertisements

- The *formatting category* includes images and icons that are used to enhance a page's visual appearance, such as spaces, bullets, borders, background, and pictures that are used merely for decoration (see Figure 5.9). Images in this category have either a specific keyword in the attributes of their HTML tag (namely 'background,' 'sprite,' or 'template') or a dimension that is below the empirically selected threshold of 100 pixels.



Figure 5.9: Examples of formatting images

Images may satisfy the criteria of multiple categories. In this case, we use a decision tree to assign an image to the correct category (see Figure 5.10).

The criteria for each category are derived based on an empirical examination of training images from 50 web pages. Similar approaches were used in [67, 82, 85]. For instance, decorative images were identified by the words 'bullet,' 'button,' 'rule,' and 'line' in [82], while advertisement images

were identified by aspect ratio and keywords such as 'free,' 'shop,' 'ad,' and 'soon' in [85].

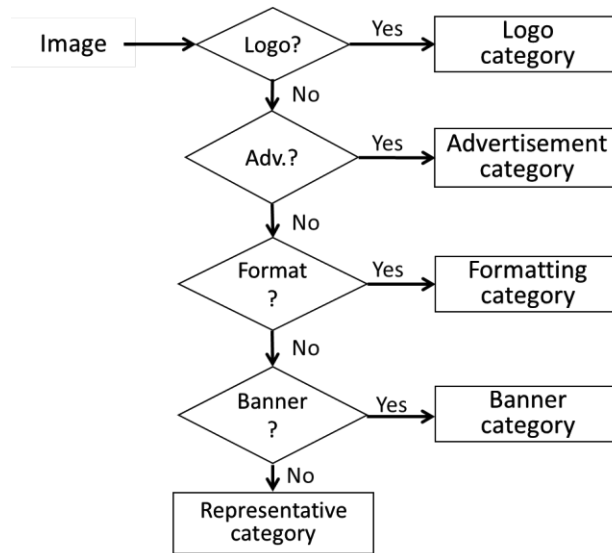


Figure 5.10: Decision tree used for image categorization

Image categorization is advantageous in several applications, such as automatically identifying advertisements, saving web crawler bandwidth by carefully downloading only the most relevant media objects, and automatically converting web pages for consumption on mobile small screen devices [44].

The fourth step is image ranking. After image categorization, we score the images based on a set of rules, as shown in Table 5.1. Several elements in the DOM tree share information with the images and tend to provide useful hints about images' content [116]. These elements include alternative text (alt attribute), image title, image link, page title, and headings [3]. *Alt attribute* provides a short description of an image's content. It replaces images in environments where the image cannot be rendered (e.g., the command line of a web browser), image display is disabled in the browser, or the user is utilizing a screen reader. Image alt has been considered an appropriate source for annotating an image if used properly [3]. *Image title* provides additional information about an image and follows page title rules (i.e., should be short, concise, and relevant to the

image's content). *Image link* contains an image's file name and its path on the web server. It can include keywords related to the image's content.

Table 5.1: Rules used for image scoring.

Rule	Score
Image size ≥ 10.000 px	1
Image alt or image title has a value	1
Keywords of alt or title are in web page <title>	1
Keywords of alt or title are in web page <h1>	1
Keywords of path are in web page <title> or <h1>	1
The image is in the sub-tree of <h1> or <h2> tags	1
Format : jpg/jpeg	1
Format: svg	0.5
Format: png	0.5
Format: gif	0.5

The aforementioned elements may not be adequately used by the author of the web page [26]. For example, web-designing tools often generate *alt* attributes such as 'img1' or 'img005.jpg,' which is too generic to be of value. As a result, the keywords in these sources need to be verified. To compensate for this problem, we use the page title and level one and two headings of web pages for matching keywords, as they usually provide short but important hints on what a web page is about.

Similarly to [116], every time a rule is met, we assign the corresponding image one point. For example, images that have the right size or are located in the subtree under *h1* and *h2* get one point every time the relevant rule is satisfied. The rules related to image format are an exception: here we assign one point to jpg/jpeg images but 0.5 to other formats, given that most content images are in jpeg format [67].

The scores are then summed up, and the images in each category are ranked based on their scores. An exception is the logo, category where the images are sorted by size (as we assume that a web page's logo is larger than other logos on the page).

Our experiments using the WebIma data set¹⁷ show that users pay little attention to image size but prefer square images to images such as banners. About 65% of the representative images chosen by users in these experiments were in jpeg format, which indicates that aspect ratio and image format are more important than other features. We did not mark special observation about other features.

The experimental results show that our method finds the correct image in 64% of the cases. It thus outperforms Google+ (48%) and Facebook (39%), both of which ignore small images regardless of their category. Our findings also suggest that omitting size and aspect ratio criteria when selecting logo images is beneficial, as 8% of the images chosen by users were logos.

¹⁷ <http://cs.uef.fi/mopsi/WebIma/>

6 Applications

6.1 INTRODUCTION TO MOPSI

The work outlined in this thesis has been carried out within Mopsi [28], which is a software application developed by the Machine Learning Group at the School of Computing at the University of Eastern Finland. It implements various location-based services and applications, such as mobile search engines, data collection, user tracking, and route recording. The Mopsi application is a real-time working environment for which new research ideas and solutions in the areas of location-based applications and web content mining are tested. It has a website and mobile applications for major platforms, namely Android, iOS, Windows Phone, and Symbian. These user interfaces enable the collection of location-based data (e.g., photos and GPS trajectories), which can then be analyzed using system-provided tools. Mopsi also has a chat feature and offers integration with Facebook, the most popular sharing network.

The Mopsi environment has over 2,400 registered users, and its database contains three types of data: geo-tagged photos (over 35,000) GPS trajectories (over 10,000), and points of interest (over 414). Photos and points of interest are further discussed below.

Photos are collected by mobile users and presented to the web user on Mopsi websites. Each photo has a location and time stamp information; users can also provide a description. Photos can be shared among Mopsi users, who can browse them utilizing time query filtering. Photos that match the search criteria are displayed on both a map and a time stamp navigation bar (see Figure 6.1).

Users can also create points of interest, which are known as *services* (see Figure 6.2). Each service has a web link, title, list of keywords, photo, additional description (optional), and address

that are manually added by users and further verified by an administrator.



Figure 6.1: Set of photos in the Mopsi system collected by all users in different locations within a year

The main research topics addressed in Mopsi are web page summarization [P1, P2, P4, P5], the collection and analysis of location-based data, the location-based recommender system [103, 104], the analysis of GPS trajectories [105, 106], transportation mode detection [102], and location-based games that uses the Mopsi data collection [97].

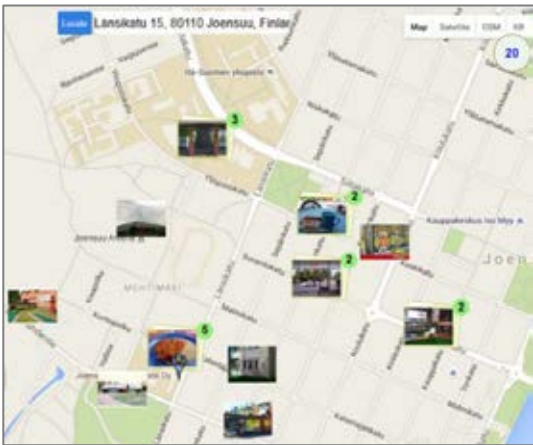


Figure 6.2: Set of services in the Mopsi system collected by users in Joensuu city

In the following sections, we briefly discuss applications that have integrated and are using the methods we have developed.

6.2 LOCATION-BASED SEARCH AND RECOMMENDER

The location-based web search in Mopsi [28] is established based on mining three kinds of data from web pages: postal address, title, and representative image. Location-based searches aim at finding points of interest within a specified distance from the user's current location. It takes a search keyword and user location as input and outputs a list of results with the following information: rank, web link, title, image, and distance to location (see Figure 6.3).

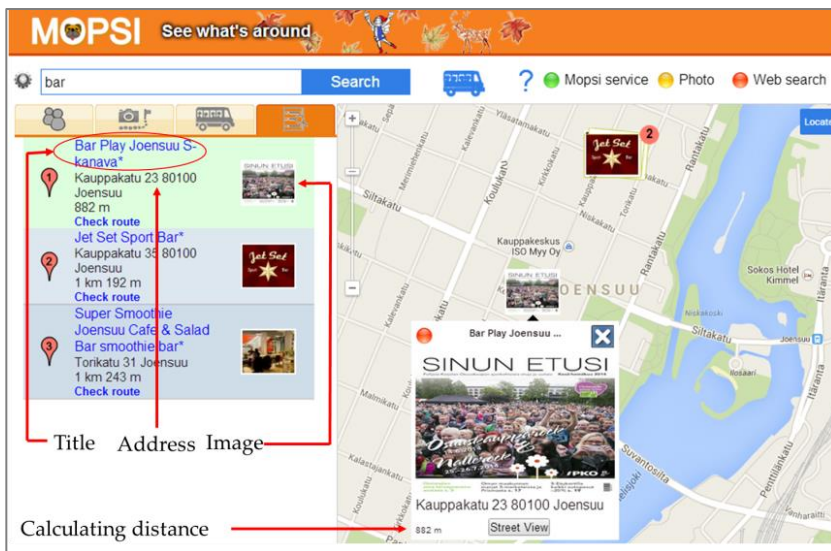


Figure 6.3: Location-based web search in Mopsi

The Mopsi web search is a meta search engine that uses an external engine to collect results, which it then processes to produce output based on distance to the user's location. The advantage of meta-searching is that it does not require web pages to be crawled and indexed; it instead combines results from multiple search engines, which allows users to access more information.

We use results from two search engines: BOSS API by Yahoo!¹⁸ and custom search by Google.¹⁹ To keep the computations reasonable, we process only the first ten web pages identified.

The list of the retrieved web pages then undergoes a post-processing step that entails extracting and validating postal addresses using the algorithm in [96]. We extract a title and representative image from each web page, which we then aggregate to form entities shown in the search results page. The list displayed to the end user is sorted by distance, as a service that is closer to the user location is assumed to be more relevant.

The same entities are also needed in the Mopsi recommender system. This system suggests points of interest and things to do that are nearby the user (without needing a search keyword).

6.3 THE INTERACTIVE TOOL FOR SERVICES

Summaries are utilized to improve the interactive tool for adding and editing services in the Mopsi database using mined data from given web pages. When a user creates or edits a point of interest, he or she must add essential information (namely the web link, title, keywords, image, and postal address). Adding this information manually is time-consuming and prone to error. A better solution is to utilize a user-provided web link and automatically extract the rest of the information (which the user can then check and edit, if necessary). This kind of semi-automatic content creation can speed the process up significantly. In Figure 6.4, the title, keywords, and representative images are extracted from the example web page PizzaBuffa²⁰ using the link provided by a user. The extracted title is identical to the one annotated manually. The user has provided two keywords (ravintola, pizza); our method extracted

¹⁸ <http://developer.yahoo.com/boos>

¹⁹ <http://developers.google.com/custom-search>

²⁰ <https://www.raflaamo.fi/fi/joensuu/pizzabuffa-prisma-joensuu>

four. Although the image selected by the user no longer exists when we process the page, our method suggests four candidate images – three of which describe the page's content quite well.

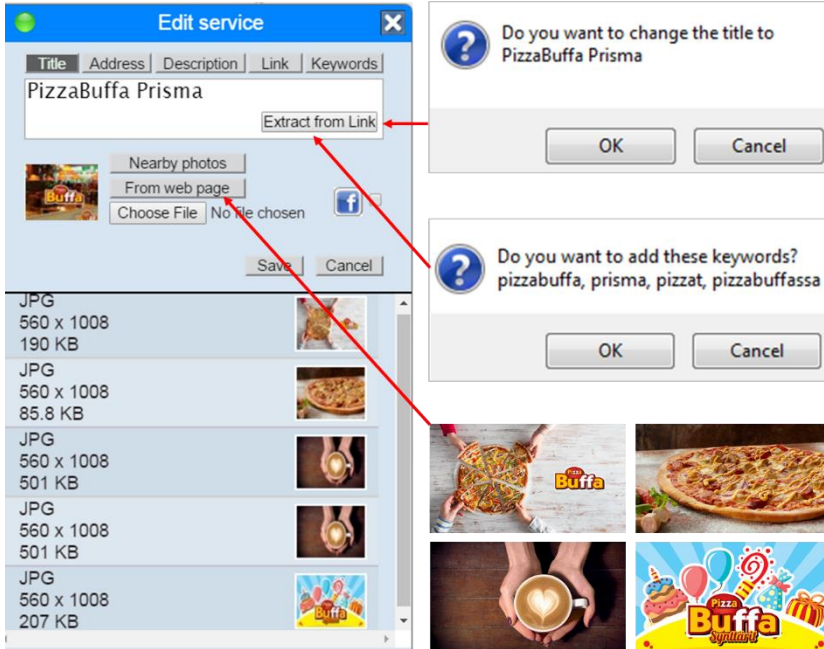


Figure 6.4: Service edits in Mopsi

6.4 CLUSTERING

We cluster Mopsi data using the title and the keywords of the photos and services. Photos are clustered based on their titles, while services are clustered by keywords.

Photo clustering allows users to organize their photos for easy access. For example, consider a user who has a collection of 1,000 photos, some of which are of the same object, place, or event (but all taken at different times). Without clustering, this user has to repeatedly navigate through the entire collection to locate this subset of photos. In contrast, clustering enables him or her to find similar photos by just navigating through the clusters. The user wants to neither delete these memories nor put an effort into browsing through the collection, and

clustering eases the workload. Figure 6.5 shows an example of a cluster of building photos taken by the user Pasi. Photo clustering also allows users to quickly find information about places they have never visited from their friends' collections.



Figure 6.5: Example cluster: Building images

The clustering of services (i.e., web pages) is another application of our keyword extraction method. In Mopsi, the user has the option to examine a set of services in three different ways:

- In a list;
- On a map; or
- In clusters.

A list displays all services and is easy to read, but its size can be exhaustive. Sorting the list by title, time, address, or first keyword may be helpful; however, if the user does not know the exact title, keyword, or other information that would identify the service, searching can still take a long time.

The map is useful when the user knows the (exact or approximate) location, which can be found by panning and zooming in the map interface. The same actions also enable users to find all services in the same location, which is useful when they just want to know ‘what is there.’ However, if a user wants to find, say, all of the sport facilities in town, neither location nor keywords will suffice.

Clustering is useful for identifying duplicate records, grouping related services together, and recognizing chains of services (such as franchises). For example, the following services are all grouped together because they offer lunch and coffee:

- Jokiasema
- Kahvila Luuppi
- Heinosen leipomo
- Café Delivo
- Lounasravintola Tuulikissa

In Mopsi, points of interest (e.g., restaurants, shops, and cafes) can be clustered and displayed in a categorized manner rather than randomly. Grouping entities in a results page helps users to locate and compare information faster and with less effort.

Beyond Mopsi, web page clustering is useful in several other applications, such as improving the quality of web searches [89]. When a user submits a query, it frequently occurs that the words in that query mismatch the words in a web page, even if they are semantically related. Inconsistency between the query and page words is a well-known problem in information retrieval; it was first recognized by Furnas [31]. If a query results in a web page, other web pages in same cluster might also be relevant; web page clustering ensures that these other pages are also returned to the user. Web page clustering can also be useful in determining the similarities between queries; for instance, two queries resulting in two different web pages within the same clusters can be recognized as being similar [110].

7 *Summary of Contributions*

This chapter summarizes the contributions of our five publications. Publications [P1], [P2], [P4], and [P5] concern the extraction of web page content, while publication [P3] presents a systematic study of existing similarity measures for title comparison.

In [P1], we introduce a method for extracting the title from an HTML web page. The main goal is to create a simple and effective way to complete the task. We show that while information contained in the title and meta fields can be utilized to extract candidate titles, it should not be used as such because it also often contains irrelevant data. A better approach is to divide the content of these tags into segments, which are further analyzed using the text of the rest of the page. Furthermore, we use the title field as the primary source for extracting the candidate title, unlike other approaches (which rarely utilize this field). We study the effect of using three sources to select the correct title: a web page's link, placement of segment in the title and meta tags, and popularity among heading tags. Our results show that a page's link has the most significant effect. Despite the simplicity of this method, it outperforms the comparative heuristic and visual-based methods by 28 percentage points and 40 percentage points. Moreover, the method is independent of both language and domain.

In [P2], we improve the work of [P1] by utilizing a web page's text content and incorporating natural language processing so that only grammatically correct candidate titles are chosen. Our aim is to provide a compact title for service-based web pages to display on mobile devices. We introduce a novel linguistic model for English titles and apply it to extract potential phrases from a web page. Unlike existing methods that consider the entire text of the DOM nodes as potential candidates, we consider only those phrases that fit certain

grammatical patterns. This segmentation approach improves the extraction accuracy of the comparative method by 13 percentage points (as measured by F-score). We define new features (such as a phrase's syntactic structure and similarity with a web page link) and subsequently discover that these types of features are more relevant than visual features for service-based web pages. We also address the choice of classifiers, as they have been insufficiently studied in the literature. We compare four classifiers (Naive Bayes, SVM, clustering, and k-NN), showing that simple classifiers such as clustering and k-NN perform almost as well as SVM. The new method we develop improves the F-score of the results in [P1] by 10 percentage points. The methods in both [P1] and [P2] outperform Google's F-scores by 17 percentage points and 27 percentage points and offer readily available solutions for title extraction problems.

In [P3], we provide a systematic study of 21 existing similarity measures with the goal of ascertaining which measure should be used to determine the similarity between two titles (as called for in the evaluation of the results in [P1] and [P2]). Based on our experiments, we provide recommendations on which measure to use in which scenario: Soft-TFIDF (which combines Jaro-Winkler with TF-IDF) works well with real data and would probably be the best choice in that context. Damerau-Levenshtein and Trigram also work well with real data, although the former is vulnerable to token swap and the latter to character changes; if these artifacts are not critical, these measures are recommended.

In [P4], we study keyword extraction from HTML web page with the aim of developing a method that is domain independent. Instead of considering each word individually, we group semantically related words (e.g., 'cost,' 'price,' and 'charges') together so that each cluster represents one concept. We cluster nouns based on their semantic similarity and rank the clusters according to the distribution of the nouns in the DOM tree. Experiments on a data set collected from five different domains (namely education, news, tourism, beauty and fitness, and food and drink) show that complete linkage is

more stable than average linkage on the selection of the merge threshold. Our findings suggest that the distribution of nouns over the DOM text nodes is a more informative feature than term frequency. Results also show that our method outperforms the state-of-the-art (i.e., TextRank) by 10 percentage points in relation to F-score.

In [P5], we study how to select a representative image from a web page. Like title and keywords, such an image is needed as part of a web page's summarization. Despite it being in use on social media sites, no good solutions were identified in the literature. The closest examples can be found on Facebook and Google+, which use simple heuristic solutions. Our rule-based method provides 64% accuracy, which is better than that of both Google+ (48%) and Facebook (39%). Besides being used in Mopsi searches, the proposed method would also be directly applicable to both social media sites.

8 *Conclusions*

In this thesis, we have proposed new methods for summarizing a web page's content using its title, keywords, and a representative image. The problem of matching title phrases has also been addressed.

We have also presented new approaches, such as extracting only potential phrases from the DOM text node for titling. This method is shown to be more appropriate than just considering the entire text of the DOM nodes in the case of service-based web pages. A page's visual features are also observed to be less relevant to title extraction; as such, other types of features (e.g., statistical and linguistic features) might be considered.

The semi-structured environment of a web page is useful in keyword extraction. Despite the fact that term frequency has been used widely in keyword extraction, the distribution of words over the DOM text nodes can be utilized to select keywords more effectively.

Categorizing web page images according to their functional role is useful as a preliminary step in distinguishing between images of different types like logo and formatting images that have similar features (such as size and aspect ratio).

A novel setup of experiments to evaluate similarity measures for comparing two title phrases has been introduced. While Soft-TFIDF produces satisfying results, other combinations of measures (e.g., Damerau-Levenshtein and Dice) may yield better results.

Although we have already seen many examples of successful applications in which using a web page summary to compose all or individual components is useful, many problems remain unsolved. Potential future research thus includes:

- Analyzing the effect of all of the features used for title extraction with different pre-processing steps. This could entail utilizing entire text node, n-grams, and POS tag

patterns to model a method that can be generalized to all types of web pages.

- Investigating the effects of different similarity measures and clustering methods on keyword extraction to determine which subproblem should be focused on more to improve the quality of the extracted keywords.
- Bridging the gap between keyword extraction and keyword assignment by using web crawling, where keywords from similar web pages can be used to annotate the web page in question.
- Investigating solutions to keyword extraction that are independent of language (unlike current solutions), so that they better fit into real applications.
- Finding ways to further improve WebIma (even though it already outperforms Facebook and Google+). This may include developing techniques for extracting features from an image itself and applying the machine learning approach to the identification of representative images.

References

- [1] Adam, G., Bouras, C., & Pouloupoulos, V. (2010). Image Extraction from Online Text Streams: A Straightforward Template Independent Approach without Training. *Advanced Information Networking and Applications Workshops (WAINA), 2010 IEEE 24th International Conference on* (pp. 609-614). IEEE.
- [2] Alam, H., Hartono, R., Kumar, A., Rahman, A. F. R., Tarnikova, Y., & Wilcox, C. (2003). Web Page Summarization for Handheld Devices: A Natural Language Approach. In *ICDAR* (Vol. 3, p. 1153).
- [3] Alciç, S. (2011). *Web image context extraction: methods and evaluation* (Doctoral dissertation, PhD thesis, Heinrich-Heine-University of Düsseldorf).
- [4] Al-Shammari, E., & Lin, J. (2008). A novel Arabic lemmatization algorithm. In *Proceedings of the second workshop on Analytics for noisy unstructured text data* (pp. 113-118). ACM.
- [5] Aquino, G., Hasperué, W., Estrebou, C., & Lanzarini, L. C. (2013). A Novel, Language-Independent Keyword Extraction Method. In *XVIII Congreso Argentino de Ciencias de la Computación*.
- [6] Armano, G., Giuliani, A., & Vargiu, E. (2011). Experimenting Text Summarization Techniques for Contextual Advertising. *Italian Information Retrieval (IIR) Workshop*, 2011.
- [7] Azad, H. K., Raj, R., Kumar, R., Ranjan, H., Abhishek, K., & Singh, M. P. (2014). Removal of noisy information in

- web pages. In *Proceedings of the 2014 International Conference on Information and Communication Technology for Competitive Strategies* (p. 88). ACM.
- [8] Bougouin, A., Boudin, F., & Daille, B. (2013). Topicrank: Graph-based topic ranking for keyphrase extraction. In *International Joint Conference on Natural Language Processing (IJCNLP)* (pp. 543-551).
 - [9] Bracewell, D. B., Ren, F., & Kuriowa, S. (2005). Multilingual single document keyword extraction for information retrieval. In *Natural Language Processing and Knowledge Engineering, 2005. IEEE NLP-KE'05. Proceedings of 2005 IEEE International Conference on* (pp. 517-522). IEEE.
 - [10] Breiman, L. (2001). Random forests. *Machine learning*, 45(1), pp. 5-32.
 - [11] Brew, C., & McKelvie, D. (1996). Word-pair extraction for lexicography. *Int. Conf. on New Methods in Language Processing*, pp. 45-55.
 - [12] Brin S., & Page L. (1998). The anatomy of a large-scale hypertextual Web search engine. *Computer networks and ISDN systems*, 30(1), 107-117.
 - [13] Budanitsky, A., & Hirst, G. (2006). Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1), 13-47.
 - [14] Cai, D., Yu, S., Wen, J. R., & Ma, W. Y. (2003). *VIPS: a vision-based page segmentation algorithm*. Microsoft technical report, MSR-TR-2003-79.
 - [15] Carter, B. (2014). HTML Architecture, a Novel Development System (HANDS): An Approach for Web Development. In *Information and Computer Technology*

- (GOCICT), 2014 Annual Global Online Conference (pp. 90-95). IEEE.
- [16] Changuel, S., Labroche, N., & Bouchon-Meunier, B. (2009). A general learning method for automatic title extraction from HTML pages. In *Machine Learning and Data Mining in Pattern Recognition* (pp. 704-718). Springer Berlin Heidelberg.
 - [17] Cohen, W., Ravikumar, P., & Fienberg, S. (2003). A comparison of string metrics for matching names and records. In *Kdd workshop on data cleaning and object consolidation* (Vol. 3, pp. 73-78).
 - [18] Cohen, W., Ravikumar, P., & Fienberg, S. (2003b). A comparison of string metrics for matching names and records. In *Kdd workshop on data cleaning and object consolidation* (Vol. 3, pp. 73-78).
 - [19] Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1), 21-27.
 - [20] Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3), 171-176.
 - [21] D'Avanzo, E., Magnini, B., & Vallin, A. (2004). Keyphrase extraction for summarization purposes: The LAKE system at DUC-2004. In *Proceedings of the 2004 document understanding conference*.
 - [22] Domingos, P., & Pazzani, M. (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine learning*, 29(2-3), 103-130.
 - [23] Dostal, M., & Jezek, K. (2011). Automatic Keyphrase Extraction based on NLP and Statistical Methods. In *DATESO* (pp. 140-145).

- [24] Fan, J., Luo, P., & Joshi, P. (2011). Identification of web article pages using HTML and visual features. In *IS&T/SPIE Electronic Imaging* (pp. 78790K-78790K). International Society for Optics and Photonics.
- [25] Fauzi, F., Hong, J. L., & Belkhatir, M. (2009). Webpage segmentation for extracting images and their surrounding contextual information. In *Proceedings of the 17th ACM international conference on Multimedia* (pp. 649-652). ACM.
- [26] Feng, H., Shi, R., & Chua, T. S. (2004). A bootstrapping framework for annotating and retrieving WWW images. In *Proceedings of the 12th annual ACM international conference on Multimedia* (pp. 960-967). ACM.
- [27] Frank, E., Paynter, G. W., Witten, I. H., Gutwin, C., & Nevill-Manning, C. G. (1999). Domain-specific keyphrase extraction. In *IJCAI* (Vol. 99, pp. 668–673).
- [28] Fränti, P., Chen, J., & Tabarcea, A. (2011). Four Aspects of Relevance in Sharing Location-based Media: Content, Time, Location and Network. In *WEBIST* (pp. 413–417).
- [29] Fränti, P., & Kivijärvi, J. (2000). Randomised local search algorithm for the clustering problem. *Pattern Analysis & Applications*, 3(4), 358–369.
- [30] Friedman, C., & Sideli, R. (1992). Tolerating spelling errors during patient validation. *Computers and Biomedical Research*, 25(5), 486–509.
- [31] Furnas, G. W., Landauer, T. K., Gomez, L. M., & Dumais, S. T. (1987). The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11), 964-971.
- [32] Gazendam, L., Wartena, C., & Brussee, R. (2010). Thesaurus based term ranking for keyword extraction.

- In *Database and Expert Systems Applications (DEXA), 2010 Workshop on* (pp. 49-53). IEEE.
- [33] Gibson, D., Punera, K., & Tomkins, A. (2005). The volume and evolution of web page templates. In *Special interest tracks and posters of the 14th international conference on World Wide Web* (pp. 830-839). ACM.
 - [34] Google+ platform, 2014, <https://developers.google.com/+/web/snippet/?hl=no>
 - [35] Gotoh, O. (1982). An improved algorithm for matching biological sequences. *Journal of molecular biology*, 162(3), 705-708.
 - [36] Haggag, M. H. (2013). Keyword Extraction using Semantic Analysis. *International Journal of Computer Applications*, 61(1).
 - [37] Hammer, J., Garcia-Molina, H., Cho, J., Aranha, R., & Crespo, A. (1997). Extracting Semistructured Information from the Web.
 - [38] Hammouda, K. M., Matute, D. N., & Kamel, M. S. (2005). Corephrase: Keyphrase extraction for document clustering. In *Machine Learning and Data Mining in Pattern Recognition* (pp. 265-274). Springer Berlin Heidelberg.
 - [39] Han, P., Shen, S., Wang, D., & Liu, Y. (2012). The influence of word normalization in English document clustering. In *Computer Science and Automation Engineering (CSAE), 2012 IEEE International Conference on* (Vol. 2, pp. 116-120). IEEE.
 - [40] Helfman, J. I., & Hollan, J. D. (2000). Image representations for accessing and organizing Web information. In *Photonics West 2001-Electronic Imaging* (pp. 91-101). International Society for Optics and Photonics.

- [41] Huang, Z., Jin, H., Yuan, P., & Han, Z. (2006). Header metadata extraction from semi-structured documents using template matching. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"* (pp. 1776-1785). Springer Berlin Heidelberg.
- [42] Hulth, A. (2003). Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing* (pp. 216-223). Association for Computational Linguistics.
- [43] Humphreys, J. K. (2002). PhraseRate: An HTML Keyphrase Extractor. *Dept. of Computer Science, University of California, Riverside, California, USA, Tech. Rep.*
- [44] Hu, J., & Bagga, A. (2003). Functionality-Based Web Image Categorization. *WWW (Posters)*, 2(003).
- [45] Hu, Y., Xin, G., Song, R., Hu, G., Shi, S., Cao, Y., & Li, H. (2005). Extraction from bodies of HTML documents and its application to web page retrieval. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 250-257). ACM.
- [46] Hyerle, D. (2000). Thinking maps: Visual tools for activating habits of mind. In A. L. Costa, B. Kallick, and D. Perkins (Eds.), *Activating and engaging habits of mind* (pp. 46-58). Alexandria, VA: Association for Supervision and Curriculum Development.
- [47] Jansen, B. J., & Spink, A. (2005). An analysis of web searching by European AlltheWeb.com users. *Information Processing & Management*, 41(2), 361–381.
- [48] Jaro, M. A. (1989). Advances in record-linkage methodology as applied to matching the 1985 census of

- Tampa, Florida. *Journal of the American Statistical Association*, 84(406), 414–420.
- [49] Jensen, E. (2008). *Brain-based learning: The new paradigm of teaching*. Corwin Press.
 - [50] Jeong, O. R., Oh, J., Kim, D. J., Lyu, H., & Kim, W. (2014). Determining the titles of Web pages using anchor text and link analysis. *Expert Systems with Applications*, 41(9), 4322- 4329.
 - [51] Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Tenth European Conference on Machine Learning*, Springer Berlin Heidelberg, 137–142.
 - [52] Joorabchi, A., & Mahdi, A. E. (2013). Automatic keyphrase annotation of scientific documents using Wikipedia and genetic algorithms. *Journal of Information Science*, 0165551512472138.
 - [53] Joshi, P. M., & Liu, S. (2009). Web document text and images extraction using DOM analysis and natural language processing. In *Proceedings of the 9th ACM symposium on Document engineering* (pp. 218-221). ACM.
 - [54] Jo, T. (2003). Neural based approach to keyword extraction from documents. In *Computational Science and Its Applications—ICCSA 2003* (pp. 456-461). Springer Berlin Heidelberg.
 - [55] Kang, S. S. (2003). Keyword-based document clustering. In *Proceedings of the sixth international workshop on Information retrieval with Asian languages-Volume 11* (pp. 132-137). Association for Computational Linguistics.
 - [56] Kan, M. Y., & Thi, H. O. N. (2005). Fast webpage classification using URL features. In *Proceedings of the*

14th ACM international conference on Information and knowledge management (pp. 325-326). ACM.

- [57] Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5), 604-632.
- [58] Kohlschütter, C., Fankhauser, P., & Nejdl, W. (2010). Boilerplate detection using shallow text features. In *Proceedings of the third ACM international conference on Web search and data mining* (pp. 441-450). ACM.
- [59] Kolcz, A., Prabakarmurthi, V., & Kalita, J. (2001). Summarization as feature selection for text categorization. In *Proceedings of the tenth international conference on Information and knowledge management* (pp. 365-370). ACM.
- [60] Lafferty, J., McCallum, A., & Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the eighteenth international conference on machine learning*, 282-289.
- [61] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions and reversals. In *Soviet physics doklady* (Vol. 10, p. 707).
- [62] Lin, C. Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop* (Vol. 8).
- [63] Liu, Z., Li, P., Zheng, Y., & Sun, M. (2009). Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing* (Vol. 1, pp. 257-266). Association for Computational Linguistics.

- [64] Li, Y., Zaragoza, H., Herbrich, R., Shawe-Taylor, J., & Kandola, J. (2002). The perceptron algorithm with uneven margins. In *ICML* (Vol. 2, pp. 379-386).
- [65] Lopez, C., Prince, V., & Roche, M. (2011). Automatic titling of articles using position and statistical information. In *RANLP'11: Recent Advances in Natural Language Processing* (pp. 727- 732).
- [66] Lopez, C., Prince, V., & Roche, M. (2014). How can catchy titles be generated without loss of informativeness? *Expert Systems with Applications*, 41(4), 1051–1062.
- [67] Maekawa, T., Hara, T., & Nishio, S. (2006). Image classification for mobile web browsing. In *Proceedings of the 15th international conference on World Wide Web* (pp. 43-52). ACM.
- [68] Malakasiotis, P., & Androutsopoulos, I. (2007). Learning textual entailment using SVMs and string similarity measures. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing* (pp. 42-47). Association for Computational Linguistics.
- [69] Mani, I., & Maybury, M. T. (Eds.). (1999). *Advances in automatic text summarization* (Vol. 293). Cambridge, MA: MIT press. Chicago
- [70] Marujo, L., Gershman, A., Carbonell, J., Frederking, R., & Neto, J. P. (2013). Supervised topical key phrase extraction of news stories using crowdsourcing, light filtering and co-reference normalization. *arXiv preprint arXiv:1306.4886*.
- [71] Medelyan, O., & Witten, I. H. (2006). Thesaurus based automatic keyphrase indexing. In *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries* (pp. 296-297). ACM.

- [72] Medelyan, O., & Witten, I. H. (2008). Domain-independent automatic keyphrase indexing with small training sets. *Journal of the American Society for Information Science and Technology*, 59(7), 1026-1040.
- [73] Michael Pucher, F. T. W. (2004). Performance Evaluation of WordNet-based Semantic Relatedness Measures for Word Prediction in Conversational Speech.
- [74] Michelson, M., & Knoblock, C. A. (2007). Unsupervised information extraction from unstructured, ungrammatical data sources on the World Wide Web. *International Journal of Document Analysis and Recognition (IJ DAR)*, 10(3-4), 211–226.
- [75] Mihalcea, R., Corley, C., & Strapparava, C. (2006). Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI* (Vol. 6, pp. 775–780).
- [76] Mihalcea, R., & Tarau, P. (2004). TextRank: Bringing order into texts. Association for Computational Linguistics.
- [77] Mohammadzadeh, H. (2013). Improving Retrieval Accuracy in Main Content Extraction from HTML Web Documents. PhD dissertation, The University of Leipzig.
- [78] Mohammadzadeh, H., Gottron, T., Schweiggert, F., & Heyer, G. (2012). TitleFinder: extracting the headline of news web pages based on cosine similarity and overlap scoring similarity. In *Proceedings of the twelfth international workshop on Web information and data management* (pp. 65-72). ACM.
- [79] Monge, A. E., & Elkan, C. (1996). The Field Matching Problem: Algorithms and Applications. In *KDD* (pp. 267–270).

- [80] Needleman, S. B., & Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3), 443-453.
- [81] Ortiz, R., Pinto, D., Tovar, M., & Jiménez-Salazar, H. (2010). BUAP: An unsupervised approach to automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th international workshop on semantic evaluation* (pp. 174-177). Association for Computational Linguistics.
- [82] Paek, S., & Smith, J. R. (1998). Detecting image purpose in World Wide Web documents. In *Photonics West'98 Electronic Imaging* (pp.151-158). International Society for Optics and Photonics.
- [83] Palshikar, G. K. (2007). Keyword extraction from a single document using centrality measures. In *Pattern Recognition and Machine Intelligence* (pp. 503-510). Springer Berlin Heidelberg.
- [84] Park, G., Baek, Y., & Lee, H. K. (2006). Web image retrieval using majority-based ranking approach. *Multimedia Tools and Applications*, 31(2), 195–219.
- [85] Parmar, H. R., & Gadge, J. (2011). Removal of Image Advertisement from Web Page. *International Journal of Computer Applications*, 27(7).
- [86] Pasquier, C. (2010). Task 5: Single document keyphrase extraction using sentence clustering and Latent Dirichlet Allocation. In *Proceedings of the 5th international workshop on semantic evaluation* (pp. 154-157). Association for Computational Linguistics.
- [87] Paukkeri, M. S., & Honkela, T. (2010). Likey: unsupervised language-independent keyphrase

- extraction. In *Proceedings of the 5th international workshop on semantic evaluation* (pp. 162-165). Association for Computational Linguistics
- [88] Potter, M. C., Wyble, B., Hagmann, C. E., & McCourt, E. S. (2014). Detecting meaning in RSVP at 13 ms per picture. *Attention, Perception, & Psychophysics*, 76(2), 270-279.
- [89] Qi, X., & Davison, B. D. (2009). Web page classification: Features and algorithms. *ACM Computing Surveys (CSUR)*, 41(2), 12.
- [90] Quinlan, J. R. (1993). *C4.5: programs for machine learning*. San Francisco: Morgan Kaufmann Publishers Inc.
- [91] Robbins, J. N. (2012). *Learning web design: A beginner's guide to HTML, CSS, JavaScript, and web graphics*. "O'Reilly Media, Inc."
- [92] Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010). Automatic keyword extraction from individual documents. *Text Mining*, 1-20.
- [93] Shen, D., Yang, Q., & Chen, Z. (2007). Noise reduction through summarization for Web-page classification. *Information processing & management*, 43(6), 1735–1747.
- [94] Smith, T. F., & Waterman, M. S. (1981). Identification of common molecular subsequences. *Journal of molecular biology*, 147(1), 195–197.
- [95] Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1), 11–21.
- [96] Tabarcea, A., Hautamäki, V., & Fränti, P. (2010). Ad-hoc georeferencing of web-pages using street-name prefix

- trees. In *Web Information Systems and Technologies* (pp. 259-271). Springer Berlin Heidelberg.
- [97] Tabarcea, A., Wan, Z., Waga, K., & Fränti, P. (2013). O-Mopsi: Mobile Orienteering Game using Geotagged Photos. In *WEBIST* (pp. 300–303).
- [98] Teevan, J., Cutrell, E., Fisher, D., Drucker, S. M., Ramos, G., André, P., & Hu, C. (2009). Visual snippets: summarizing web pages for search and revisitation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 2023-2032). ACM.
- [99] Toutanova, K., Klein, D., Manning, C. D., & Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*. (Vol. 1, pp. 173-180). Association for Computational Linguistics.
- [100] Tsymbalenko, Y., & Munson, E. V. (2001). Using HTML metadata to find relevant images on the World Wide Web. *Proceedings of internet computing*, 2, 842–848.
- [101] Turney, P. (1999). Learning to extract keyphrases from text.
- [102] Waga, K., Tabarcea, A., Chen, M., & Franti, P. (2012b). Detecting movement type by route segmentation and classification. In *Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, 2012 8th International Conference on (pp. 508-513). IEEE.
- [103] Waga, K., Tabarcea, A., & Fränti, P. (2011). Context aware recommendation of location-based data. In *Proceedings of the 15th International Conference on System Theory, Control, and Computing* (pp. 1-6).

- [104] Waga, K., Tabarcea, A., & Franti, P. (2012c). Recommendation of points of interest from user generated data collection. In *Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2012 8th International Conference on* (pp. 550-555). IEEE.
- [105] Waga, K., Tabarcea, A., Mariescu-Istodor, R., & Fränti, P. (2013). Real Time Access to Multiple GPS Tracks. In *WEBIST* (pp. 293-299).
- [106] Waga, K., Tabarcea, A., Mariescu-Istodor, R., & Fränti, P. (2012). System for real time storage, retrieval and visualization of GPS tracks. In *International Conference on System Theory, Control and Computing. Sinaia, Romania*.
- [107] Wang, C., Wang, J., Chen, C., Lin, L., Guan, Z., Zhu, J., Zhang, C. & Bu, J. (2009). Learning to extract web news title in template independent way. In *Rough Sets and Knowledge Technology* (pp. 192-199). Springer Berlin Heidelberg.
- [108] Wang, R., Liu, W., & McDonald, C. (2014). How preprocessing affects unsupervised keyphrase extraction. In *Computational Linguistics and Intelligent Text Processing* (pp. 163-176). Springer Berlin Heidelberg.
- [109] Wartena, C., & Brussee, R. (2008). Topic detection by clustering keywords. In *Database and Expert Systems Application, 2008. DEXA'08. 19th International Workshop on* (pp. 54-58). IEEE.
- [110] Wen, J. R., & Zhang, H. J. (2004). Query clustering in the web context. In *Clustering and Information Retrieval* (pp. 195–225). Springer US.
- [111] Win, C. S., & Thwin, M. M. S. (2014). Web Page Segmentation and Informative Content Extraction for Effective Information Retrieval. *IJCCER*, 2(2), 35–45.

- [112] Winkler, W. E. (1990). String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage.
- [113] Woodruff, A., Faulring, A., Rosenholtz, R., Morrisson, J., & Pirolli, P. (2001). Using thumbnails to search the Web. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 198-205). ACM.
- [114] Wu, C., Marchese, M., Wang, Y., Krapivin, M., Wang, C., Li, X., & Liang, Y. (2009). Data preprocessing in SVM-based keywords extraction from scientific documents. In *Innovative Computing, Information and Control (ICICIC), 2009 Fourth International Conference on* (pp. 810-813). IEEE.
- [115] Xue, Y., Hu, Y., Xin, G., Song, R., Shi, S., Cao, Y., Lin, C. & Li, H. (2007). Web page title extraction and its application. *Information processing & management*, 43(5), 1332- 1347.
- [116] Yanai, K. (2003). Generic image classification using visual knowledge on the web. In *Proceedings of the eleventh ACM international conference on Multimedia* (pp. 167-176). ACM.
- [117] Zervanou, K. (2010). UvT: The UvT Term extraction system in the keyphrase extraction task. In *Proceedings of the 5th international workshop on semantic evaluation* (pp. 194-197). Association for Computational Linguistics.
- [118] Zhang, K., Xu, H., Tang, J., & Li, J. (2006). Keyword extraction using support vector machine. In *Advances in Web-Age Information Management* (pp. 85-96). Springer Berlin Heidelberg.
- [119] Zhang, Y., Milios, E., & Zincir-Heywood, N. (2004). A Comparison of Keyword-and Keyterm-Based Methods for Automatic Web Site Summarization. In *AAAI04*

Workshop on Adaptive Text Extraction and Mining (pp. 15-20).

- [120] Zhang, Y., Zincir-Heywood, N., & Milios, E. (2002). World Wide Web Site Summarization. Technical report CS-2002-08
- [121] Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., & Nevill-Manning, C. G. (1999). KEA: Practical automatic keyphrase extraction. In *Proceedings of the fourth ACM conference on Digital libraries* (pp. 254-255). ACM.

NAJLAH GALI

With the rapid growth of the World Wide Web, the amount of information available online is becoming incredibly high and leading to information overload. Summarization helps the user to get a general overview of the web page's main content before deciding whether to read it in-depth. This thesis presents new methods to extract a compact summary from an HTML web page and utilize them in a location-based application called Mopsi. The proposed methods provide readily available solutions for web page summarization.



UNIVERSITY OF
EASTERN FINLAND

uef.fi

**PUBLICATIONS OF
THE UNIVERSITY OF EASTERN FINLAND**
Dissertations in Forestry and Natural Sciences

ISBN 978-952-61-2407-0
ISSN 1798-5668