Example of the Use of the Style for Writing Theses at CS, SFU

by

Brown Burger

M.Sc., Even Better University, 1994

B.Sc. (Hons.), Even Better University, 1990

Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of

Doctor of Philosophy

in the
School of Computing Science
Faculty of Applied Sciences

APPROVAL

Name:	Brown Burger
Degree:	Doctor of Philosophy
Title of Thesis:	Example of the Use of the Style for Writing Theses at CS, SFU

Examining Committee: Dr. Brian Funt
Chair

Dr. Torsten Möller,
Professor, Senior Supervisor

Dr. Ghassan Hamarneh,
Associate Professor, Senior Supervisor

Dr. Mirza Faisal Beg,
Associate Professor, SFU Examiner

Dr. Christopher R. Johnson,
External Examiner, Distinguished Professor of
Computer Science, University of Utah

Date Approved: September 9th, 2013

Abstract

Here you put the abstract of the thesis. adkgakfhlsufp osudif sDHOs dfus sdfgsuduf sf

To whomever whoever reads this!

"Don't worry, Gromit. Everything's under control!"

— The Wrong Trousers, AARDMAN ANIMATIONS*, 1993*

Acknowledgments

Here go all the people you want to thank.

Contents

Approval ii

Partial Copyright License iii

Abstract iv

Dedication v

Quotation vi

Acknowledgments vii

Contents viii

List of Tables x

List of Figures xi

List of Programs xii

Preface xiii

1 Purpose **1**

1.1 Main features of csthesis.sty 1

1.2 Distribution 2

1.3 How to use csthesis.sty 3

2 Changes from sfuthesis.sty **4**

2.1 Page layout 4

2.2 Line spacing 5

2.3 Front matter 5

2.3.1 Title page 5

2.3.2 Approval page 5
2.3.3 Dedication and Quotation pages 6
2.3.4 Contents 6
2.3.5 Preface 6
2.4 Back matter 6
2.4.1 Appendices 6
2.4.2 Bibliography 6
2.4.3 Index 7

3 introduction **8**
3.1 literature review 8
3.2 what are the algorithms in keyword extraction 8
3.3 our approach our algorithm 14
3.4 visualization of trends 14
3.5 conclusion 14

Bibliography **15**

Appendix A Appendices:Sectioning **17**
A.1 Section 17
A.1.1 Subsection 17

Appendix B Illustrating Lists **18**
B.1 List of Programs 18
B.2 Formatting of Lists 19

List of Tables

2.1 Example of a table 5

B.1 First meaningless table in Appendix B 19
B.2 Second meaningless table in Appendix B 19

List of Figures

2.1 Example of a figure 5

List of Programs

B.1 Example of the new `program` environment 18
B.2 Second program 18

Preface

Here go all the interesting reasons why you decided to write this thesis.

Chapter 1

Purpose

This document outlines the functionality of the style file (package) csthesis.sty, suitable for writing theses in $\LaTeX\,2_\varepsilon$ at SFU. The style file modifies and generalizes the sfuthesis.sty, designed by Margaret Sharon from ACS. The changes were done for two reasons: 1) to reflect the most recent regulations on SFU theses;[1] and 2) to improve the overall look of the theses. The changes will be illustrated and commented on in Chapter 2; the following section summarizes the main features of the present package.

1.1 Main features of csthesis.sty

- fully conforms to the most recent regulations on SFU theses;
- offers a visually pleasing design;
- supports all optional thesis sections (Dedication, Quotation, Preface, etc.) and simplifies their inclusion;
- supports the `twoside` option of $\LaTeX$ and coordinates it with two-sided printing—this is not allowed for the library copies, but you should definitely consider it for other copies;
- comes with a full documentation, describing all available commands and identifying which parts of the code can be modified and how to achieve specific effects (like adding/removing white space between signatures on Approval page);
- comes with two files, illustrating the use of the package in writing theses (you are reading one of them). The following section describes all the files in the distribution in more detail.

[1] Please refer to http://www.lib.sfu.ca/theses to check for any changes to requirements.

1.2 Distribution

The style file csthesis.sty is accompanied by several supporting files. These are:

- `READ.ME`: the file containing the copying and distribution limitations, the installation instructions, the address where to sent bug reports, and the listing of all the files included in the csthesis bundle;
- `csthesis.ins`: the installation file for csthesis.sty. The bundle comes unpacked and ready-to-use, so you don't need to generate csthesis.sty this way, but the file is included to allow for a more compact, packed version of the bundle.
- `csthesis.dtx`: the documented code for csthesis.sty. It lists all the user commands defined by the style file and explains the code. In particular, it points out the places where the code should be modified in case you run into troubles like not being able to fit the Approval section on one page.
- `thes-short.tex`: an example of a thesis written using csthesis.sty. The thesis is "short" in that it contains only the required parts of the front matter (Abstract, Acknowledgment, Contents) but no optional parts (Dedication, Quotation, List of Figures, List of Tables, List of "other things," Preface). In addition, the thesis does not contain the optional parts of the back matter—appendices and Index.
- `thes-full.tex`: an example of the "full" version of the same thesis, containing all the optional parts from the front matter, several appendices, and Index.
- the `files/` subdirectory: the auxiliary files for `thes-short.tex` and `thes-full.tex`.
- the `doc/` subdirectory: the "printouts" of the documentation for the package and for the two example theses.

You can use either of the two versions of the thesis as a guide in designing your own thesis file—but it's in your interest to take a look at both. The difference between the two versions is accomplished by 1) the switches located right under the beginning of the `document` environment, 2) inclusion/exclusion of the files containing optional thesis parts (Preface, appendices, Index, etc.), and 3) the inclusion of commands for generating the List of "other things" in the preamble of `thes-full.tex`.

The two thesis files demonstrate the use of the commands provided by the csthesis.sty package. If a more thorough explanation of some command(s) is required, consult the `csthesis.dvi` guide.

1.3 How to use csthesis.sty

If you're starting writing from scratch, all you need to do is to put the following two lines in the beginning of your thesis document:

```
\documentclass[11pt]{report}%% or 10pt, or 12pt
\usepackage{csthesis}
```

This makes all the commands defined in csthesis.sty available to you; you can consult the two examples theses and the `csthesis.dvi` to see what commands you have at your disposal and how they should be used.

If you're upgrading from sfuthesis.sty to csthesis.sty, you also need to include the two lines mentioned above. In addition, though, you need to change some things in your thesis document because the two thesis formats are not fully compatible. In particular, you should:

- replace the `\afterpreface` command of sfuthesis.sty with `\lists` and `\beforetext`, inserted in appropriate places (see the example theses);
- in case you're having one or more appendices, issue `\appendix` command after the last chapter of your thesis before the first appendix. In addition, treat each appendix as a chapter, i.e., the main heading of the appendix should use the `\chapter` command.

These should be all changes you need to do—if there is more, let me know (I haven't used sfuthesis.sty, so I am to some extent guessing here based on the coding differences between the two packages).

Chapter 2

Changes from sfuthesis.sty

2.1 Page layout

All the changes in this section are not required by the regulations; they were made for better visual impact.

The four margins were set by sfuthesis.sty to the values recommended in the regulations; these were kept unchanged. The space between the header and the text was slightly reduced and made dependent on the chosen point size of the font (10, 11, or 12)[1]—it now corresponds to one empty line of text.

The space between the text and the footer is set to the result of adding half an empty line to the value of the space between the header and the text. This centers the text vertically on the page with respect to the other material. If, for any reasons, you decide to adjust the vertical placement of the text, only the value of `\headsep` needs to be modified.

The first footnote is pushed slightly lower under the horizontal bar which separates footnotes from the text (see below). In addition, small vertical space separates individual footnotes on the page.[2]

The vertical space before the `paragraph` and `subparagraph` sections was slightly reduced from the original values set by LaTeX. The new values work better for the line spacing used in the text (see Appendix A).

[1]This text is set with 10pt size, which seems to be the best choice overall from the three possibilities.

[2]This is an example of a second footnote on the page. I will make it longer than the first one. As can be seen, the two footnotes are separated by small vertical space.
This is a new paragraph in the second footnote. There is no additional vertical space between paragraphs in a footnote.

2.2 Line spacing

The style file defines three line spacing lengths: `\textstretch`, `\tighttextstretch`, and `\doublestretch`. The first is used for the regular text. The second, which basically corresponds to the single spacingon a typewriter, is used for footnotes (see page 4) and inside tables (see Table 2.1) and figures (see Figure 2.1)—including captions. The last corresponds to the double spacing on a typewriter and is used for the thesis title, as required by the regulations.

This shows that single line spacing
is used inside figures.

Figure 2.1: Example of a figure

This shows	the same thing
for tables.	

Table 2.1: Example of a table

In addition, I recommend using single spacing also for bibliography and index (if included), as it is done in this document.[3] The corresponding files `files/bibl.tex` and `files/ind.tex` show how to specify that single spacing be used.

2.3 Front matter

2.3.1 Title page

The title itself is set in bold face, capitalized, and double spaced. The vertical white space was changed from fixed to stretchable (and partially shrinkable) to allow for a better fit of the material—titles have varying amouns of lines and people can have several previous degrees.

2.3.2 Approval page

Essentially the same, only made easier to modify if the text doesn't fit on a single page or if you want to change the amount of white space between signatures.

[3]This is now the accepted format for references.

2.3.3 Dedication and Quotation pages

The csthesis.sty package has a built-in support for optional Dedication and Quotation pages. Both dedication and quotation are typeset flushed right, dedication in *italics* and quotation *slanted* (both are illustrated in the full version of this document). The pages are not titled, but they are numbered, and a corresponding entry is added to the Contents table.

There are no constraints on the look of these pages imposed by the regulations, you can change the design as you see fit.

2.3.4 Contents

The table of Contents was reverted back to the original design (for report.sty) from LaTeX. There are no specific requirements in the regulations as to how the table should look like, but all the changes made to it in sfuthesis.sty—reducing white space between chapters, putting dotted lines on the chapter level, or redefining the behavior of the `\appendix` command—looked decidedly awful and so I got rid of them.

2.3.5 Preface

The regulations allow for a Preface or Foreword section, which could contain a short blurb about "why the author came to study the subject of the thesis." It's probably useless because nobody will want to write anything like that, but it's incorporated in the package for completeness. The optional inclusion of Preface necessitated the modification of the `\afterpreface` command of sfuthesis.sty, which was replaced by two commands: `\lists`, and `\beforetext`.

2.4 Back matter

2.4.1 Appendices

As mentioned in Section 2.3.4, sfuthesis.sty redefines the `\appendix` command of LaTeX. Again, there does not seem to be any reason to do that (the regulations are silent on the formatting of appendices); and the original design looks better. Thus, the present style reverts to LaTeX's original design, where appendices are treated as chapters and numbered alphabetically. (The change in numbering is accomplished by issuing the `appendix` command before the text of the first appendix.)

2.4.2 Bibliography

The csthesis.sty package does not influence the typesetting of Bibliography directly, but—as I said in Section 2.2—I recommend changing the line spacing from normal to single, as was done in `files/bibl.tex`. See the (fake) Bibliography of this document as an example of the result.

2.4.3 Index

The same comment applies to the optional Index section (see `files/ind.tex` for how to adjust the line spacing and the Index section of the full version of this document for the result).

Chapter 3

introduction

motivation , primarily problem to solve
we need jobs to match skills we need to detect skills
real motivation secondary problem that we are actually solving
how to detect skills. why it is challenging and different than usual keyword extraction problems

3.1 literature review

3.2 what are the algorithms in keyword extraction

NLP introduction The automatic analysis of
text involves a deep understanding of
natural language by machines, a reality
from which we are still very far off [3]. Hitherto, online information
retrieval, aggregation, and processing
have mainly been based on algorithms
relying on the textual representation of
web pages. Such algorithms are very
good at retrieving texts, splitting them
into parts, checking the spelling and
counting the number of words. When
it comes to interpreting sentences and
extracting meaningful information, however, their capabilities are known to
be very limited. NLp in fact, requires high level symbolic capabilities(Dyer 1994) including:
1-creation and propagation of dynamic

bindings; 2-manipulation of recursive, constituent structures, acquisition and access of lexical, semantic, and episodic memories; control of multiple learning/process- ing modules and routing of informa- tion among such modules; grounding of basic-level language constructs (e.g., objects and actions) in perceptual/motor experiences; representation of abstract concepts. All such capabilities are required to shift from mere NLP to what is usually referred to as natural language under- standing (Allen, 1987). Today, most of the existing approaches are still based on the syntactic representation of text, a method that relies mainly on word co- occurrence frequencies. Such algorithms are limited by the fact that they can pro- cess only the information that they can see. As human text processors, we do not have such limitations as every word we see activates a cascade of semantically related concepts, relevant episodes, and sensory experiences, all of which enable the completion of complex NLP taskssuch as word-sense disam- biguation, textual entailment, and semantic role labelingin a quick and effortless way.[3]

what is tagging? Tagging is the process of labeling web resources based on their content. Each label, or tag, corresponds to a topic in a given document. Unlike metadata assigned by authors, or by professional indexers in libraries, tags are assigned by end- users for organizing and sharing information that is of interest to them. The organic system of tags assigned by all users of a given web platform is called a folksonomy. [17]

what is keyphrase? what is keyphrase extraction?what is the goal of extracting keyphrases? Many journals ask their authors to provide a list of keywords for their articles. We call these keyphrases, rather than keywords, because they are often phrases of two or more words, rather than single words. We define a keyphrase list as a short list of phrases (typically five to fifteen noun phrases) that capture the main topics discussed in a given document. This paper is concerned with the automatic extraction of keyphrases from text. Keyphrases are meant to serve multiple goals. For example, (1) when they are printed on the first page of a journal article, the goal is summarization. They enable the reader to quickly determine whether the given article is in the readers fields of interest. (2) When they are printed in the cumulative index for a journal, the goal is indexing. They enable the reader to quickly find a relevant article when the reader has a specific need. (3) When a search engine form has a field labelled keywords, the goal is to enable the reader to make the search more precise. A search for documents that match a given query term in the keyword field will yield a smaller, higher quality list of hits than a search for the same term in the full text of the documents. Keyphrases can serve these diverse goals and others, because the goals share the requirement for a short list of phrases that captures the main topics of the documents. We define automatic keyphrase extraction as the automatic selection of important, topical phrases from within the body of a document. Automatic keyphrase extraction is a special case of the more general task of automatic keyphrase generation, in which the generated phrases do not necessarily appear in the body of the given document.

Automatic keyphrase extraction concerns the automatic selection of important and topical phrases from the body of a document (Turney, 2000). In other words, its goal is to extract a set of phrases that are related to the main topics discussed in a given document (Tomokiyo and Hurst, 2003; Liu et al., 2009b; Ding et al., 2011; Zhao et al., 2011).[15]

what is the differences between indexing and keyphrase list

We discuss related work by other researchers in Section 3. The most closely related work involves the problem of automatic index generation (Fagan 1987, Salton 1988, Ginsberg 1993, Nakagawa 1997, Leung and Kan 1997). One difference between keyphrase extraction and index generation is that, although keyphrases may be used in an index, keyphrases have other applications, beyond indexing. Another difference between a keyphrase list and an index is length. Because a keyphrase list is relatively short, it must contain only the most important, topical phrases for a given document. Because an index is relatively long, it can contain many less important, less topical phrases. Also, a keyphrase list can be read and judged in seconds, but an index might never be read in its entirety. Automatic keyphrase extraction is thus a more demanding task than automatic index generation. [18]

diffrence between keyphrase extraction and information extraction Keyphrase extraction is also distinct from information extraction, the task that has been studied in depth in the Message Understanding Conferences (MUC-3 1991, MUC-4 1992, MUC-5 1993, MUC-6 1995). Information extraction involves extracting specific types of task-dependent information. For example, given a collection of news reports on terrorist attacks, information extraction involves finding specific kinds of information, such as the name of the terrorist organization, the names of the victims, and the type of incident (e.g., kidnapping, murder, bombing). In contrast, keyphrase extraction is not specific. The goal in keyphrase extraction is to produce topical phrases, for any type of factual, prosaic document. [18]

in [18] automatic keyphrase extraction is approach as a supervised learning task. We treat a document as a set of phrases, which must be classified as either positive or negative examples of keyphrases. This is the classical machine learning problem of learning from examples. In Section 5, we describe how we apply the C4.5 decision tree induction algorithm to this task (Quinlan 1993). There are several unusual aspects to this classification problem. For example, the positive examples constitute only 0.2The experiments in this paper use five collections of documents, with a combined total of 652 documents. The collections are presented in Section 4. In our first set of experiments (Section 6), we evaluate nine different ways to apply C4.5. In preliminary experiments with the training documents, we found that bagging seemed to improve the performance of C4.5 (Breiman 1996a, b, Quinlan 1996). Bagging works by generating many different decision trees and allowing them to vote on the classification of each example. We experimented with different numbers of trees and different techniques for sampling the training data. The experiments support the hypothesis that bagging improves the performance of C4.5 when applied to automatic keyphrase extraction. During our experiments with C4.5, we came to believe that a specialized algorithm, developed specifically for learning to extract keyphrases, might achieve better results than a general-purpose learning algorithm, such as C4.5. Section 7 introduces the GenEx algorithm. GenEx is a hybrid of the Genitor steady-state genetic algorithm (Whitley 1989) and the Extractor parameterized keyphrase extraction algorithm (Turney 1997, 1999).4 Extractor works by assigning a numerical score to the phrases in the input document. The final output of Extractor is essentially a list of the highest scoring phrases. The behaviour of the scoring function is determined by a dozen numerical parameters. Genitor tunes the setting of these parameters, to maximize the performance of Extractor on a given set of training examples. The second set of experiments (Section 8) supports the hypothesis that a specialized algorithm (GenEx) can generate better keyphrases than a general-purpose algorithm (C4.5). Both algorithms incorporate significant amounts of domain knowledge, but we avoided embedding specialized procedural knowledge in our application of C4.5. It appears that some degree of specialized procedural knowledge is necessary for automatic keyphrase extraction. The third experiment (Section 9) looks at subjective human evaluation of the quality of the keyphrases produced by GenEx. On average, about 80

Importance of keyphrase extraction and its applications Document keyphrases have enabled fast and ac- curate searching for a given document from a large text collection, and have exhibited their potential in improving many natural language processing (NLP) and information retrieval (IR) tasks, such as text summarization (Zhang et al., 2004), text categorization (Hulth and Megyesi, 2006), opin- ion mining (Berend, 2011), and document index- ing (Gutwin et al., 1999).[15]
Owing to its importance, automatic keyphrase extraction has received a lot of attention. However, the task is far from being solved: state-of-the-art performance on keyphrase extraction is still much lower than that on many core NLP tasks (Liu et al., 2010).[15]

textbfwhat are the difficulties involved in keyphrase extraction Automatic keyphrase extraction systems have been evaluated on corpora from a variety of sources ranging from long scientific publications to short paper abstracts and email messages. Ta- ble 1 presents a listing of the corpora grouped by their sources as well as their statistics.1 There are at least four corpus-related factors that affect the difficulty of keyphrase extraction. Length The difficulty of the task increases with the length of the input document as longer doc- uments yield more candidate keyphrases (i.e., phrases that are eligible to be keyphrases (see Sec- tion 3.1)). For instance, each Inspec abstract has on average 10 annotator-assigned keyphrases and 34 candidate keyphrases. In contrast, a scientific paper typically has at least 10 keyphrases and hun- dreds of candidate keyphrases, yielding a much bigger search space (Hasan and Ng, 2010). Conse- quently, it is harder to extract keyphrases from sci- entific papers, technical reports, and meeting tran- scripts than abstracts, emails, and news articles. Structural consistency In a structured doc- ument, there are certain locations where a keyphrase is most likely to appear. For instance, most of a scientific papers keyphrases should appear in the abstract and the introduction. While structural information has been exploited to extract keyphrases from scientific papers (e.g., title, section information) (Kim et al., 2013), web pages (e.g., metadata) (Yih et al., 2006), and chats (e.g., dialogue acts) (Kim and Baldwin, 2012), it is most useful when the documents from a source exhibit structural similarity. For this reason, structural information is likely to facilitate keyphrase extrac- tion from scientific papers and technical reports because of their standard format (i.e., standard sections such as abstract, introduction, conclusion, etc.). In contrast, the lack of structural consistency in other types of structured documents (e.g., web pages, which can be blogs, forums, or reviews)may render structural information less useful. Topic change An observation commonly ex- ploited in keyphrase extraction from scientific ar- ticles and news articles is that keyphrases typically appear not only at the beginning (Witten et al., 1999) but also at the end (Medelyan et al., 2009) of a document. This observation does not neces- sarily hold for conversational text (e.g., meetings, chats), however. The reason is simple: in a conversation, the topics (i.e., its talking points) change as the interaction moves forward in time, and so do the keyphrases associated with a topic. One way to address this complication is to detect a topic change in conversational text (Kim and Baldwin, 2012). However, topic change detection is not al- ways easy: while the topics listed in the form of an agenda at the beginning of formal meeting tran- scripts can be exploited, such clues are absent in casual conversations (e.g., chats). Topic correlation Another observation com- monly exploited in keyphrase extraction from scientific articles and news articles is that the keyphrases in a document are typically related to each other (Turney, 2003; Mihalcea and Tarau, 2004). However, this observation does not nec- essarily hold for informal text (e.g., emails, chats, informal meetings, personal blogs), where people can talk about any number of potentially uncorre- lated topics. The presence of uncorrelated topics implies that it may no longer be possible to exploit relatedness and therefore increases the difficulty of keyphrase extraction.

what are the different approaches to this problem? two general steps in all approaches A keyphrase extraction system typically operates in two steps: (1) extracting a list of words/phrases that serve as candidate keyphrases using some heuristics (Section 3.1); and (2) determining which of these candidate keyphrases are correct keyphrases using supervised (Section 3.2) or un- supervised (Section 3.3) approaches.[15]

approaches to solve the first step:extracting candidates As noted before, a set of phrases and words is typically extracted as candidate keyphrases using heuristic rules. These rules are designed to avoid spurious instances and keep the number of candi- dates to a minimum. Typical heuristics include (1) using a stop word list to remove stop words (Liu et al., 2009b), (2) allowing words with certain part- of-speech tags (e.g., nouns, adjectives, verbs) to be candidate keywords (Mihalcea and Tarau, 2004; Wan and Xiao, 2008b; Liu et al., 2009a), (3) al- lowing n-grams that appear in Wikipedia article titles to be candidates (Grineva et al., 2009), and (4) extracting n-grams (Witten et al., 1999; Hulth, 2003; Medelyan et al., 2009) or noun phrases (Barker and Cornacchia, 2000; Wu et al., 2005) that satisfy pre-defined lexico-syntactic pattern(s) (Nguyen and Phan, 2009).[15] Many of these heuristics have proven effective with their high recall in extracting gold keyphrases from various sources. However, for a long docu- ment, the resulting list of candidates can be long. Consequently, different pruning heuristics have been designed to prune candidates that are un- likely to be keyphrases (Huang et al., 2006; Kumar and Srinathan, 2008; El-Beltagy and Rafea, 2009; You et al., 2009; Newman et al., 2012).[15]

approaches for the second step:choose correct candidates amongst all candidtes Supervised approaches: //can be copied form [15]

unsupervise approaches:

1- graph based Ranking

2- Topic-Based Clustering

3- Simultaneous Learning 4- Language Modeling [15]

however here we focus on the graphbased methods. **Focuse on Graph Based ranking WHy should we create a graph of tags and why intuitively it works?(rewrite the concept)**

Intuitively, keyphrase extraction is about finding the important words and phrases from a document. Traditionally, the importance of a candidate has often been defined in terms of how related it is to other candidates in the document. Informally, a candidate is important if it is related to (1) a large number of candidates and (2) candidates that are important. Researchers have computed relatedness between candidates using co-occurrence counts (Mihalcea and Tarau, 2004; Matsuo and Ishizuka, 2004) and semantic relatedness (Grineva et al., 2009), and represented the relatedness information collected from a document as a graph (Mihalcea and Tarau, 2004; Wan and Xiao, 2008a; Wan and Xiao, 2008b; Bougouin et al., 2013).[15] The basic idea behind a graph-based approach is to build a graph from the input document and rank its nodes according to their importance using a graph-based ranking method (e.g., Brin and Page (1998)). Each node of the graph corresponds to a candidate keyphrase from the document and an edge connects two related candidates. The edge weight is proportional to the syntactic and/or semantic relevance between the connected candidates. For each node, each of its edges is treated as a vote from the other node connected by the edge. A nodes score in the graph is defined recur- sively in terms of the edges it has and the scores of the neighboring nodes. The top-ranked candidates from the graph are then selected as keyphrases for the input document. TextRank (Mihalcea and Ta- rau, 2004) is one of the most well-known graph- based approaches to keyphrase extraction. This instantiation of a graph-based approach overlooks an important aspect of keyphrase ex- traction, however. A set of keyphrases for a doc- ument should ideally cover the main topics dis- cussed in it, but this instantiation does not guaran- tee that all the main topics will be represented by the extracted keyphrases. Despite this weakness, a graph-based representation of text was adopted by many approaches that propose different ways of computing the similarity between two candidates.[15]

supervised
unsupervised
graph-based algorithms

what works and what fails
Automatic keyphrase extraction

3.3 our approach our algorithm

our algorithm a bit of introduction
data acquisition
data cleaning
the results

3.4 visualization of trends

3.5 conclusion

Bibliography

[1] Hiyan Alshawi. *Memory and Context for Language Interpretation*. Studies in Natural Language Processing. Cambridge University Press: Cambridge, New York, 1987.

[2] Nicholas Asher. From discourse macro-structure to micro-structure and back again: Discourse semantics and the focus/background distinction. In Hans Kamp and Barbara H. Partee, editors, *Proceedings of the workshops on Context Dependence in the Analysis of Linguistic Meaning*, volume 1: Papers, pages 21–51. IMS, University of Stuttgart, 1995.

[3] E. Cambria and B. White. Jumping nlp curves: A review of natural language processing research [review article]. *Computational Intelligence Magazine, IEEE*, 9(2):48–57, May 2014. 8, 9

[4] Gennaro Chierchia. *Dynamics of Meaning: Anaphora, Presupposition and the Theory of Grammar*. The University of Chicago Press: Chicago, London, 1995.

[5] Herbert H. Clark and Susan E. Haviland. Comprehension and the given–new contract. In Roy O. Freedle, editor, *Discourse Production and Comprehension*, number 1 in Discourse Processes: Advances in Research and Theory, pages 1–40. Ablex Publishing Corporation: Norwood, NJ, 1977.

[6] Östen Dahl. Topic-comment structure revisited. In Östen Dahl, editor, *Topic and Comment, Contextual Boundness and Focus*, number 6 in Papers in Textlinguistics, pages 1–24. Helmut Buske Verlag: Hamburg, 1974.

[7] Paul Dekker and Herman Hendriks. Files in focus. In Elisabet Engdahl, editor, *Integrating Information Structure into Constraint-based and Categorial Approaches*, ESPRIT Basic Research Project 6852, Dynamic Interpretation of Natural Language, DYANA-2 Deliverable R1.3.B, pages 29–37. ILLC, University of Amsterdam, 1994.

[8] Elisabet Engdahl and Enric Vallduví. Information packaging and grammar architecture: A constraint-based approach. In Elisabet Engdahl, editor, *Integrating Information Structure into Constraint-based and Categorial Approaches*, ESPRIT Basic Research Project 6852, Dynamic Interpretation of Natural Language, DYANA-2 Deliverable R1.3.B, pages 41–79. ILLC, University of Amsterdam, 1994.

[9] Nomi Erteschik-Shir. *The Dynamics of Focus Structure*. Number 84 in Cambridge Studies in Linguistics. Cambridge University Press: Cambridge, New York, Melbourne, 1997.

[10] L. T. F. Gamut. *Logic, Language, and Meaning*, volume 1: Introduction to Logic. The University of Chicago Press: Chicago, London, 1991.

[11] L. T. F. Gamut. *Logic, Language, and Meaning*, volume 2: Intensional Logic and Logical Grammar. The University of Chicago Press: Chicago, London, 1991.

[12] Barbara J. Grosz and Candace L. Sidner. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12:175–204, 1986.

[13] Jeanette K. Gundel. *The Role of Topic and Comment in Linguistic Theory*. PhD thesis, University of Texas, 1974. Published by the Indiana University Linguistics Club, Bloomington, 1977.

[14] Jeanette K. Gundel. Universals of topic–comment structure. In Michael Hammond, Edith A. Moravcsik, and Jessica R. Wirth, editors, *Studies in Syntactic Typology*, volume 17 of *Typological Studies in Language*, pages 209–239. John Benjamins: Amsterdam, Philadelphia, 1988.

[15] Kazi Saidul Hasan and Vincent Ng. Automatic keyphrase extraction: A survey of the state of the art. 10, 11, 12, 13, 14

[16] Herman Hendriks and Paul Dekker. Links without location. In Paul Dekker and Martin Stokhof, editors, *Proceedings of the Tenth Amsterdam Colloquium*, pages 339–358. ILLC—Dept. of Philosophy, University of Amsterdam, 1996.

[17] Olena Medelyan, Eibe Frank, and Ian H. Witten. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1318–1327, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. 9

[18] PeterD. Turney. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2(4):303–336, 2000. 10

[19] Kai von Fintel. A minimal theory of adverbial quantification. In Hans Kamp and Barbara H. Partee, editors, *Proceedings of the workshops on Context Dependence in the Analysis of Linguistic Meaning*, volume 1: Papers, pages 153–193. IMS, University of Stuttgart, 1995.

Appendix A

Appendices:Sectioning

Appendices appear in the Contents table on the level of chapters and numbered alphabetically starting from A. You can include normal sectioning commands (notice the vertical spacing between different levels):

A.1 Section

A section is numbered and appears in the Contents.

A.1.1 Subsection

Same holds for a subsection.

Subsubsection

But not for other sectioning commands, unless you adjust the `secnumdepth` and `tocdepth` counters to higher values (by default both are set to `2`).

Paragraph A paragraph is inlined.

Subparagraph A subparagraph is inlined and indented.

Appendix B

Illustrating Lists

This appendix is present only in the full version of the thesis. It illustrates in more detail the formatting of the Lists of Tables, Figures, etc. In addition, it also illustrates the use of the “other list” facility of csthesis.sty. Let us start with the latter.

B.1 List of Programs

In the preamble of `thes-full.tex`, a new type of a floating environment—`program`—is defined, together with the `\otherlist` command for typesetting the List of Programs. The list is formatted in the same way as the lists of Figures and Tables, and an appropriate entry is added into Contents.

Because programs are defined here as floating environments, they are typeset with the same (tighter) line-spacing as figures and tables:

This shows that single line spacing
is used inside programs.

Program B.1: Example of the new `program` environment

A second example of a program environment.

Program B.2: Second program

B.2 Formatting of Lists

The tables (figures, programs, etc.) are organized and sorted by chapters (appendices), with additional small vertical space separating the chapter blocks. To illustrate this, I include two new tables here (see List of Tables, etc., in the beginning of the thesis for the result):

To be silly or not to be silly
that is the question!

Table B.1: First meaningless table in Appendix B

$$F = nd^2$$

Table B.2: Second meaningless table in Appendix B