Department of Computer Science

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

Hamilton, New Zealand

Human-competitive automatic topic indexing

by

Olena Medelyan

This thesis is submitted in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy in Computer Science
at The University of Waikato

July 2009

Abstract

Topic indexing is the task of identifying the main topics covered by a document. These are useful for many purposes: as subject headings in libraries, as keywords in academic publications and as tags on the web. Knowing a document's topics helps people judge its relevance quickly. However, assigning topics manually is labor intensive. This thesis shows how to generate them automatically in a way that competes with human performance.

Three kinds of indexing are investigated: term assignment, a task commonly performed by librarians, who select topics from a controlled vocabulary; tagging, a popular activity of web users, who choose topics freely; and a new method of keyphrase extraction, where topics are equated to Wikipedia article names. A general two-stage algorithm is introduced that first selects candidate topics and then ranks them by significance based on their properties. These properties draw on statistical, semantic, domain-specific and encyclopedic knowledge. They are combined using a machine learning algorithm that models human indexing behavior from examples.

This approach is evaluated by comparing automatically generated topics to those assigned by professional indexers, and by amateurs. We claim that the algorithm is "human-competitive" because it chooses topics that are as consistent with those assigned by humans as their topics are with each other. The approach is generalizable, requires little training data and applies across different domains and languages.

Acknowledgements

Ian Witten deserves the foremost mention in these acknowledgements, because without him this PhD would not have been possible. Nearly five years ago I came to the University of Waikato as a Ukrainian/German exchange student with an unpronounceable name and a poor knowledge of English. Like with many other students, Ian has been a patient, encouraging and supporting teacher, who helped me to bring out the best possible of my potential.

Ian, I consider myself extremely lucky to have been invited to come back for a PhD with you. The last three years have shaped me into who I am and opened many new opportunities for my future life. Thanks for teaching me to be patient and focused in the research, for encouraging the highly rewarding voluntary work and for your continuous support in dealing with difficulties in professional and personal life. Thank you and Pam for inviting us to parties and dinners at your place, for organizing the wonderful Waiheke holiday and for letting me stay in your house in the last weeks of my PhD. Please remember that you have a free life subscription to fire performances and yoga lessons.

I would also like to thank my other supervisor Eibe Frank, who was always promptly available for help whenever I knocked on his door. Thank you, Eibe, for sharing your machine learning expertise and for developing the Kea algorithm, which provided a great basis for Maui's architecture.

I am very proud to have been a part of the multi-cultural group of Ian's PhD students and to have collaborated with other Waikato researchers like Cathy Legg and Gian Perrone. Thank you, girls and guys, for listening to my progress, for helping with practicing presentations and interviews, and for your great company. I look forward to staying in touch and hearing about your future successes!

I would like to thank Google for offering the scholarship for my PhD and one of their employees and Ian's former student, Craig Nevill-Manning, for helping me with organizing the extremely valuable internship at Google New York. Craig,

thanks also for providing the best possible motivation with your post-PhD success and for staying in touch with us, Waikato students, as an unofficial mentor.

Knowing that other people work on similar problems has been incredibly motivating. Through conference trips—sponsored by Google and BuildIT—I met many talented researchers who influenced my work. Thanks to Alberto Pepe and Andras Csomai for exchanging research ideas, and to Dmitry Lizorkin, Alan Aronson and Annette Holtkamp for sharing their data sets and results with me.

Since I am not a native speaker, proofreading help from friends and colleagues was needed. Ian once noted that I use exactly the right amount of "a"s and "the"s, but all in the wrong places. My knowledgeable and experienced proofreaders helped not only with grammar, but also with the organization and clarity of this thesis. A huge thanks to Anna Huang, Craig Schock, David Milne, Gian Perrone, James Gorman, Kathryn Hempstalk, Kerry Hannan, Nathan Holmberg, Rob Akscyn, Shaoqun Wu, Tom Manschreck, and Tudor Leu.

I have been lucky to do my PhD at the same time as David Milne, whose name frequently re-occurs in this thesis because our topics were closely related. Several parts of this thesis (e.g., Sections 4.2.1 and 5.1.2) are the result of our collaboration. Although at times dissecting the topics we shared into two PhDs has been difficult, the benefits were greater. Thank you, Dave, for being a valuable discussion partner, a patient co-author, and for building the Wikipedia Miner, the coolest tool on SourceForge. Also, thank you for being a great friend.

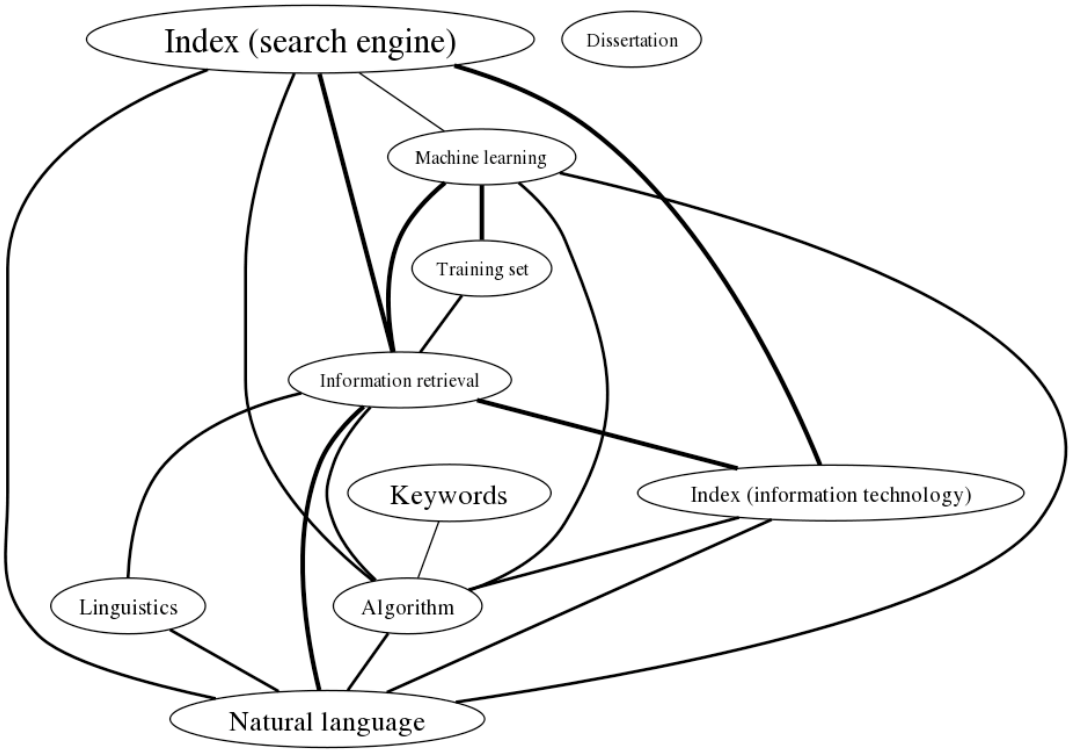
I also want to thank the many other friends around the world who made the past years into a special and unforgettable time. Without committing to this PhD I wouldn't have met Nathan, who, as it happens, gave me the reason to finish it on time. Thank you so much for all the support and wonderful distractions of the last months.

Finally, a huge thanks to my parents who have taught me the value of a good education. Thanks to them I was the first person in the class to own a computer, which eventually lead to doing a PhD in Computer Science. I guess I should be careful with what to buy for my children one day. Mum, you have been the driving force behind my career and my dearest friend. I love you and dedicate this work to you.

Table of contents

List of tables

List of figures



This diagram shows the main topics extracted from the text of Chapter 1 using the methods developed in this thesis, and has been generated entirely automatically (see Section 6.6). Font size indicates significance and line thickness corresponds to the strength of the semantic relations between topics. Similar diagrams precede each chapter.

Chapter 1
Introduction

The exciting idea that one day computers will understand human language remains an elusive dream, far from reality. Tasks that require understanding natural language, such as machine translation and text summarization, have been researched for many decades, but rarely solved at a level comparable to human performance. Researchers must deal not only with the incredible complexity of our language, but also with the highly subjective nature of the tasks. Judging whose translation is more accurate or whose summary is better is sometimes even harder than producing one.

Topic indexing is different. It is an easier task. The question "What is this document about?" does assume some understanding of natural language, but restricts the answer to a handful of phrases that describe the document's main topics. Deep understanding is not necessarily required: even human indexers tend to skim the text rather than read and understand it. Topic indexing is also easier to evaluate. Those topics on which the majority of people agree must surely be the right ones. Multiple votes transform a subjective view into an objective answer.

This thesis investigates traditional and modern forms of topic indexing and develops a method called Maui (multi-purpose automatic topic indexing) for identifying the main topics automatically from document text. Maui is tested on data sets from various sources, and evaluated against human indexers of different skill levels. The algorithm is described as "human-competitive" because it matches the agreement achieved by people on the same data.

1.1 Motivation

In physical and digital libraries, professional indexers are employed to organize documents based on their main topics in a way that facilitates access by users. Categories and subject headings are used to describe topics. They most commonly originate from a pre-defined controlled vocabulary, where equivalent phrases referring to the same concept are grouped under the same "topic". When no controlled vocabulary is used, topics are expressed as freely chosen keywords and keyphrases. The tasks involving selecting appropriate topics according to pre-defined cataloguing rules (if such exist) can be summarized under the term *topic indexing*.

Ideally, professional indexers should see the document as a whole, understand the emphasis of its parts, and know its potential readers (Bonura, 1994). Topic indexing is a labor-intensive and time-consuming process that became infeasible with the explosion of electronically available information, offline and on the web. The majority of electronic holdings in any document repository lack subject headings and keyphrases. They remain restricted to basic metadata like author and title. Retrieving all documents on a particular topic simply by searching through authors and titles is hardly possible, countless searches would be required to capture potential ways of how this topic may be expressed in a title—and many documents would remain undiscovered because their titles are too figurative. The need for subject headings and keyphrases is apparent. Nowadays libraries spend more money on employees than on new books or journal subscriptions (Hilberer, 2003) and would clearly benefit from methods to automate the indexing process.

Full text search, the primary means of navigation on the web, is directly related to topic indexing. No Internet user spends a day without looking for documents by formulating a search query describing their main topics. Search engines respond to user queries by matching them against the full text index of the repository, which lists every single word as it appears in a document. With billions of web sites, most searches inevitably result in plenty of matches, which search engines rank based on a secret mixture of word occurrence statistics, anchor text and web graph analysis.

Tagging is a recent attempt to address web search from a different angle, using metadata contributed by users. It originates in the traditional subject indexing performed in libraries. Tags, like keyphrases, are single words and phrases that describe the main topics of websites, blogs, research papers, as well as multimedia content, and are assigned by the authors themselves or, collaboratively, by readers. The multitude of tags represents a "folksonomy" that, like a catalog in a library, allows users to browse through topics of the collection.

Tag folksonomies are useful but have several limitations. They are inconsistent, because there is no control over what terms may be added and what are the preferred tags to describe certain topics. They are unstructured, because relations between individual tags are not encoded. Browsing is driven by co-occurrence of tags within the same tag sets and items, rather than by their topical relatedness. Tagging is popular in domains like blogs and file sharing, where users are motivated and active, but tags would be highly useful in many other areas. However, tagging is not yet widespread, because the assignment of tags is not an easy task. Automatic tag suggestion tools are believed to improve the efficiency and consistency of human tagging (Brooks and Montanez, 2006), which might facilitate the spread of this useful approach to web search.

A document's main topics are not only an important form of metadata in libraries and on the web; they also have a variety of indirect uses. Keyphrases improve the performance of tasks such as text clustering and categorization (Jones and Mahoui, 2000), content-based retrieval and topic search (Arampatzis *et al.*, 1998), automatic text summarization (Barker and Cornacchia, 2000), thesaurus construction (Paynter *et al.*, 2000), search results representation (Hulth, 2004), and navigation (Gutwin *et al.*, 1998). The effectiveness and utility of these tasks depend not only on the availability of keyphrases, but also on their quality. Ideally, topics should be provided by human experts or be of comparable quality.

The importance of automatic topic indexing is evident. In libraries and any centralized document repositories, automatic indexing would lift a significant burden from librarians' shoulders. On the web, tag suggestion tools would guide users to more useful documents. In natural language processing, automatically assigned keyphrases would provide a highly informative semantic dimension in

keyphrases would provide a highly informative semantic dimension in document representation that would benefit new applications.

1.2 Thesis statement

This thesis claims that

> *With access to domain and general semantic knowledge,*
> *computers can index as well as humans.*

Computer systems will hardly ever be able to compete with human performance in tasks that require understanding the meaning of human language. However, indexing tasks potentially do not require deep understanding. Our main goal is, given a document, to find a handful of concepts that describe its topics. The depth of the intermediate analysis is not important as long as the results are competitive with human performance.

The main topics of a document are usually expressed as noun phrases that may consist of a single or a multi-word noun (e.g. *biology* or *computer science*). More specific topics may be expressed with a modifying adjective (e.g. *theoretical computer science*) or a prepositional phrase (e.g. *biology of gender*). Topics are referred to by different names depending on the task. The classical names used by librarians are *subject headings*, *index terms* and *descriptors*, all of which are synonyms. In digital libraries, *keywords* and *keyphrases* are more common, the former referring to single words and the latter to multi-word phrases. The main difference to subject headings is that keyphrases are freely chosen, whereas subject headings are derived from a *controlled vocabulary*. Recently, a new synonym for keyphrases has been created—*tags*, perhaps to be in line with other new English one-syllable words like *web* and *blog*. Chapter 2 investigates different types of topic indexing and their applications in more detail.

Given the techniques developed in this thesis, we investigate whether computers can identify topics *as well as humans*. First, human performance is measured, and then the algorithm is compared against the humans using the same metric. The main criterion here is indexing quality, as opposed to, for example, indexing

speed. Speed comparisons are expected to favor algorithms; therefore, the more interesting question is whether the algorithms can achieve the same *quality* as humans.

Analysis of indexing performed by professionals shows that the correctness of assigned topics is very subjective (Chapter 4). Therefore, instead of relying on the judgments of one human, *multiply indexed documents* are used, where topic choices of several indexers are available. The goal of the algorithm is to match topics on which these indexers have agreed. The gold standard is the average *indexing consistency* within a group of human indexers, computed using traditional methods (Section 2.2). In these terms, an algorithm that matches or outperforms human consistency on a wide variety of data sets indexes as well as humans.

People acquire indexing skills through training and work experience. In an algorithm this process can be simulated using *machine learning*. If manually specified topics are given, the document collection serves as a *training set* and all the documents' candidate topics are seen as positive and negative examples. A *supervised* learning scheme is applied to analyze values of properties that distinguish manually assigned topics from other document phrases. Then these observations are applied to new, unseen documents. It is essential to determine a useful set of such properties, also called *features*. In this thesis, properties are chosen to represent different types of knowledge about topics, which are typically chosen by human indexers.

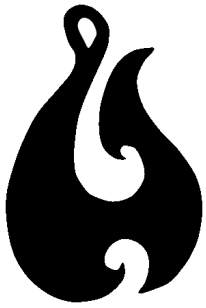
The thesis statement specifies that access to domain and semantic knowledge is required to achieve human-competitive performance. *Domain knowledge* is specific to the indexing of a particular document collection, e.g. a set of computer science technical reports or publications in bioinformatics. For example, some terms are commonly chosen as topics in a particular domain and should therefore be given preference in the automatic selection process. Statistical methods can be applied to deduce further domain-specific characteristics useful for topic indexing. The commonly used TF×IDF statistics (Salton and Buckley, 1988) identify terms that are frequently used across an entire collection and those that are specific for a particular document.

Semantic knowledge is required to understand the meaning of a document's concepts and their linguistic characteristics. This knowledge is not specific to a particular domain or document collection. For example, semantic relatedness is important when deducing the meaning of ambiguous words by relating them to the context. Furthermore, concepts that are semantically related to other concepts in the document are more likely to represent the main topics. An algorithm can derive such knowledge from various sources, including controlled vocabularies used for indexing, contain manually encoded semantic relations between terms. Linguistic resources like WordNet (Fellbaum, 1998) can also be used for the same purpose. Large corpora or the entire web can be mined for semantic data such as co-occurrence statistics.

A popular source of linguistic knowledge is the online encyclopedia Wikipedia (Medelyan *et al.*, 2009). Apart from being an easily accessible subset of the web and a large multi-lingual corpus of clearly formulated definitions, it exhibits structural elements that bear useful semantic knowledge. Individual articles correspond to common language concepts and named entities, disambiguation pages list meanings of ambiguous words, and hyperlinks between pages indicate semantic relatedness. Wikipedia's hypertext can be mined for words that are likely to describe certain concepts. Chapter 5 describes how different types of knowledge derived from Wikipedia, manually encoded thesauri, and corpora can be used in automatic topic indexing.

1.3 History

Although this thesis is an independent piece of research, it represents a continuation of the work in my Master's thesis (Medelyan, 2005). The Master's thesis itself was build on experiments in automatic keyphrase extraction at the University of Waikato, which produced the Kea algorithm (Witten *et al.*, 1999; Frank *et al.*, 1999). The main goal of that thesis was to apply Kea's methods to keyphrase indexing with a controlled vocabulary and to investigate the usefulness of semantic features. The outcome was Kea++, an extension of Kea for indexing agricultural documents with topics from a domain-specific thesaurus.



Maui is an algorithm for **m**ulti-purpose **au**tomatic topic **i**ndexing. It is named after the Polynesian mythological hero and demi-god, known in the Māori mythology as Māui.

Māui is famous for fishing out the North Island of New Zealand with a hook made out of his jaw-bone. The logo of the algorithm (left) depicts this fishing hook, a shape that is commonly used in traditional jewelry carving.

Māui sometimes would transform himself into different kinds of birds to perform many of his exploits. Similarly, the Maui algorithm assimilates two software tools named after New Zealand native birds Kea and Weka.

Figure 1.1 The legend of Maui

The aim of the Master's thesis was to answer three questions. First, can keyphrase extraction be improved by using a controlled vocabulary? Indeed, mapping document phrases to terms in a controlled vocabulary covered just as many manually assigned topics, while avoiding significant noise. Second, can semantic knowledge enhance indexing performance? In fact, a new feature based on semantic relations encoded in the vocabulary resulted in performance gain. Third, can automatic keyphrase indexing perform as well as professional indexers? The final algorithm performed worse than human indexers, and thus, unlike the first two, this third hypothesis remained unproven.

This PhD work builds on the positive outcomes of the Master's thesis. It is deeper, in that it investigates the potential of controlled indexing and semantic features in a more systematic way, and broader, because it explores two additional areas. First, we devise ways of applying controlled indexing in cases where no controlled vocabulary is available. Here Wikipedia serves as a source of indexing terminology and semantic relations. Second, we investigate keyphrase extraction in the context of collaborative tagging, which offers a solid but as yet unexplored basis for testing indexing techniques. Third, we substantiate the third hypothesis of the Master's thesis by creating a new algorithm that indexes as consistently as humans.

This new algorithm is called *Maui*, after the Polynesian mythological hero. Figure 1.1 explains the origins of the name. With access to domain and background knowledge, Maui performs three kinds of topic indexing—indexing with a controlled vocabulary, indexing with Wikipedia terms and automatic tagging—at a similar performance level as humans. Maui is open source and is distributed under the GNU General Public License through Google Code.[1] Additional information about Maui, its usage and application can be found in Chapter 6 and Appendices E and F.

Maui supports graphical visualization of automatically computed topics. It outputs plain-text graph descriptions using Wikipedia Miner (Milne, 2009) for identifying semantic links between topics. The GraphViz software[2] can then be used to plot the resulting graphs. Each chapter begins with a visualization of its main topics, automatically identified by Maui and shown as a semantically connected graph generated with GraphViz (see Section 6.6).

1.4 Research questions

The thesis' hypothesis can be formulated as a research question: Is it possible to perform automatic indexing at the same level as humans do? Instead of tackling this question head on, we follow the divide and conquer principle by identifying smaller research problems that together enable the understanding of the hypothesis and its solution.

1. How can the indexing performance be measured?

Topic indexing can be performed at qualitatively different levels, which means that some topics are more "correct" than others. Since indexing is a subjective task and even professionals disagree on their topics, it is necessary to abstract over the meaning of "correct" in this context. Multiply indexed document collections can be used to compare humans against each other and to determine the topics agreed on by a majority. This thesis employs such data sets where possible.

---

[1] *http://maui-indexer.googlecode.com/*
[2] *http://www.graphviz.org/*

With the increasing popularity of collaborative tagging platforms, simultaneous human judgments of important topics for the same document emerge naturally. We acquire examples of such data from the web and show how the algorithm can be evaluated against the most prolific and consistent taggers.

2. What is the performance of human indexers?

Know thine enemy! Before committing to a competition as challenging as topic indexing, it is necessary to investigate the task from the perspective of human indexers. How well do people perform indexing? How does the performance of professional librarians differ from that of amateurs indexers or taggers, who index for their own benefit? This thesis addresses this question by analyzing manually indexed collections, some created specifically for the purposes of this thesis and others obtained from a collaboratively tagged bookmarking service on the web.

3. How can a computer understand document concepts?

Given the document terminology and that of a prescribed indexing vocabulary, we need to identify phrases that mean the same thing in order to bridge the gap between the author's language and that of his colleagues, as well as current and future readers. Ambiguity and synonymy in human language are the main obstacles. Where one speaker is more accustomed to *peppers*, another prefers to call them *capsicums*; and while both might mention *apple*, one refers to the fruit and the other to the computer company.

While analyzing a text, phrases need to be linked to corresponding concepts. These can originate from controlled vocabularies—manually created domain-specific thesauri. Such vocabularies are useful, but they are expensive to create and maintain and not always available. A solution to this problem is investigated in the next question.

4. Is controlled indexing possible in the absence of a vocabulary?

Controlled vocabularies not only help combat the polysemy and synonymy of human language, but also make topic indexing consistent across the document collection. They are used for browsing the collection and "getting the feel" for its content. In automatic indexing, controlled vocabularies serve as sources of semantic knowledge that improves performance (Section 1.2). However, creating and

maintaining such vocabularies is problematic, particularly for swiftly changing domains such as politics, business, current affairs, entertainment, and new technologies. Unfortunately, these are the most broadly covered domains on the web.

Recently, Wikipedia has been discovered as the largest available source of topics in many languages (Milne *et al.*, 2006). It creates an opportunity for controlled indexing in situations where manually constructed vocabularies are not available. This thesis investigates how to utilize Wikipedia as a vocabulary and how to efficiently map document terms onto Wikipedia articles—a non-trivial task given the millions of articles. Furthermore, we also mine Wikipedia for semantic features that improve automatic topic indexing.

5. How can main *concepts be identified automatically?*

Once the concepts that are mentioned in a text are known, we need to determine the most significant ones for that text. Professional indexers follow instructions about specificity and exhaustiveness of the topics, but how they make the final decisions can only be guessed. Section 1.2 noted that human indexers fall back upon their knowledge of the domain and of the controlled vocabulary (if one is used). They also reason about the meaning of a document using linguistic knowledge about language concepts. These processes can be simulated by statistical analysis of document collections and by employing manually encoded knowledge bases and linguistic resources such as Wikipedia. Each meaningful property of a manually assigned topic can be taken as a cue by the machine learning algorithm. The goal is to identify such properties, or features, and compute their individual contribution, as well as the impact of using them combined.

6. How much training is necessary?

Experiments in this thesis involve supervised indexing, where the algorithm *learns from examples* how to combine the features for classifying candidate terms into topics and non-topics. This has the advantage over symbolic approaches that the features are weighted according to data set characteristics instead of manual tweaking. The disadvantage is the dependence on the training data. Since generating manually indexed documents is expensive, it is instructive to investigate how much data is required to produce good results. By gradually adding more manually

indexed documents into the training set and re-assessing the algorithm's performance, the effect of increasing the training size can be evaluated.

7. Can an indexing algorithm be domain and language independent?

The Master's thesis preceding this PhD tested the automatic indexing technique on agricultural documents written in English, but suggested that the proposed approach is domain and language independent. This thesis investigates this assumption experimentally by testing the algorithm on various data sets, including computer science technical reports, physics, medical and scientific publications, as well as collections in languages other than English.

1.5 Contributions

The thesis makes the following research contributions:

1. Survey of the task

An array of highly related research problems emerging in different fields has resulted in a confusion of terminology. Chapter 2 surveys the nature of tasks related to topic indexing and brings them together into a single picture, while Chapter 3 reviews existing methods of topic indexing and identifies their potential drawbacks. The thesis also offers a glossary of terms related to topic indexing in Appendix A, and a collection of relevant web resources in Appendix G.

2. New evaluation method

The thesis proposes evaluating automatic indexing systems using traditional techniques for measuring the performance of professional human indexers. Instead of a clear-cut decision as to whether a topic is correct or incorrect based on the judgment of a single person, our gold standard is the average inter-indexer consistency of a group of humans on the same documents (Section 2.2). This thesis also shows how to utilize collaboratively tagged data for evaluating automatic topic indexing (Section 4.3).

3. New techniques

New features for automatic topic indexing are proposed and evaluated. These are derived statistically from the corpus and from external resources like Wikipedia

(Section 5.2). Additionally, a new method is proposed for performing controlled indexing in the absence of a controlled vocabulary, using Wikipedia article titles (Section 5.1).

4. New tools

Two new pieces of open-source software were produced for this thesis:

- **Kea-5.0** – A new version of the keyphrase extraction algorithm Kea (Witten *et al.*, 1999; Frank *et al.*, 1999) that can be used for two tasks: keyphrase extraction and term assignment with any controlled vocabulary in a predefined format. It is an extension of Kea++, the algorithm in my Masters thesis, which was restricted to term assignment for agricultural documents only.
- **Maui** – A multi-purpose topic indexing algorithm that offers all the functionality of Kea and also permits the use of Wikipedia as a controlled vocabulary. Maui implements new features and a new classifier, which significantly improve the performance of topic indexing tasks compared to Kea. The Maui algorithm is described in Chapter 6, whereas Appendices E and F list details of its usage.

5. New data

Two multiply indexed collections were generated for this thesis:

- In a user experiment, 15 teams consisting of two students each assigned topics from Wikipedia to 20 computer science articles. The teams were competing against each other by trying to match topics picked by other teams, ensuring high quality of the assigned topics.
- Given collaboratively tagged data on the bookmarking web site *CiteULike.org*, we automatically extracted corpus of 180 science research papers each tagged by at least three users. This sample is, to our knowledge, the first available multiply indexed data set of this size created in a natural setting.

6. Publications

This PhD research resulted in articles in the *Journal of American Society for Information Science and Technology* and the *International Journal of Human-Computer Studies*, and nine articles published in proceedings of peer-reviewed national and international computer science research conferences and workshops. A full list of publications, including succinct summaries, appears in Appendix B.

1.6 Thesis outline

Chapter 2 provides background knowledge about topic indexing. Section 2.1 surveys different topic indexing tasks and groups them based on commonalities. It also specifies the three tasks addressed in this thesis: term assignment, keyphrase extraction and tagging. Section 2.2 surveys methods used to evaluate the quality of topics assigned by humans and algorithms.

Chapter 3 categorizes and reviews existing approaches to automatic term assignment, keyphrase extraction and tagging. Section 3.4 identifies gaps at intersections of these fields and discusses how these gaps are addressed in this thesis.

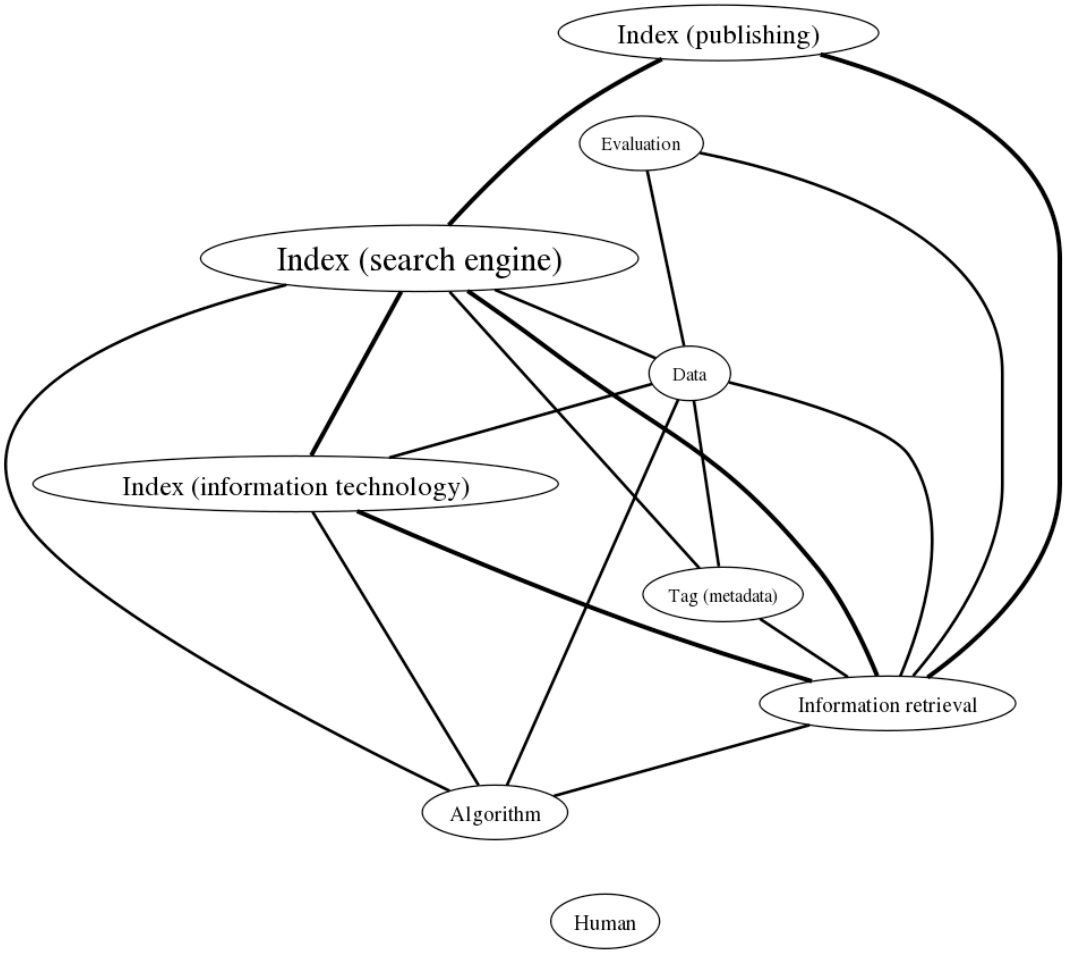
Chapter 4 provides statistical analysis of experimental data sets, including an in-depth analysis of topics assigned by people. Section 4.1 compares vocabularies used in traditional term assignment. Section 4.2 answers research question 4 (Section 1.4), namely how term assignment is possible when vocabularies are unavailable. Section 4.3 demonstrates how a high-quality multiply indexed collection can be automatically extracted from a collaboratively tagged bookmarking site. This chapter also tackles research question 2—how well humans perform topic indexing—and specifies the gold standard for the algorithm's performance.

Chapter 5 explains the two stages of the automatic topic indexing approach: candidate generation and filtering. Candidate topics are concepts discussed in a document. Section 5.1 presents the candidate generation algorithm and addresses research question 3—how computer can "understand" document concepts. Section 5.2 surveys typical properties of candidate topics used to determine the most prominent ones. These properties reflect statistical, semantic, domain and back-

ground knowledge required for the algorithm to succeed in the topic indexing task.

Chapter 6 brings the individual indexing steps together into the single algorithm Maui. It explains how Maui selects the main topics in a document by retrieving candidate topics, computing their properties and analyzing examples of indexing performed by humans.

Chapter 7 evaluates Maui on data sets described in Chapter 4. It answers and discusses the main hypothesis of this thesis: whether computers will index as well as humans if sufficient domain and general semantic knowledge is provided.


Index (publishing)
Evaluation
Index (search engine)
Data
Index (information technology)
Tag (metadata)
Information retrieval
Algorithm
Human

Chapter 2
Scope of the thesis

Topic indexing has received much attention in all areas where large document collections are created, collected, stored and regularly accessed by users. Traditionally, libraries have been employing professional indexers to identify the main topics of the documents, as well as to record other metadata such as author name and publication type. However, libraries are no longer the only institutions where a thematic overview of the holdings is needed. Nowadays, companies, organizations and even individuals own extensive collections. It is no longer cost-effective to employ professionals for the topic indexing task. Instead, collection owners rely on search technologies to efficiently access and manage their data. A variety of algorithms have been developed to automate or semi-automate topic indexing. Recently, on the web, users have been encouraged to create topical metadata themselves.

This chapter gives an overview of manually and automatically performed tasks related to topic indexing. Section 2.1 organizes the tasks based on criteria like the source of terminology and the number of identified topics. The tasks that share prominent characteristics define the scope of the thesis. Section 2.2 surveys existing methods for assessing the quality of topics produced by human indexers and by algorithms. This section also discusses similarities, advantages and disadvantages of these methods and chooses those that build the scope of the evaluation methodology for this thesis.

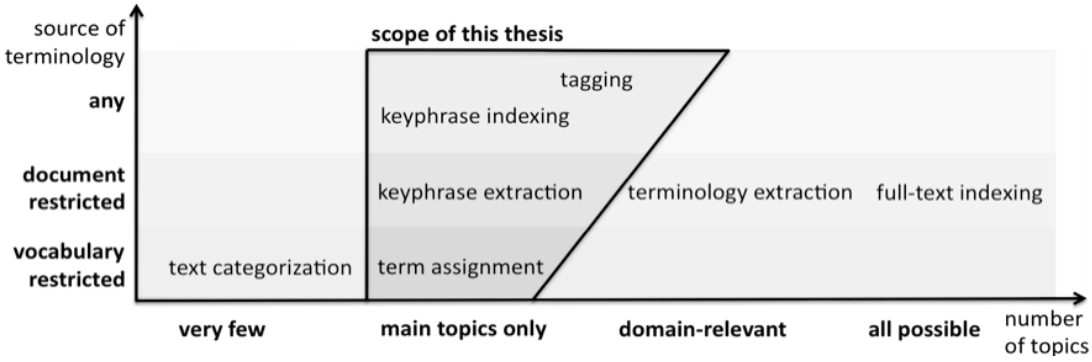


Figure 2.1 Tasks related to topic indexing

2.1 Types of topic indexing

Tasks related to topic indexing can be broadly organized according to two criteria:

- source of the terminology used to refer to a document's concepts;
- number of topics assigned per document.

Figure 2.1 plots the relevant tasks using these criteria as dimensions. The "number of topics" axis ranges from a few topics to as many as possible, where the number is implicit in the task's definition. The "sources of terminology" axis lists possible sources of topics such as a vocabulary, the document itself or any possible source.

The diagram places closely related tasks next to each other, because they share similar characteristics. For example, *keyphrase extraction* is similar to *terminology extraction* but extracts fewer topics per document. Table 2.1 briefly describes each task in the Figure. Four tasks are grouped in the trapezoidal area representing the scope of the thesis. In the following, these tasks are discussed in more detail.

2.1.1 Term assignment

Term assignment expresses the main topics in a document using terms from a pre-defined controlled vocabulary, e.g. a domain-specific thesaurus. These terms do not necessarily appear within the document. *Subject indexing* is a more common term for the same task in the context of library science.

A controlled vocabulary lists concepts relevant to a given domain using two sets of terms: descriptors and non-descriptors. *Descriptors* are the preferred terms for referring to the concepts. *Non-descriptors*, also called *entry terms*, are usually

Task name	Also known as	Description
text categorization	text classification	Very few general categories, like *Politics* or *News*, are assigned from usually a small vocabulary
term assignment	subject indexing	Main topics are expressed using terms from a large vocabulary, e.g. a thesaurus
keyphrase extraction	keyword extraction, key term extraction	Main topics are expressed using the most prominent words and phrases in a document
terminology extraction	back-of-the-book indexing	All domain relevant words and phrases are extracted from a document
full-text indexing	full indexing, free-text indexing	All words and phrases, sometimes excluding stopwords, are extracted from a document
keyphrase indexing	keyphrase assignment	A general term, which refers to both term assignment and keyphrase extraction
tagging	collaborative tagging, social tagging, auto-tagging, automatic tagging	The user defines as many topics as desired. Any word or phrase can serve as a tag. Applies mainly to collaborative websites

Table 2.1 Tasks related to topic indexing

synonyms for the corresponding descriptors. Using descriptors in indexing improves consistency. Non-descriptors are helpful when searching for a concept (Wright and Budin, 2001).

An example of term assignment can be found in PubMed, an online database maintained by the U.S. National Library of Medicine, which provides access to millions of citations via the Medical Subject Headings (2005) or MeSH.[1] The MeSH vocabulary contains over 25,000 descriptors and a further 160,000 non-descriptors (called entry terms) associated with them. Figure 2.2 shows an excerpt from the MeSH hierarchy focusing on the descriptor **Analgesics**. The broader terms and the Scope Note tell us that *Analgesics* are chemical agents that affect the nervous system and are used to relieve pain. Below the Scope Note, four Entry Terms are associated with this descriptor (e.g. *Analgesic Agents*).

Returning to Figure 2.1, *text categorization* appears to the left of term assignment. The controlled vocabulary used in text categorization contains general categories rather than specific concepts and is usually relatively small (from around

[1] *http://www.nlm.nih.gov/mesh/*

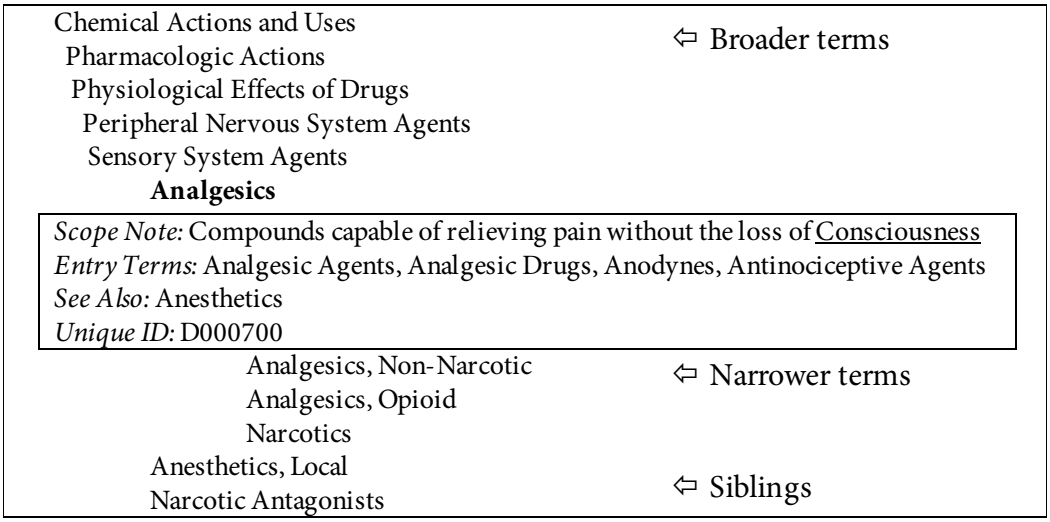


Figure 2.2 Entry for *Analgesics* in the MeSH vocabulary

ten up to a few hundreds categories). For example, Reuters news—a common data set in text categorization—contains just over 100 categories (Dumais *et al.*, 1998). *Categories* are used to organize documents thematically into broad areas like *Politics* or *Entertainment*, whereas subject headings describe specific topics like *U.S. Presidential Elections* or *Academy Awards*.

Most documents belong to one general category, just as newspaper articles are organized into distinct sections like *Politics*, *Fashion*, *Sport*, *Ads*. Articles that belong to two or more categories are unusual. A category can be assigned based not only on the document's content, but also on its style. For example, articles in the category *Ads* are characterized by short sentences, abbreviations and numbers. Automatic text categorization lies outside of the scope of this thesis because it requires a different approach than automatic term assignment. However, some text categorization methods have been employed for term assignment, as discussed in Section 3.1.1.

2.1.2 Keyphrase extraction

Moving upwards in the scope of the thesis area, Figure 2.1 lists *keyphrase extraction* and *keyphrase indexing*. The general task here is to define document topics. However, unlike term assignment, a controlled vocabulary is optional. *Keyphrase extraction* usually refers to a task performed by an algorithm that selects prominent phrases appearing in a document. Keyphrase indexing refers to a more gen-

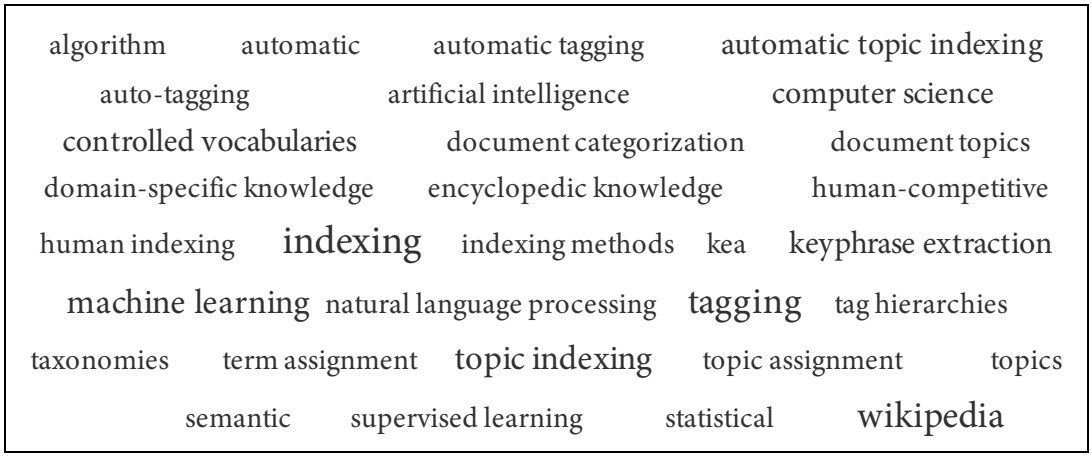


Figure 2.3 Tag cloud of topics assigned to this thesis by its proofreaders

eral task than keyphrase extraction and term assignment, where the source of terminology is not restricted. For example, academic publishers encourage their authors to assign freely selected keyphrases. These phrases may or may not appear in the text.

Two tasks that are similar to keyphrase extraction are *terminology extraction* (further right in Figure 2.1), where every domain-relevant phrase needs to be extracted from a document, and *full-text indexing* (far right), where the entire vocabulary of the document is transformed into an *index*. These tasks lie outside the scope of the thesis. However, terminology extraction can be addressed with keyphrase extraction methods, as discussed in Section 3.2.

2.1.3 Tagging

Finally, Figure 2.1 places *tagging* approaches between the "main topics only" and "domain-relevant" columns. Like keyphrases, *tags* can be chosen freely. There are no formal guidelines. Users decide which terms and how many terms should be assigned and perform the assignment principally for their own benefit.

Tagging is encouraged on websites that host user-generated content, such as blogging platforms, online bookmarking services and file sharing sites. Often several users tag the same object and their tags are merged into a single set. The entire set of tags assigned by all users of a given website is called a *folksonomy*. The process of producing a folksonomy is known variously as collaborative tagging, social tagging and social indexing.

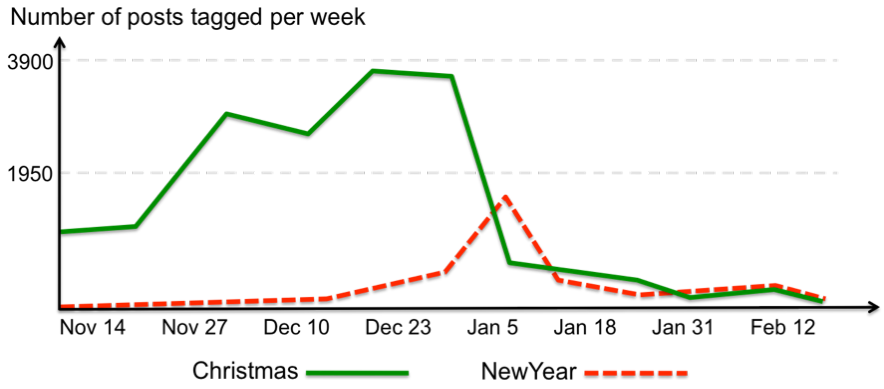


Figure 2.4 Popularity of tags *NewYear* and *Christmas* in blogs over time

Some services represent folksonomies as *tag clouds* like the one in Figure 2.3, depicting tags assigned independently to this thesis by its proofreaders. The tags are ordered alphabetically and the font size reflects the relative number of taggers who agreed on the tag. Note how similar topics were expressed with different tags: *automatic tagging* and *tagging*, *topic assignment* and *topic indexing*.

Web services like *technorati.com*, *del.icio.us* and *flickr.com* use tags to organize and provide access to user-supplied data. *Technorati.com* also uses tags to identify discussion trends and their change over time just as Google Zeitgeist[2] generates trends from search queries. Figure 2.4 depicts the popularity of *Christmas* and *NewYear* tags from November 2008 to February 2009.

Recently, researchers began developing methods to assign tags automatically (Section 3.3). Such methods can provide suggestions that can facilitate tagging and encourage more users to supply tags. *Automatic tagging*, or *autotagging*, derives tags from varying sources, including the terminology of the document itself, other documents owned by the users (e.g. located on their desktop), and tags previously assigned to similar documents.

2.1.4 Index terms, keyphrases and tags

Topic indexing tasks discussed in this section share the same goal: producing content-based metadata for document collections. This metadata is named differently depending on the task. In term assignment topics are *subject headings*, *index*

[2] *http://www.google.com/zeitgeist*

terms and *descriptors*; in keyphrase extraction they are *keywords* and *keyphrases*; in tagging they are simply *tags*. In fact, these are all essentially the same thing and therefore in this thesis are often referred with the generic term *topics*.

There are slight differences between different kinds of topics. Subject headings originate from a controlled vocabulary, whereas keyphrases and tags are unrestricted. Subject headings and keyphrases tend to be domain-specific, whereas tags reflect everyday language and commonly used expressions. For example, as of February 2009, *technorati.com* shows over 400 posts tagged with *painkillers*, and less than 50 tagged with *analgesics*. Both WordNet and Wikipedia list these terms as synonyms, but the MeSH vocabulary only contains *analgesics* and does not mention *painkillers*. This example indicates that MeSH terms are targeted at professionals.

Three types of topic indexing are addressed in this thesis: term assignment, keyphrase extraction and tagging. The Maui algorithm applies the same general strategy for solving these tasks and only requires minor adjustments. For example, in term assignment, Maui accesses a controlled vocabulary, whereas in tagging it does not. Maui uses machine learning in order to capture the indexing behavior of humans in each task, e.g. the specificity of indexing or word preferences. Its indexing performance is then compared directly to the performance of human indexers using evaluation techniques discussed below.

2.2 Evaluation methods

The tasks related to topic indexing have developed somewhat independently in disciplines such as library and information science, computer science and on the web. As a result, each discipline has created its own methods for measuring indexing quality. Simple methods just compare the generated topics for an exact match. Complex ones pay attention to details such as the varying degree of correctness or the semantic similarity of topics. This section discusses existing measures, their origins and applications. It also reveals interesting relations and missing links between different them.

2.2.1 Inter-indexer consistency

Two sets of topics can be compared by measuring the number of matching topics relative to the sizes of the two sets. When topics sets are assigned by people, the result reflects the *inter-indexer consistency*, also called *inter-indexer agreement* or simply *indexing consistency*, between these people. Inter-indexer consistency is defined as "the degree of agreement in the representation of the (essential) information content of a document by certain sets of indexing terms selected individually and independently by each of the indexers" (Zunde and Dexter, 1969). Hooper (1965) quantifies this metric as

$$Hooper = \frac{C}{C + M + N}$$

where C is the number of terms two indexers have in common, and M and N respectively are the number of *idiosyncratic* terms that they assign.

Another measure was proposed by Rolling (1981):

$$Rolling = \frac{2C}{A + B}$$

where again C is the number of terms the indexers have in common and A and B are the *total* number of terms they assign.

Both measures range from 0 when the two indexers assign disjoint sets to 1 when they assign identical sets. The result can be multiplied by 100 to express the percentage, as done in this thesis. The choice of the Hooper or the Rolling measure for a particular study seems to depend on the personal preference of the researcher.

To determine the overall indexing quality, consistency values are averaged across all documents, and across all co-indexers. In traditional catalogues, professional indexers do not index the same documents, but rather work on their own share of the catalogue. Multiply indexed data sets are either created specifically for evaluation studies, or through accidentally generated duplicates.

Studies of indexing consistency show that the greater the indexing consistency is in a catalogue, the more reliable the indexing is and the more efficient the search is in this catalogue (Leonard, 1975; Saracevic and Kantor, 1988; Iivonen 1995).

2.2.2 Precision and recall

Surprisingly, inter-indexer consistency is rarely mentioned in the context of automatic indexing. Here, the standard information retrieval measures *precision* and *recall* are popular (van Rjsbergen, 1979). Given two topics sets, one assigned by an algorithm and another by a human, the human's set is considered to be "correct". This set serves as the gold standard against which algorithm's topics are compared.

Precision (P) expresses the number of matching ("correct") topics as a proportion of all algorithm's topics and *recall* (R) is the proportion of human's topics that are covered:

$$P = \frac{\#\,correct\ extracted\ topics}{\#\,all\ extracted\ topics} \qquad R = \frac{\#\,correct\ extracted\ topics}{\#\,manually\ assigned\ topics}$$

The *F-measure* (F_β) combines the two, and in its full generality involves a parameter β, which allows the evaluator to give more weight to either P or R (van Rijsbergen, 1979):

$$F_\beta = \frac{\left(1+\beta^2\right)PR}{\beta^2 P + R}$$

In this thesis, F_1 is reported ($\beta = 1$), which makes precision and recall equally important.

An algorithm's topic sets are usually matched against manually assigned sets. Before applying the measures, the freely chosen keyphrases and tags can be first reduced to their base form using a stemmer (e.g. Porter, 1980). Achieving a 100% exact matching, i.e. P=1 and R=1, is practically impossible, because even humans disagree on the "correct" topics for a document. Non-matching terms may still be correct, yet are ignored by automatic evaluation.

A more flexible way of estimating the number of correct topics is to use human judgments (e.g. Pouliquen *et al.,* 2003). In fact, in some studies humans rate not only automatically but also *manually* assigned topics (Mishne, 2006; Sood *et al.*, 2007). As long as the evaluation involves several human raters and their *inter-rater agreement* is reported, such evaluation methods are acceptable. However, if the study relies on the judgments of a single human, the evaluation results are just as

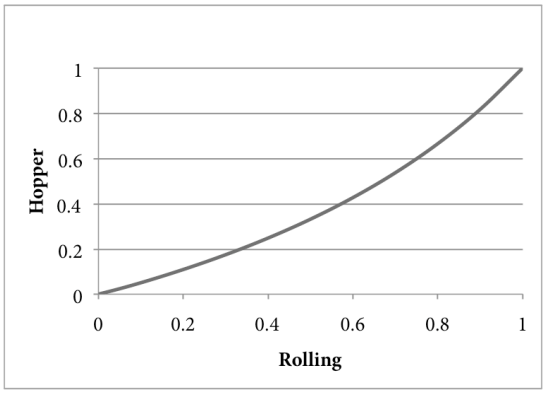


Figure 2.5 Relation between the Rolling and Hooper consistency measures

subjective and unreliable as those using automatic matching against singly assigned keyphrase sets.

2.2.3 Relations between the measures

The above measures compare the number of matching items in two sets, denoted as *C*, against the total number of items in these sets, *A* and *B* respectively. Expressing all measures in these terms reveals that they are closely related.

Reformulating the Hooper measure shows that it is always smaller than Rolling, except for the two extremes 0 and 1, since

$$Hooper = \frac{C}{A + B - C} = \frac{Rolling}{2 - Rolling}$$

Figure 2.5 plots this relationship. Before comparing inter-indexer consistency studies, their results need to be converted to the same measure.

Furthermore, the Rolling and Hooper consistency measures are the same as the Jaccard and Dice coefficients used to measure statistical similarity between sets *A* and *B* (Manning and Schütze, 2002):

$$Jaccard = \frac{A \cap B}{A \cup B} = Hooper \qquad Dice = \frac{2|A \cap B|}{|A| + |B|} = Rolling$$

Expressing precision and recall in the same terms as the inter-indexer consistency measures shows that the F-measure coincides with the Rolling measure:

$$F_1 = \frac{2PR}{P + R} = \frac{2C}{A + B} = Rolling$$

The Kappa statistic (Cohen, 1960) computes indexing consistency while taking into account the proportion of times the indexers would agree by chance. It addi-

tionally uses the "negative" counts—that is, the number of possible topic choices that the indexers did *not* make. When no vocabulary is used in topic indexing, or the vocabulary is large, Kappa is the same as the F-measure (Hripcsak and Rothschild, 2005).

For example, given two keyphrase sets {*complex systems, network, small world*} and {*theoretical, small world, networks, dynamics*}, the intersection, or the set of matching terms (after stemming) is {*network, small world*}. This gives the Rolling consistency of 2×2/(3+4) = 0.57. To compute precision and recall, one of the sets needs to be seen as the gold standard. If the first set was assigned automatically, the precision is 2/3 = 0.66 and recall is 2/4 = 0.5. The F-measure is 2×0.66×0.5/(0.66+0.5) = 0.57, the same as Rolling.

2.2.4 Alternative evaluation methods

Some researchers argue that simple measures are not sufficient to assess indexing quality. Zunde and Dexter (1969) show that the degree of topics' "correctness" varies. Given a document and four indexers, assume that they all agree on topic *X*, but only one assigns topic *Y*. It follows that topic *X* is four times less significant than topic *Y*. Zunde and Dexter propose a consistency measure based on fuzzy sets, which take into account the relative frequency of topics across all assignments.

However, the original inter-indexer consistency measure also considers the relative significance of topics, indirectly and as long as three or more indexers are involved. If a human indexer, or an algorithm, has chosen a topic agreed on by many co-indexers, its overall consistency naturally increases. Fuzzy sets seem unnecessary.

Soergel (1994) considers indexing consistency by itself as problematic, because indexing can be consistently incorrect. He suggests to measure the correctness of indexing using two additional metrics: completeness and purity. *Completeness* computes the proportion of the correctly assigned terms out of all possible correct ones and is directly related to recall. *Purity* is the number of correctly rejected index terms out of all those that should have been rejected. Soergel states that cor-

rectness should be defined by the cataloguing rules. However, he agrees that measuring it is difficult and laborious.

Outside the library context, topic indexing rules are rarely specified. Academic authors are never given instructions when they are asked to provide keyphrases for their articles. In tagging, a user may add any tag as long as it *makes sense* (Golder and Huberman, 2006). Without strict rules and guidelines that eliminate subjectivity from the indexing, assessing the purity and completeness is hardly possible.

Many studies of indexing quality have noted the distinction between consistency at the terminological and conceptual levels (e.g. Markey, 1984; David *et al.*, 1995; Iivonen, 1995; Saarti, 2002). Measures discussed in Section 2.2.1 evaluate *terminological consistency*, where topics must match exactly. *Conceptual consistency* allows matches that are semantically related such as abbreviations (*NZ* and *New Zealand*), synonyms (*painkillers* and *analgesic*), or terms of varying specificity (*iPod* and *iPod Touch*).

Markey (1984) and Iivonen (1995) count such cases as "correct" and then compute consistency as usual. In both studies, human evaluators assessed semantic relatedness manually. The authors report that subjects' conceptual consistency is higher than terminological one, but do not discuss the relative significance of terminological and conceptual matches, or their effect on retrieval effectiveness.

Current search interfaces rarely provide search based on concepts. If two documents are indexed with conceptually related but different terms, they cannot be found using just one search query. Although conceptual consistency does reflect the actual agreement of indexers on what the document is about, this measure cannot be directly associated with high retrieval effectiveness in a way the terminological consistency does.

Semantically enhanced evaluation can be useful when comparing the performance of algorithms. Suppose two algorithms have produced topic sets for a particular document. Neither of the two sets match the manually assigned topics exactly. The first algorithm's topics might be completely wrong, whereas the second algorithm's ones might be semantically similar to those assigned by a human. Current

precision and recall analysis would not be able to identify the second algorithm as the better one.

Barrière and Jarmasz (2004) suggest computing the average semantic similarity between automatically and manually assigned topic sets. The similarity between each pair of topics is determined using the Pointwise Mutual Information measure that compares their co-occurrence statistics in a large corpus.

Medelyan and Witten (2006) define an indexing consistency measure based on vector similarity of topic sets. The vector elements for matching terms are set to 1, and 0 for non-matching. For those terms that do not match but are related to a term in the other topic set, the vector element is set to a value between 0 and 1, reflecting the strength of the semantic relation. The relatedness is determined using links in the controlled vocabulary.

These two studies show that with access to large corpora and controlled vocabularies, semantic similarity does not need to be assessed manually, as done in the studies by Markey and Iivonen. However neither of the methods have become generally accepted by other researchers.

2.2.5 Evaluation methods used in this thesis

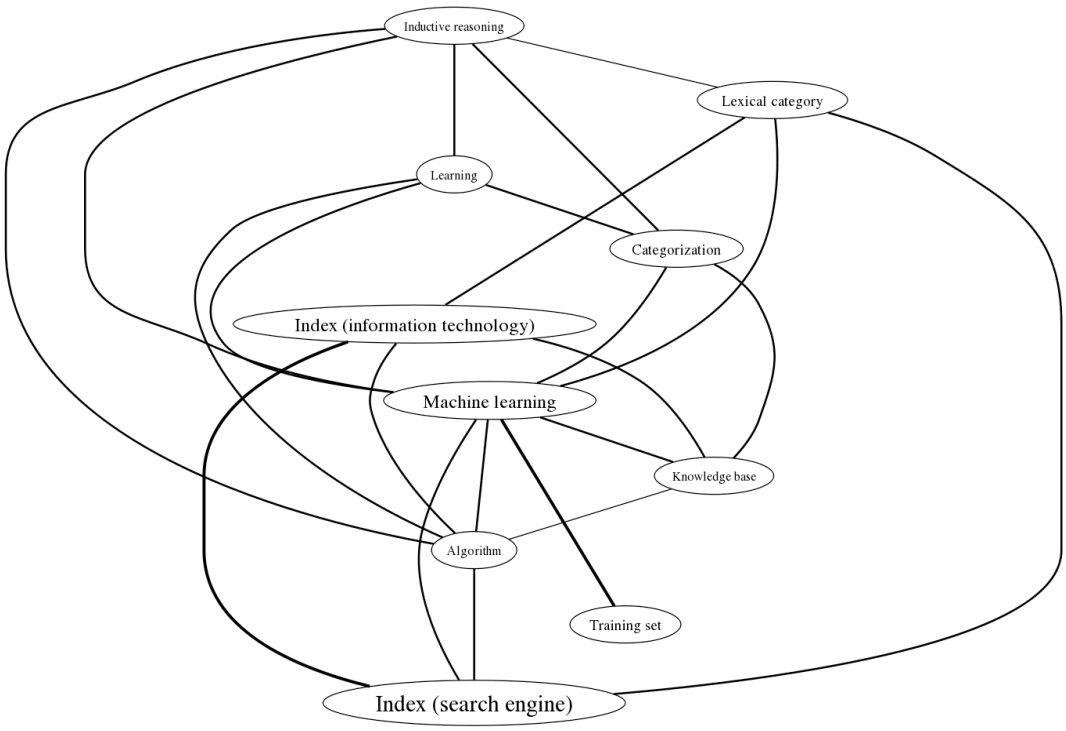
The methods presented in this section differ in their complexity and application areas. Whereas indexing performance of people is assessed in large groups, algorithms are evaluated by matching their topics to those assigned by just one person. On the one hand, creating multiply indexed collections, particularly on a large scale, is costly. On the other hand, algorithm developers rarely mention that such collections are needed. There is not much cross-referencing between the studies of human and automatic topic indexing. Why else would there exist two identical formulas with different names like Rolling and F-measure?

This thesis addresses the missing link between evaluation of human and automatic indexing in three ways. First, each topic indexing task (term assignment, keyphrase indexing and tagging) is evaluated using carefully created and analyzed multiply indexed collections described in Chapter 4. Although some of these collections are small, the evaluation is meaningful because it reflects the difficulty of

indexing from the human perspective. Second, a new multiply indexed collection is created from scratch using assignments by 15 teams of students. The analysis of their performance reveals interesting factors affecting indexing consistency. Third, we propose a new method for extracting a large and accurate corpus of multiply indexed documents from a collaboratively tagged corpus. These two new collections with several topic sets per document have been made available to other researchers.[3]

To provide an easy comparison to existing algorithms, the standard measures of precision and recall are computed using large data sets and the results are given alongside those produced using multiply indexed collections. However, it is the inter-indexing consistency analysis that provides a direct comparison of the algorithm to humans and shows whether the research hypothesis of this thesis can be proven.

[3] *http://code.google.com/p/maui-indexer/wiki/MultiplyIndexedData*


Inductive reasoning
Lexical category
Learning
Categorization
Index (information technology)
Machine learning
Knowledge base
Algorithm
Training set
Index (search engine)

Chapter 3
Related work

This thesis investigates topic indexing—discovering the main topics in a document. In Chapter 2, we noted that tasks sharing this activity differ in how the topics are selected. Indexing strategies vary radically depending on whether the topics originate from a controlled vocabulary, document text or terms assigned to similar documents. This chapter organizes and surveys the large number of published methods for automatic topic indexing in three major groups:

- *Term assignment* methods, which use a controlled vocabulary (Section 3.1);
- *Keyphrase extraction* methods, which derive topics from document text (Section 3.2);
- *Tagging* methods, which mine topics from any possible source (Section 3.3).

Each group shares a similar general strategy, but the individual realizations differ significantly. Nevertheless, techniques commonly used in one group are often adopted in other groups. We identify the strengths and weaknesses of each method and, in Section 3.4, discuss how current limitations in topic indexing can be addressed by taking the best from existing approaches.

3.1 Assigning terms from a vocabulary

Term assignment, also called subject indexing, has been addressed in two ways. The first is similar to text categorization, where document properties such as words and phrases are analyzed and the document is classified into terms listed in the vocabulary using manually generated or automatically induced rules. In this context, terms are also called "categories". The second approach generates candidate topics

by mapping document phrases onto vocabulary terms and then analyzes their characteristics to determine the most significant ones.

3.1.1 Classification and rule building

Early methods recruited knowledge-engineering experts who created classification rules manually. Fuhr and Knorz (1984) describe a decision-making system that uses approximately 150,000 such rules to map physics documents to vocabulary terms. The rules have the form

IF <property> is identified in the document THEN <descriptor>,

where <property> is a word, phrase, or a physics formula that appears in the text.

During the 1990s, the focus shifted towards machine learning techniques (Sebastiani, 2002). Various inductive learning schemes have been applied to analyze the characteristics of manually classified documents and build classification rules automatically. Each rule is weighted with a confidence value ranging from 0 to 1. The algorithm uses a threshold to decide whether a particular document should be assigned to a given class or not. A classifier for the category *interest* might be:

if (*interest* AND *rate*) OR (*quarterly*), then confidence(*interest*) = 0.9.

In vector-based classification, training documents are first mapped into a vector space. Each word appearing in the collection serves as a dimension and documents are represented as vectors, where each element indicates the presence or absence of a particular word in a document, or a weight representing the importance of this word. A classifier analyzes the similarity of a new document to those manually assigned to vocabulary terms and determines its topics.

If all words in a collection are used, the resulting vector space is high-dimensional and requires substantial computational resources. The number of dimensions can be reduced by applying stemming, ignoring stopwords, and retaining only those words that are most useful for the classification, defined by a specified function (Sebastiani, 2002).

The performance of such approaches depends on the classifier. Dumais *et al.* (1998) compares the text categorization performance of classifiers such as *Find*

Similar (vector-based), *Decision Trees*, *Naïve Bayes*, *Bayes Nets* and *Support Vector Machines* (SVMs). The SVM method achieves the best results, with an average accuracy of 87% for classifying news stories into 118 Reuters categories. In a similar study, Sebastiani (2002) successfully applies *classifier committees*, where the decision is made by an ensemble of classifiers that can be combined in different ways.

In an experiment reported by Pouliquen *et al.* (2003), a much larger indexing vocabulary is used, which stretches the capacity of the vector space representation. Documents are automatically classified under 6075 descriptors from the Eurovoc thesaurus covering fields such as politics, law, finance, and sociology. To create a classifier for each descriptor in Eurovoc, a very large training set of 60,000 documents is employed. Each word is weighted using the TF×IDF score (Section 5.2.1), and only commonly occurring words are included. Given a new document, a vector is produced in the same way and compared to the descriptor vectors using several similarity metrics. A human evaluator assessed topics automatically assigned to 160 documents, judging a topic as correct if it described the document well or was semantically related to a possible descriptor. The reported precision for the 8 top terms was 67%. To compute recall, human evaluators assigned 8 terms per document, which were then compared to automatically generated ones. Recall of 63% was reported. It is unclear whether the recall evaluation was performed after automatically assigned terms were assessed, which could have affected indexers' choices. Pouliquen *et al.* report that on Spanish texts even better results were achieved.

There are two main issues with vector-based classification. First, a classifier needs to be created for each vocabulary term. Pouliquen *et al.* address this by using a large training corpus. Second, the multi-dimensionality of the resulting vector space is computationally expensive. Pouliquen *et al.* restrict the analysis to the most common words.

Plaunt and Norgard (1998) describe a different solution to these problems. Instead of building a vector space, they generate association rules using a contingency table, which records how often a document phrase co-occurs with a vocabulary term manually assigned to this document. Only phrases appearing in the

document title and abstract are considered. Each rule maps a phrase to a vocabulary term with a confidence value computed using the likelihood ratio statistic over the co-occurrence values. The scheme is evaluated on a subset of the INSPEC database—documents that contain the words *libraries*, *library*, *information science*, *linguistics*, or *sigir* in their titles. From this highly focused collection of 4,100 training documents, 27,500 rules were extracted that covered 1,100 of the 6,500 descriptors in the INSPEC thesaurus. Evaluation was performed automatically by matching the algorithm's top 10 descriptors to manually assigned ones, which resulted in precision and recall of 21% and 64% respectively. No error analysis is given, but the authors note a very low coverage of vocabulary descriptors (17%).

Aronson *et al.* (2000) and Markó *et al.* (2004) automatically assign terms from the Medical Subject Headings vocabulary (Section 2.1). Like Plaunt and Norgard, they apply rules based on conditional probabilities of vocabulary terms to co-occur with document phrases in a training corpus. To achieve high vocabulary coverage, a large corpus is used and document terms are pre-processed before generating the association rules. Aronson *et al.* split words into tri- or bi-grams of characters, whereas Markó *et al.* decompose them into subwords. The latter approach is particularly interesting. It is based on a manually created dictionary, MorphoSaurus,[1] that ensures orthographic, morphologic and semantic normalization of document terms to a set of meaningful identifiers. Normalization is particularly useful in terminologically rich domains such as medicine, and compositional languages such as German. The probability of a vocabulary term being a topic is the product of the conditional probabilities of its subword trigrams in documents to which the term was manually assigned, divided by the probability of the trigrams over all training documents. After training on 35,000 abstracts, Markó *et al.* assign MeSH terms to unseen documents with precision and recall of around 30% for the top 10 terms. Aronson *et al.* report similar results, as described in Section 3.1.2.

---

[1] MorphoSaurus is maintained manually and comprises over 20,000 equivalent classes for English, German, Spanish and Portuguese subwords. Subwords are semantic units and morphemes that link to equivalent expressions—for example, *leukemia* becomes *leuk, em, iag*.

The particular advantage of Markó *et al.*'s approach is the language independence of the classifiers. A classifier created from documents in one language can be used to index collections in any other language encoded in MorphoSaurus. Comparable results are reported for indexing German and Portuguese documents after training on English ones. Unfortunately, creating knowledge bases like MorphoSaurus requires diligent efforts by experts in both linguistics and the given domain (e.g., medicine).

The main advantage of classification-based techniques is that even terms that do not appear verbatim in the document can be assigned as its topics. A classifier is a generalized model of how a topic is represented using language elements like phrases, words, morphemes or frequent letter combinations. Classifiers created using machine learning perform extremely accurately on small vocabularies (Dumais *et al.* 1998, Sebastiani 2002), or when a sufficient training data is available (Pouliquen *et al.* 2003). However, this is where the disadvantages of classification lie. Whether rules are constructed manually or automatically, training data is required for *every* term in the vocabulary. Only terms that appear in the training data can be assigned to new documents, and, for an adequate model, several documents per term are required.

Existing approaches rely on thousands of manually pre-indexed documents. For example, Pouliquen *et al.* used a training set with 60,000 documents for a vocabulary with 6,000 terms. Most domain-specific thesauri are far larger. Markó *et al.*'s technique appears to be more parsimonious: they use 35,000 abstracts to create a training model for a subset of the 20,000-item MeSH vocabulary. However, in both cases it is not reported how many vocabulary terms were actually covered. Training data of this size and coverage is rarely available, which makes rule-based topic assignment methods difficult to transfer to other domains. Tools such as Markó *et al.*'s lexicon for morphological decomposition make the indexing methods completely domain dependent, because for a new domain a new lexicon is required, as well as the training data. Finally, rule-based techniques are prone to errors when assigning vocabulary terms that are less popular—they will receive weaker classifiers than frequent index terms.

3.1.2 Candidate generation and filtering

Term assignment with very large vocabularies can be performed with an alternative approach: candidate generation and filtering. In the first stage, candidate topics are determined by mapping document phrases to vocabulary terms. In the second stage, significant candidates are computed based on their properties. Training data for each vocabulary term is not required. However, the approach introduces new challenges. Mapping implies dealing with language phenomena like synonymy and polysemy. In filtering, meaningful properties that differentiate topics from non-topics need to be discovered. A similar two-stage approach is also used in keyphrase extraction (Section 3.2.1).

Candidate topics for a given document are computed step-by-step. From each sentence of the document, sequences of words up to a predefined length (*n*) are extracted. These sequences, called *n*-grams, are then matched against the vocabulary terms. Prior to matching, both *n*-grams and vocabulary terms may be stemmed and trimmed of stopwords. If the vocabulary contains non-descriptors, document phrases are matched against them to determine the corresponding descriptor. This process is called *semantic conflation*. It represents the main advantage of using controlled vocabularies: terminology across documents is normalized to a standard set of controlled terms. Additionally, vocabulary terms can be accessed via synonyms from lexical resources (Tiun *et al.*, 2001) or by decomposing document phrases and vocabulary terms into morphemes (Markó *et al.*, 2004). If more than one vocabulary match is possible, word sense disambiguation is required. However, none of these approaches report such cases, perhaps because vocabularies are usually domain-specific.

In the next stage, filtering, most approaches apply heuristic methods, where candidate topics are weighted according to some observations. Machine learning techniques that outperform heuristic weighting in keyphrase extraction approaches (Section 3.2) were introduced in my Master's thesis (Medelyan, 2005).

Tiun *et al.* (2001) automatically index webpages with categories in the Yahoo directory. To improve coverage, they augment each Yahoo category with synonyms from WordNet (Fellbaum, 1998). The webpage is split into phrases, which

are matched against titles of Yahoo categories and WordNet terms. Given a set of candidate categories, each category is weighted using the total frequency of phrases mapped to it, the type of the mapping performed (direct or through WordNet synonyms) and the weights assigned to the child nodes. The authors report precision of 30% for mapping 202 documents into 107 Yahoo categories (recall is not mentioned).

Golub (2006) uses a larger vocabulary, the Engineering Information thesaurus, which comprises 800 main terms, referred to as categories. Additionally, there are 20,000 terms linked to one or more category, 11 terms per category on average. After document phrases have been stemmed and stopwords removed, phrases are mapped to terms in the thesaurus, and categories associated with these terms are included in the candidate list. Thesaurus links are used (instead of Tiun *et al.'s* WordNet's synonyms) for semantic conflation. Candidates are then sorted by frequency multiplied by the weight of their location in the document. Location weights are obtained empirically and boost candidates that appear in the title or in metadata. The author reports an F-measure of 26%.

Aronson *et al.* (2000) describe the Medical Text Indexer—a complex system for indexing medical texts with MeSH terms. They first decompose document phrases into letter trigrams and use vector similarity to map them to concepts in the UMLS thesaurus. The UMLS structure allows these concepts to be converted to MeSH terms. The candidates are augmented by additional MeSH terms from the 100 most similar documents in the manually indexed PubMed collection. This introduces an element of text classification, because the test document is compared to all documents in the collection. The final stage weights each term according to where it was detected and combines the weights of semantically similar terms representing a cluster into a final score. In an experiment with 500 full text articles, Medical Text Indexer achieves 60% recall and 31% precision for the top 25 extracted topics (Gay *et al.*, 2005). However, it seems to use the entire PubMed corpus for training, i.e. many millions of manually pre-indexed documents. It is not clear whether the results would hold with less training data.

Markó *et al.* (2004) propose an alternative method for generating candidate topics from MeSH vocabulary. They apply orthographical, morphological and semantic normalization by mapping every document word and vocabulary term onto MorphoSaurus subwords. Candidate topics are MeSH terms mapped to the same subwords as those appearing in the document. The candidates are then ranked using a weighting scheme based on parameters such as the longest match and most significant position in a document. Markó *et al.* report 23% precision and 25% recall when evaluating 4,000 abstracts with 10 MeSH topics each. Their results improve to 30% precision and 33% recall if heuristic weighting is combined with probabilistic filtering, as described in Section 3.1.1.

Performance of candidate generation and filtering approaches does not seem to depend on vocabulary type and size. Tiun *et al.'s* (2001) method is generalizable to other directory trees and Golub's (2006) weighting techniques are easily transferred to other controlled vocabularies. Comparable results are achieved with only a few hundred terms (Tiun *et al.*; Golub) and with a very large vocabulary (Aronson *et al.* 2000; Markó *et al.* 2004). However, the two latter systems are closely tailored to the medical domain and require a huge database of pre-indexed documents for training (Aronson *et al.*) or a specialized manually constructed lexicon for morphological decomposition (Markó *et al.*).

Term assignment by mapping document phrases to vocabulary terms has great advantages over classification-based methods. It generally needs little training data and applies to many domains, so long as the vocabulary structure is similar. However, only a few researchers have used this approach, and its potential is still not fully explored. In keyphrase extraction, similar steps take place and the results are greatly improved by a machine learning model that combines several features. These methods have not yet been integrated with term assignment. This thesis fills this gap by applying supervised keyphrase extraction methods, described below, to term assignment.

3.2 Extracting keyphrases from text

When researchers submit publications to journals or conferences, they are asked to provide keyphrases. The goal of keyphrase extraction is to perform the same task automatically. The keyphrases are not restricted to a pre-defined vocabulary. This provides the advantage of assigning very specific or newly invented terms that might not be included in the vocabulary. The techniques are generally applicable to any domain, regardless of whether a tailored vocabulary is available.

Keyphrase extraction is realized in two stages: candidate generation and filtering, similar to term assignment approaches presented in Section 3.1.2. Methods for candidate generation vary from *n*-gram extraction (Turney, 1999; Witten *et al.,* 1999; Barker and Cornacchia, 2000) to shallow parsing (Hulth, 2004). *N*-gram extraction often results in ungrammatical phrases. Parsing considers part-of-speech information and is more accurate, however it is only available in some languages.

This section groups keyphrase extraction methods based on their filtering techniques. First supervised methods are presented, which rank candidates using a statistical model derived from documents with manually assigned keyphrases. Then heuristic methods are discussed, where ranking is determined using a fixed weighting scheme. After presenting the two approaches individually, we discuss their strengths and weaknesses.

3.2.1 Machine learning

The history of supervised keyphrase extraction began with two competing methods: GenEx (Turney, 1999), developed first, closely followed by Kea (Witten *et al.*, 1999). Kea received more attention because it is publicly available and simple enough to be extended with new features. It serves as the state-of-the-art baseline, over which new systems, including the Maui algorithm developed in this thesis, can potentially improve by using better candidate generation, more features and a different classifier. This section first describes GenEx and Kea, and then other methods that build on these two.

Turney (1999) proposes GenEx, a hybrid genetic algorithm for keyphrase extraction, consisting of two components: Genitor and Extractor. The Extractor is applied to document text in order to determine a set of weighted keyphrases. Candidate keyphrases are all phrases consisting of up to three consecutive words that are not stopwords. The candidates are stemmed by truncation to five characters.[2] Next, each candidate is scored by its frequency multiplied by its position in text. Scores of multi-word candidates are boosted. After selecting the most frequent full form for each stemmed phrase, Extractor presents the top-ranked phrases as output based on 12 numeric parameters, such as the boosting factor for longer phrases and the size of the final keyphrase set. Genitor is a genetic algorithm that uses training data to determine the best parameter settings. Evaluated on 360 articles from various domains, GenEx achieves precision of 24%, while recall is not reported. Human subjects judged 80% of keyphrases as acceptable.

Witten *et al.* (1999) develop Kea, the Keyphrase Extraction Algorithm,[3] based on similar principles but using a different learning technique. In the candidate generation stage, Kea first determines textual sequences defined by orthographical boundaries such as punctuation marks, numbers, and newlines. These sequences are then split into tokens. Next, Kea extracts candidate phrases that consist of one or more words and do not begin or end with a stopword. The minimum and maximum length of a keyphrase can be set by the user. Each candidate is stemmed using the iterated Lovins (1968) stemmer and the most frequent full version is saved for the output. In the filtering stage, two features for each candidate are computed: the TF×IDF measure (a phrase's frequency in a document compared to its inverse frequency in the document collection, discussed in Section 5.2.1) and the position of the first occurrence (Section 5.2.2). A Naïve Bayes classifier (Domingos and Pazzani, 1997) analyzes training data and creates two sets of weights: for candidates matching manually assigned keyphrases and for all other candi-

---

[2] Stemming by truncation is fast, but has disadvantages: words with different meaning are stemmed to the same string (e.g. *center* and *century*), allomorphs are disregarded (*moderate* and *modest* receive different stems) and short words remain unstemmed (e.g. *terms* and *term*).

[3] Information about this early version of Kea is available at *http://www.nzdl.org/kea/*.

dates. In the filtering stage, the overall probability of each candidate being a keyphrase is calculated based on these weights. The candidates are ranked according to their probabilities, and the top ranked phrases are included into the resulting keyphrase set. After training on 100 documents and testing on 500, KEA extracts 0.9 correct keyphrases among the top 5. The authors do not report precision and recall values.

Frank *et al.* (1999) compare Turney's GenEx and Kea directly on the same data sets and find that their precision is similar, but Kea creates the model much faster (it takes minutes instead of hours required by GenEx). They introduce a new feature, called *keyphrase frequency*, which counts the number of times a candidate appears as a keyphrase in the training collection. Adding this feature significantly improves the results. Depending on the corpus, the top 5 Kea's keyphrases contain on average 1.35 or 1.46 correct keyphrases, which corresponds to precision of 27 and 29%, respectively. The results continue to improve as the size of the training collection increases.

Turney (2003) modifies Kea by adding a semantic feature enhancing the coherence of the resulting keyphrase sets. Coherence is computed using Pointwise Mutual Information (PMI) (Church and Hanks, 1989). Turney first ranks candidate keyphrases using Kea's three features (Frank *et al.*, 1999). Next, he uses PMI to compare the similarity of top L candidates to the top K candidates, where $K < L$. PMI is computed from co-occurrences retrieved using a search engine. For the top L candidates, Turney computes how often they co-occur with top K candidates in the search results. These values are added as new features for final processing by the classifier. Evaluated on two different collections, the quality of the resulting keyphrase sets improves. However, querying the search engine significantly slows down the extraction process. Csomai and Mihalcea (2008) compute PMI statistics offline, using co-occurrence in Wikipedia articles—a faster alternative. Their technique is discussed below.

Hulth (2004) proposes both new candidate generation and filtering methods. For candidate generation, she compares the original n-gram extraction with shallow parsing and part-of-speech sequence matching, which extract only valid noun

phrases. The most accurate results are achieved with parsing and the least accurate with *n*-gram extraction. For filtering, Hulth separates TF×IDF into term frequency and inverse document frequency features, adopts Kea's first-occurrence feature, and adds a new feature that records the part-of-speech pattern of the candidate. Certain patterns are more likely to denote keyphrases. Experiments with classifiers, including Naïve Bayes, bagged decision trees and other ensembles of classifiers, show that a combination of several prediction models yields the best results: an F-measure of over 45%, one of the highest reported among keyphrase extraction methods. However, precision and recall are computed based not on the total number of assigned keyphrases, but on those that actually appear in the documents. Therefore Hulth' figures are not directly comparable with others.

Nguyen and Kan (2007) extend Kea with several new features: the part-of-speech sequence as in Hulth (2004), the suffix sequence of the candidate and a binary feature recording whether the candidate is an acronym. They use a classifier to identify document's structural parts, such as *introduction*, *applications* and *references*, and include this information as a nominal feature, which lists parts in which a candidate occurs. Given 120 test documents, the authors achieve somewhat better results than the original Kea baseline: precision improved from 30% to 32% (recall is not reported). Unfortunately, the individual contribution of their new features is not clear.

Csomai and Mihalcea (2008) propose a supervised method for *back-of-the-book indexing*, a task related to keyphrase extraction. They combine common features, such as term frequency, TF×IDF and term length, with novel features, which utilize discourse, syntactic and encyclopedic information. For computing discourse features a shallow parser first, sentence by sentence, extracts noun phrases, which are then treated as nodes in a graph. Edges in this graph are weighted with scores derived using lexical semantic analysis (LSA) and PMI (Turney, 2003). Next, PageRank (Brin and Page, 1998) is applied to find the most central nodes in this graph. After adding new noun phrases from each batch of sentences, the scores are re-computed. Three features record a) how often a noun phrase receives the central

rank across all sentences; b) how often is it central given its total number of occurrences; and c) the maximum centrality it achieves.

Csomai and Mihalcea transform Hulth's (2004) nominal part-of-speech feature into a numeric one by computing the probability of the phrase's part-of-speech pattern to denote a keyphrase. A further encyclopedic feature is the Wikipedia keyphraseness that is also used in Maui (Section 5.2.3). In the evaluation, Csomai and Mihalcea automatically create back-of-the-book indexes for 30 manually annotated books, after training on 259 books. The best results, an F-measure of 28% in both cases, are reported using a multilayer perceptron and a decision tree classifier. The authors note that nearly as good results were achieved with only 10% of the training corpus, 25 books.

Because keyphrase extraction is a clearly defined task and both the data sets and the baseline systems are publicly available, this area has been exhaustively explored in the machine learning community. Many other experiments have been reported but are for space reasons excluded from this overview (e.g. Barrière and Jarmasz, 2004; HaCohen-Kerner *et al.*, 2005; D'Avanzo and Magnini, 2005).

Machine learning provides an elegant solution for the keyphrase extraction task. The methods are subdivided into clear steps: identifying the candidates, defining the features, computing feature values, and finally deciding on the significance of a candidate. Maui's design follows these principles and, like many methods described in this section, builds on the original Kea system (Witten *et al.*, 1999; Frank *et al.*, 1999).

The results achieved by some supervised keyphrase extraction methods are impressive. As reported above, 80% of automatically determined keyphrases are acceptable (Turney, 1999) and over 45% match keyphrases assigned by the authors (Hulth, 2004). However, existing approaches are rarely applied in real-world situations. Some of them are fairly complex, others have unsustainably long processing times; but the main problem is perhaps the requirement of training data. Best results have been reported in experiments incorporating a few hundred manually indexed documents. This is significantly less than in classification-based approaches to topic indexing (Section 3.1), but is still a major obstacle in practice.

3.2.2 Heuristic methods

There are many universally applicable keyphrase extraction techniques that do not require training data; in other words they are *unsupervised*. Researchers manually analyze the data and identify the strongest properties of typical keyphrases, which they then combine into a fixed scoring function. Examples of this kind of ranking were mentioned in Section 3.1.2 where candidate vocabulary terms were filtered to identify the topics. This section presents keyphrase extraction methods that use heuristic filtering.

Barker and Cornacchia (2000) describe one of the earliest keyphrase extraction systems that utilizes part-of-speech information for parsing grammatically correct candidates. They use a dictionary to assign basic part-of-speech tags to each word and then extract all nouns and, optionally, their adjectival or nominal modifiers. All noun phrases computed in this fashion serve as candidate keyphrases. In the filtering stage, Barker and Cornacchia compute the frequency of the head noun in each candidate phrase and keep all candidates with the N most frequent heads. Each candidate is then scored using its frequency multiplied by phrase length and the top K highest scoring phrases are selected as keyphrases. N and K are user-specified thresholds. Evaluation experiments involving human judges have shown that this unsupervised approach performs as well as the more complex GenEx system (Turney, 1999).

A different way of improving candidate generation is proposed by Paice and Black (2003). They add new steps to the standard n-gram extraction process, which results in a stronger conflation factor. Given document n-grams up to a length of four, stopwords are removed and the remaining words are stemmed and sorted alphabetically. For example, similar phrases such as *algorithm efficiency*, *efficiency of algorithms*, *the algorithm's efficiency* and even *the algorithm is very efficient* map to the same "pseudo phrase" *algorithm effici*. The most frequent original form is preserved to display in the final result. This conflation strategy identifies morphological similarity more efficiently than mere stemming and provides a stronger boost factor for the overall score of a phrase group. In the filtering stage, each pseudo phrase is weighted as: $score = W \times (F - 1) \times N^2$, where F is the fre-

quency of the phrase, *N* is the number of words in it and *W* is the sum of their individual frequencies. Next, the best scoring candidate phrases are collected. Paice and Black discuss how to use the generated keyphrases for information extraction, but do not provide evaluation of their method.

Like Barker and Cornacchia (2000), Mihalcea and Tarau (2004) begin the extraction process by annotating the documents with part-of-speech tags. Unlike many other systems that filter candidates using weighting formulas, their unsupervised method uses a graph-based ranking model. First, all nouns and adjectives are extracted and added as nodes in the document graph. Edges are added between those words that co-occur within a pre-defined window. The graphs are constructed from abstracts only and are therefore relatively small. The nodes are weighted iteratively using TextRank, a graph ranking technique similar to PageRank (Brin and Page 1998). The top third of the best scoring nodes are analyzed in the post-processing stage to determine single and multi-word keyphrases. On the same data set, this approach outperforms Hulth's (2003) supervised keyphrase extraction in terms of F-measure (36% instead of 34%), however its recall is much lower (43% instead of 52%). In later work (see Section 3.2.1), Hulth (2004) achieved higher results than the ones in Hulth (2003) used for evaluating TextRank.

Paukkeri *et al.* (2008) propose a language-independent keyphrase extraction method. Instead of the popular TF×IDF weighting, they rank all candidate n-grams up to a length of four words based on counts determined from the multilingual reference corpus Europarl. All n-grams are ranked according to their frequency in the document, divided by their frequency in the reference corpus of the same language. The frequencies are then normalized so that the most frequent phrase receives the count of one and longer *n*-grams have the same distribution as single words. For the evaluation, Paukkeri *et al.* use Wikipedia articles in different languages and treat their internal links as keyphrases. Precision and recall values are better than in the TF×IDF baseline, averaging 15% and 25% respectively across all languages. Note that the average number of links in Wikipedia articles is significantly higher than the number of manually assigned topics in a typical key-

phrase set. Thus, Paukkeri *et al.* perform terminology extraction rather then keyphrase extraction.

Unsupervised methods employ more accurate candidate generation techniques than supervised ones. While the majority of machine learning approaches simply extract word n-grams, heuristic methods compensate the lack of training data by complex analysis using shallow parsing (Barker and Cornacchia, 2000), morphological conflation (Paice and Black, 2003) and reference corpora (Paukkeri *et al.*, 2008). However, unlike supervised methods, they do not take into account characteristics of a particular document set. Depending on the domain and document type, the significance of ranking features may vary. Thus it is questionable whether a fixed ranking function derived from particular documents will perform as well on any collection. Machine learning techniques are more flexible in this respect and are therefore applied in this thesis.

The main disadvantage of both supervised and unsupervised extraction is that the resulting keyphrases are inconsistent. In term assignment (Section 3.1), a prespecified vocabulary controls the terminology for referring to concepts. In keyphrase extraction, topics can only be as consistent as the word choices made by the document's authors. Section 3.4.2 discusses this problem in more detail.

Keyphrase extraction is often applied to related tasks such as terminology extraction (Paukkeri *et al.* 2008), back-of-the-book indexing (Csomai and Mihalcea, 2008), information extraction (Paice and Black, 2003) and text summarization (Mihalcea and Tarau, 2004). Instead of limiting the output to the top scored keyphrases all significant terms in a given document or collection can be extracted. Coherence analysis and lexical patterns can be applied to identify relations between the keyphrases. Text summaries can be generated using sentences containing the keyphrases.

Among the topic indexing tasks, keyphrase extraction has the most clear objective: to extract the main phrases from text, no pre-requisites like a vocabulary are given. This has led to much competition and the creation of many versatile methods and applications. In this thesis, keyphrase extraction is integrated with the less

thoroughly explored area of term assignment and with the newly discovered task of automatic tagging.

3.3 Generating tag suggestions

Automatic tagging differs from keyphrase extraction in that any possible source of terminology can be explored to determine the topics. The goal is to generate tags in a way that matches the choices of human taggers. These methods can be then used for tagging suggestion, to support users in adding more metadata for the public benefit.

Research on automatic tagging began about three years ago and grew independently of existing research on topic indexing. First, the characteristics of self-emerging folksonomies were assessed. Golder and Huberman (2006) analyze the tagging behaviour of users on the social bookmarking website *del.icio.us*. They define tagging as *sensemaking*, a process of categorizing and labeling information through which meaning emerges. This definition applies to topic indexing in general, but in tagging sensemaking usually takes place for the user's own benefit, not as a service for the public. Analysis of tags shows that they overwhelmingly determine the "aboutness" of the document. Only a few rare tags are personal, e.g. attributes (*stupid*, *inspirational*), self-references (*mystuff*), and task organizers (*toread*). Another interesting finding is that most taggers assign general tags first and then augment them with more specific ones.

Golder and Huberman note high variation and inconsistency of tagging. One tagger speaks of *television* whereas another chooses *tv*; two taggers assign *apple* but one refers to a computer company and the other to fruit. Inconsistency also stems from varying degrees of specificity: one article is tagged *JavaScript* whereas another on the same topic is tagged *programming*. For a tag to be useful, people need to agree on its meaning. Otherwise, as in keyphrase extraction, tagging will produce inconsistent results, which lowers their social value. On *del.icio.us*, taggers are supplied with automatic tag suggestions, which address inconsistency and help stabilize the folksonomy. However, suggestions are only available if a bookmark has already been tagged by others.

Brooks and Montanez (2006) analyze tags assigned to blog posts on *technorati.com*. They find that inconsistency is common even among the most frequent tags (*video* and *videos*); tagging data is not separated by language (*games* and *juegos*); and very general, category-like tags are particularly common (*blogs*, *fashion*). To evaluate the effectiveness of tags, they propose statistical analysis. Given blogs tagged with the same term, they create TF×IDF weighted term-vectors and analyze their cosine similarity. The similarity between such blogs is low, but higher than between randomly selected ones. The conclusion is that tags do manage to group blogs by topic, but there is room for improvement.

It is not surprising that volunteer taggers produce tags of bad quality. After all, traditional libraries specifically train professional indexers to assign topical metadata that is of use to their patrons. Automatic tagging might help users to improve their tagging skills by providing good-quality recommendations and by making the task less onerous.

Most automatic tagging methods use the same general strategy: First, given a document, find ones that are similar and already tagged. Next, collect their tags, rank and present them to the user. This approach is similar to the classification-based term assignment discussed in Section 3.1.1 and suffers from the same problem: only pre-existing tags are suggested. Depending on the approach, a different heuristic is applied to identify similar documents or to rank tags to determine the best suggestions.

Mishne (2006) computes similar documents in the repository, but gives additional weight to tags assigned by the user who is using the tagging suggestion tool. Manual evaluation by human judges on 30 blog posts shows a precision and recall of 38% and 47% respectively for automatically assigned tags, compared with a precision of 59% for manually assigned ones (recall for manual tags is not given). It is surprising that Mishne does not consider all manually assigned tags as correct and instead re-evaluates them based on the opinion of human judges.

A more complex ranking heuristic is suggested by Sood *et al.* (2007), who group tags into semantically related bins based on their co-occurrence. They use centroid-based tag clustering to distinguish polysemous meanings. Similar evaluation

as in Mishne (2006) on 225 posts shows that the accuracy of the automatically assigned tags is 42%, only 5 points less than that of manual ones. However, exact matching of the algorithm's tags against manual ones assigned to 1000 posts resulted in much lower precision and recall: 13% and 23%.

Chirita *et al.* (2007) aim to extract personalized tags by retrieving similar documents from a user's computer desktop. Because these documents are untagged, a keyphrase extraction-like approach is applied. *N*-grams appearing in these documents are ranked using a mixture of scoring methods, such as term and document frequency, lexical dispersion, sentence scoring, and term co-occurrence. They report that the best performing formula combines term frequency and first occurrence, but do not cite the keyphrase extraction research, where both features were originally introduced (e.g. Witten *et al.*, 1999, Section 3.2.1). On the tagging task, this scoring yields a precision of 80% for the top 4 tags assigned to 96 test pages. Budura *et al.* (2008) describe a similar approach with a scoring formula combining three features: tag frequency, tag co-occurrence and document similarity.

Surprisingly, none of these researchers cite the richly explored term assignment and keyphrase extraction literature. Features like TF×IDF and first occurrence features are re-invented in tagging. At the same time, keyphrase extraction methods do not make use of rapidly evolving collections of tagged documents on collaborative sites. While supervised keyphrase extraction methods suffer from the lack of manually annotated training data, tagged documents are available on the web in abundance.

Tagged collections have potential advantages over training data normally used in keyphrase extraction and term assignment methods. These methods are usually restricted to keyphrase sets assigned by just one author. In tagging, multiple tag sets per document are often available, so that the most agreed on topics can be determined. The more users assign a tag to a document, the more important it is. In topic indexing, multiple topic sets provide additional information for learning and allow a deeper evaluation, where automatically assigned topics can be compared to choices of several humans. We make use of these observations as described below.

3.4 Filling the gaps

Three major types to automated topic indexing have been discussed so far: term assignment, keyphrase extraction and tagging. They all address the task of identifying the main topics in text, but have slightly different formulations due to differences in the application areas: traditional libraries, scientific publications, and the web. Table 3.1 summarizes the characteristics of each approach and Table 3.2 lists their pros and cons. The comparison shows that despite the similarity of the tasks, each area grew independently, resulting in different methods and many missing connections. For example, useful features have not yet been explored in all tasks, and the latest data sets and resources created on the web are still not used by traditional approaches. This thesis shows how establishing links between the areas can provide new solutions.

3.4.1 Avoiding large training data in term assignment

In automatic *term assignment*, topics are derived from a controlled vocabulary, which produces consistent and accurate indexing results. However, most methods are classification-based and require very large training corpora to learn a model for each vocabulary term (Dumais *et al.*, 1998; Sebastiani, 2002; Pouliquen *et al.*, 2003) or to support the system with probability estimates (Plaunt and Norgard, 1998; Aronson *et al.*, 2000; Marko *et al.*, 2004). As described in Section 3.1.2, some methods instead adopt a two-step strategy of identifying candidate topics and then filtering out the most significant ones based on their properties (Tiun *et al.*, 2001; Golub, 2006). Surprisingly, these methods do not apply machine learning in the way that has been proposed in *keyphrase extraction* research (Witten *et al.*, 1999; Hulth, 2004). Although this approach still requires training data to learn the typical distribution of property values, good results can be achieved with far fewer examples (Frank *et al.*, 1999; Csomai and Mihalcea, 2008).

	SUBJECT INDEXING AND TERM ASSIGNMENT		KEYPHRASE AND TERMINOLOGY EXTRACTION		TAGGING
	Classification and rule building	Candidate generation and filtering	Machine learning filtering	Heuristic filtering	
VOCABULARY	small to medium	medium to large	no	no	no
TRAINING	Large training sets to create a model for each vocabulary term	Can be required to compute mapping probabilities	Small to mediums sets to train keyphrases vs. non-keyphrases based on their features	No training required; optional: reference corpus for occurrence statistics	Large training sets required for previously assigned tags
CANDIDATE TOPICS	Previously assigned terms only	Mapping *n*-grams to vocabulary, directly or via synonyms	• *N*-gram extraction • Part-of-speech pattern matching • Shallow parsing		Previously assigned tags only or *n*-gram extraction
FILTERING	• Manual or automatic rule induction • Cosine similarity of document vectors	• Heuristic weighting combining frequency and position • Combining with classification results	• Genetic algorithm for tuning parameters • Naïve Bayes or other method for classifying keyphrases vs. non-keyphrases	• Manual parameter tuning for weighting • Graph-based ranking using co-occurrence statistics	• Cosine similarity of document vectors • Heuristic weighting combining frequency and position

Table 3.1 Comparison of methods used in different types of topic indexing

	SUBJECT INDEXING AND TERM ASSIGNMENT		KEYPHRASE AND TERMINOLOGY EXTRACTION		TAGGING
	Classification and rule building	Candidate generation and filtering	Machine learning filtering	Heuristic filtering	
ADVANTAGES	• Consistent • Accurate results • Semantics expressed through co-occurrence statistics • Terms don't need to appear in the document	• Consistent • No training required • Independent of vocabulary size • Semantics expressed through links in the vocabulary	• No vocabulary needed • Small training sets • Clearly defined • Sensitive to domain and collection specifics • Easily extendable • Intuitive	• No vocabulary needed • No training required • Domain independent • Easy implementation • Light-weight • Fast processing	• Freedom in choosing the tags • Large training sets easily available • Feedback from taggers is possible
DISADVANTAGES	• Only for small vocabularies • Training required • Very large training sets • Long processing time	• Complex mapping • Depends on vocabulary quality • No connection to keyphrase extraction research	• Inconsistent results • Ungrammatical and ill-formed keyphrases • No semantics • Training required	• Inconsistent results • Ungrammatical and ill-formed keyphrases • No semantics • Domain dependent • Difficult to improve	• Inconsistent results • No semantics • Training required • Only pre-assigned tags possible • No connection to previous research

Table 3.2 Advantages and disadvantages of approaches to automatic topic indexing

The Maui algorithm proposed in this thesis implements a new approach to term assignment that avoids the disadvantages of classification-based approaches and instead follows the strategy of supervised keyphrase extraction. Characteristics of candidate index terms, rather than those of documents, are analyzed to determine whether they are good topics. Instead of a fixed scoring scheme, machine learning is applied to account for domain-specific differences. The method does not require training data for each vocabulary term and thus can be applied to any domain and vocabulary size so long as a small set of manually indexed documents is available.

3.4.2 Adding consistency to keyphrase extraction

The main advantage of keyphrase extraction and tagging is that they do not require a controlled vocabulary. This offers great flexibility and provides solutions when vocabularies are not available. However, *n*-gram extraction often produces ill-formed and ungrammatical candidates. More importantly, there is no homogeneity among the extracted keyphrases, which depend on the word choices of a document's author. Documents that describe the same topic in different but synonymous words (e.g. *seaweed culture* and *sea weed farming*) receive different keyphrases and cannot be grouped according to their content. Controlled vocabularies, on the other hand, keep indexing consistent by linking terms that mean the same thing to the same concept. Vocabularies also resolve polysemy through broader and narrower links between terms, and provide easy access to documents on the same topic stored in one place. But such vocabularies are expensive to construct and maintain, and they are only accessible in a few domains.

Along with the new term assignment method, Maui implements a novel approach to keyphrase extraction. It utilizes the online encyclopedia Wikipedia as a universal source of well formulated topics. Wikipedia covers many domains at a great level of specificity, not only in English but also in most other languages. This solves many problems of traditional keyphrase extraction, such as inconsistency, poorly formulated phrases and lack of semantics. The new method follows the two-stage approach of candidate generation and filtering. In the first stage, an algorithm for mapping document terms to Wikipedia articles is proposed (Section 5.1.2). In filtering, traditional features are used, as well as several new ones derived

from Wikipedia's well-structured corpus. The method is extensible, intuitive and applicable to any domain and language[4] covered by Wikipedia.

The idea of using Wikipedia for describing documents has been picked up by several other researchers at the same time. Milne *et al.* (2006) discuss using Wikipedia as an alternative to domain-specific thesauri used for indexing. Milne *et al.* (2007) use Wikipedia to create a structured knowledge base of topics discussed in a document collection for improved search retrieval in this collection. Csomai and Mihalcea (2007) describe an approach of linking concepts that appear in educational documents to Wikipedia articles to improve the learning efficiency. Later they generalize the approach with a system Wikify (Mihalcea and Csomai, 2007).

Finally, Grineva *et al.* (2009) present an approach that is the closest to the one presented in this thesis. Their system, like Maui, uses Wikipedia as a vocabulary for topic descriptors. However, the filtering is performed differently. After candidate topics are extracted Grineva *et al.* apply a community-detection algorithm to group them semantically. The groups are then ranked based on semantic similarity and keyphraseness of its members and the most significant community members are chosen as topics. They achieve an recall of 68% and precision of 46% for top five terms assigned to 250 documents. In Section 7.3 we compare the performance of this algorithm to Maui.

3.4.3 Competing with human taggers

The thesis also improves current automatic tagging techniques, which have so far been limited to training-intensive classification or simple heuristic weighting approaches. Given a corpus of collaboratively tagged documents, we apply a state-of-the-art keyphrase extraction baseline and improve its performance by adding new features. The goal is to produce tags on which the majority of taggers would agree.

This experiment also addresses the problem of evaluating keyphrase extraction on keyphrase sets assigned by just one person. Because topic indexing is a very subjective process, it is not enough to match just one indexer's topics. They might

[4] For some languages, like Chinese, Japanese and Thai, additional tools for word segmentation are required.

be of low quality or not the only correct ones. On collaborative sites several human taggers simultaneously index the same document. It seems wrong to ignore agreement and disagreement of taggers on particular terms that arises naturally in such settings.

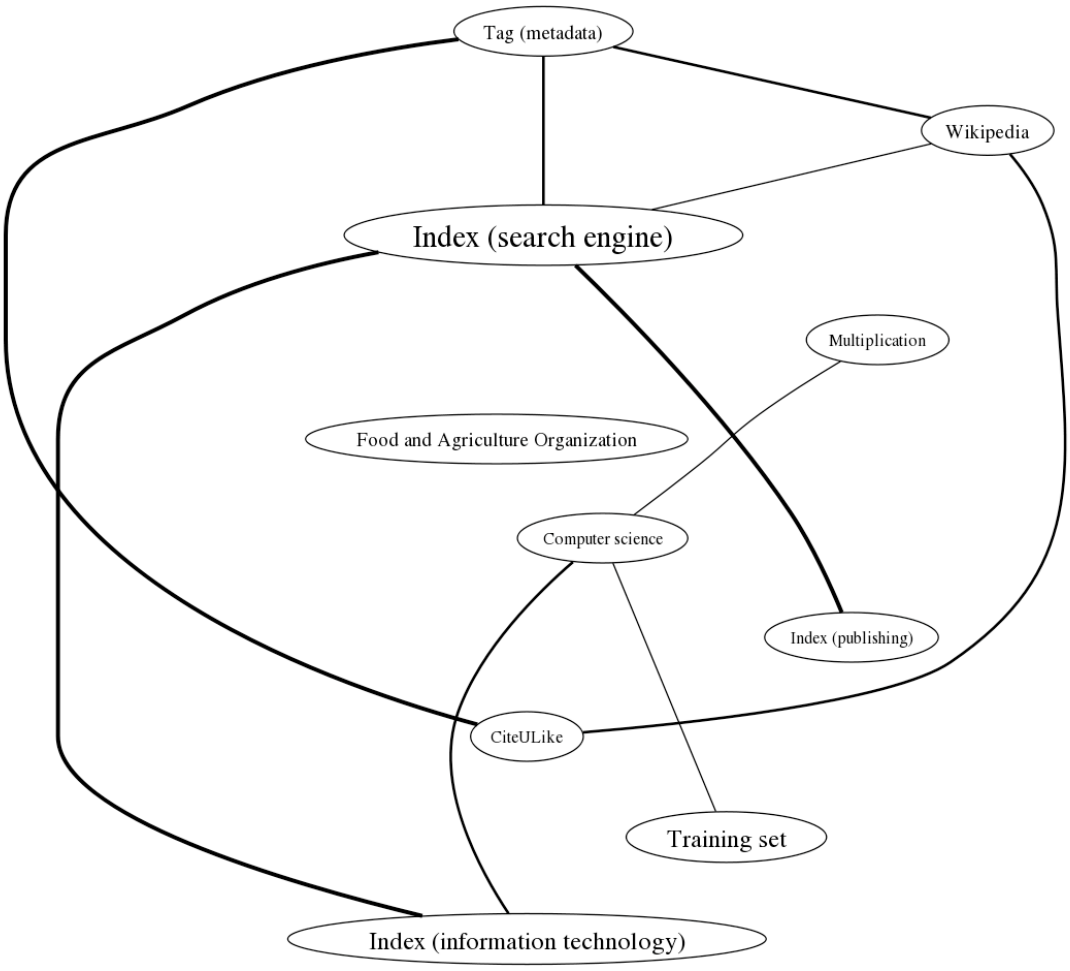
A collection of documents with reliable tags can be automatically extracted from a collaboratively tagged corpus by identifying tags on which at least two taggers have agreed. This produces multiply indexed collections of high quality, like those traditionally used in libraries to assess indexing quality. Using indexing consistency analysis, best performing taggers can be identified (Section 6.3). The Maui algorithm is then evaluated by comparing its consistency with these taggers.

3.4.3 Learning more about the features

A large number of statistical, semantic, encyclopedic, syntactic and discourse-based features has been explored in the literature. However, in most cases the contribution of each feature to topic indexing performance remains unclear. Whereas in supervised keyphrase extraction individual evaluation of features was reported (Frank *et al.*, 1999; Hulth, 2004; Csomai and Mihalcea, 2008), development of term assignment methods has been rather ad hoc. This thesis integrates into Maui the best performing features for both term assignment and keyphrase extraction and evaluates each feature individually on various data sets (Section 5.2.6).

3.5 Summary

Methods surveyed in this chapter have a common goal: they attempt to automatically determine the main topics in a document just as professional indexers, authors and users do it manually. Depending on the application, different techniques are applied to achieve this goal. This thesis shows how these techniques can be combined in a way that capitalizes upon their advantages, while avoiding their limitations.


Tag (metadata)
Wikipedia
Index (search engine)
Multiplication
Food and Agriculture Organization
Computer science
Index (publishing)
CiteULike
Training set
Index (information technology)

Chapter 4
Data sets
and human performance

This thesis shows how automatic topic indexing can produce human-competitive results. Chapter 3 surveyed existing approaches to topic indexing, identified gaps in the research literature and prefigured the tasks addressed in this thesis. This chapter examines these tasks in more detail from the perspective of experimental data. It also focuses on the disadvantage of commonly used evaluation strategies, which indicate whether an algorithm performs well or not, but do not take into account human performance on the same task. Section 4.1 analyzes domain-specific thesauri used in term assignment, and assesses the indexing performance of professionals. Section 4.2 compares the structure of Wikipedia to that of a thesaurus and juxtaposes human efforts in assigning Wikipedia terms with the topic indexing performance of professionals. Section 4.3 uses collaboratively tagged data to create a multiply indexed corpus for testing automatic tagging. Insights into how humans perform topic indexing in each of the three tasks suggest guidelines for creating and evaluating indexing algorithms.

4.1 Term assignment

One of the contributions of this thesis is a new method for automatic term assignment. It requires little training data, is general, and does not depend on the specifics of a particular vocabulary. The method is tested on three vocabularies introduced in this section. Each covers a different domain and is commonly used in digital collections.

Because the goal is to produce an algorithm that assigns vocabulary terms as accurately as humans, we first need to assess how well humans perform. After presenting the vocabularies and the experimental data, we analyze the indexing consistency of several professional indexers. The result defines a performance level that the new algorithm needs to achieve in order to be considered human-competitive.

4.1.1 Vocabularies and corpora

The domain independence of the proposed method is demonstrated using three vocabularies: the agricultural thesaurus Agrovoc (1992), the Medical Subject Headings (2005), and the High Energy Physics thesaurus.[1] Each is encoded using the SKOS (Simple Knowledge Organization System) format,[2] an RDF-based format for defining terms and semantic relations between them that also supports multilingual thesauri. SKOS is well established and the number of knowledge structures available in this format continues to grow. With this unified encoding, any vocabulary can be easily plugged into Maui.

Agrovoc thesaurus

Agrovoc is a multi-lingual thesaurus covering agriculture, forestry, fisheries, food and related domains (e.g. environment). It has been developed by the UN Food and Agriculture Organization (FAO), which maintains a large and well used online document repository (1M hits per month).[3] Professional indexers at FAO manually assign terms from Agrovoc to all documents in this repository.

The English Agrovoc defines over 28,000 concepts. As the example in Figure 4.1 illustrates, each has one preferred term (descriptor), and many have several alternative versions (non-descriptors), resulting in a total size of around 40,000 terms. The vocabulary has been translated into 23 languages. The first three rows of Table 4.1 below list the sizes of the English, French and Spanish versions used here.

---

[1] *http://www-library.desy.de/schlagw2.html*
[2] *http://www.w3.org/2004/02/skos/*
[3] *http://www.fao.org/documents/*

English Descriptor	**Epidermis**
Scope Note	Of plants; for the epidermis of animals use SKIN
Broader Terms	**BT1** Plant tissues
	BT2 Plant anatomy
Narrower Terms	**NT1** Plant cuticle
	NT2 Plant hairs
	NT3 Root hairs
	NT2 Stomata
Related Term	**RT** Peel
French Descriptor	Epiderme
Spanish Descriptor	Epidermis

Figure 4.1 Entry for *Epidermis* in the Agrovoc thesaurus

The concepts in Agrovoc are interconnected by 83,000 semantic relations of three types: related terms (RT, is-associated-with), which is a bi-directional relation, and broader (BT, has-parent) and narrower terms (NT, has-child), which are inverse. The BT and NT links build a hierarchical structure that has seven specificity levels.

Note that all thesauri used in this thesis are domain specific and rarely contain ambiguous terms. In Agrovoc, such terms are marked by brackets, e.g. *Vanilla (genus)* and *Vanilla (spice)*. Where necessary, a scope note describes the intended meaning. There are only 400 (less than 1.5%) such ambiguous terms among 28,170 English descriptors. A similar picture is observed for French and Spanish (see the last column in Table 4.1).

Agricultural document collections

The first corpus comprises 780 full-text documents selected randomly from the FAO's repository, referred to below as *FAO-780*. The documents average 30,800 words (a total of 24 million), ranging from 1200 to 257,000 words. The FAO indexers have assigned an average of 8 Agrovoc descriptors to each document, ranging from 2 to 23. This total of 6225 term assignments includes 2187 different terms.

Terms appearing in FAO-780's topic sets cover only 8% of Agrovoc's descriptors. This means that classification-based approaches described in Section 3.1.1 would create models for only this tiny subset of the thesaurus, and other terms would never be assigned to new documents. Maui learns the properties of typical

	Total terms	Descriptors	Non-descriptors	Ambiguous
English Agrovoc	38,200	28,170	10,030	400
French Agrovoc	37,350	28,160	9,190	440
Spanish Agrovoc	40,640	28,160	12,480	620
MeSH	141,220	23,890	117,330	380
HEP	16,460	16,000	460	15

Table 4.1 Size of thesauri used in this thesis

topics, as opposed to properties of specific topics, and can potentially assign any vocabulary term to a new document, regardless of whether it ever appeared as a topic in the training set.

The second corpus with 30 new agricultural documents (FAO-30) is used to determine the inter-indexer consistency of professional indexers. Each document has been independently indexed by 6 people, with an average of 10.4 Agrovoc terms per set, ranging from 4 to 52 terms. This dataset has been created at the FAO specifically for the experiments in this thesis. Section 4.1.2 analyzes the indexing consistency of these professionals, and Section 7.2.5 investigates the consistency of the algorithm with the indexers.

To test Maui's language independence, French and Spanish collections were extracted from the FAO repository. Manually indexed documents in languages other than English are rare at FAO; thus these collections are rather small: 47 Spanish documents, averaging 42,500 words; and 60 French documents, averaging 22,400 words. The documents had been indexed with English terms, which we mapped to the equivalent Spanish and French terms using Agrovoc. The resulting sets contained 2 to 35 topics each, an average of 10.2 topics for Spanish and 11.4 for French documents.

Medical Subject Headings thesaurus

The Medical Subject Headings (MeSH) thesaurus was discussed in Section 2.1. The U.S. National Library of Medicine (NLM) developed this vocabulary for indexing

the PubMed repository. The SKOS version was provided by van Assem *et al.* (2006).[4]

MeSH contains 24,000 concepts organized into a hierarchy via 32,000 BT/NT links. Descriptors in MeSH are called subject headings and are usually accompanied by several non-descriptors (entry terms). Whereas Agrovoc only defines synonymous non-descriptors, MeSH also includes spelling and formatting variants, resulting in a total of 141,000 terms (see Table 4.1). This much larger vocabulary tests not only Maui's domain independence but also its scalability.

The experimental corpus, provided by the NLM Indexing Initiative (Aronson *et al.*, 2000 and Gay *et al.,* 2005, Section 3.1.2), consists of 500 documents. This collection, *NLM-500* is heterogeneous with lengths varying from 440 to 24,500 words (4500 on average) and the number of assigned topics ranging from 2 to 30 (15 on average).

High Energy Physics thesaurus

For the physics domain, the Deutsches Elektronen-Synchrotron developed the High Energy Physics (HEP) thesaurus. The European Organization for Nuclear Research uses it for indexing the contents of the CERN Document Server.[5] This thesaurus is the smallest of the three used in this thesis, listing 16,000 concepts with rare non-descriptors. Beside 500 broader, narrower and related links, HEP defines a semantic relation called *Composite/CompositeOf*. For example, the concept *Einstein equation: solution* has two CompositeOf relations: *Einstein equation* and *Solution*. In total 15,300 such links are defined.

The experimental corpus (*CERN-290*) comprises 290 random documents from the CERN Document Server, each on average 6,300 words long. The topic sets contain 7 terms on average.

Size statistics

Table 4.1 summarizes the thesauri described in this section. Their sizes range from 16,460 (HEP) to 141,220 terms (MeSH). Some define a wide range of non-

[4] *http://thesauri.cs.vu.nl/*
[5] *http://cdsware.cern.ch/tmp/bibclassify/hep.html*

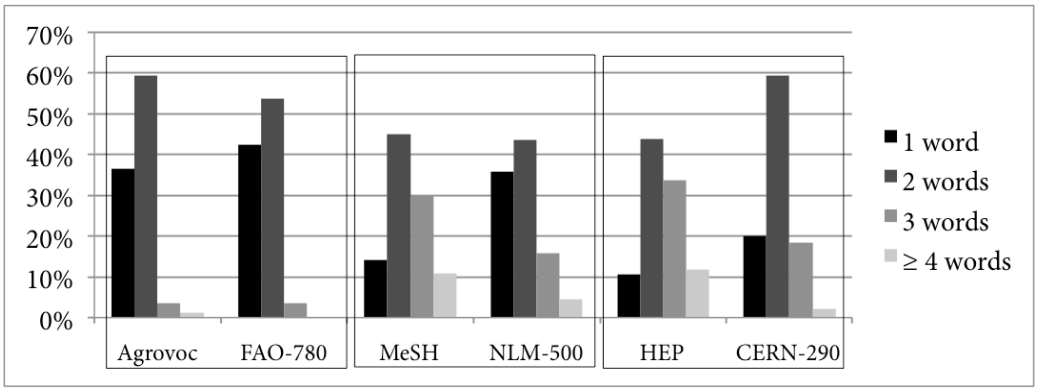


Figure 4.2 Term length distribution in different corpora

descriptors (MeSH); others only a few (HEP). Little ambiguity was observed: less than 2% in each case. Most ambiguous terms have just two meanings each.

Figure 4.2 compares the distribution of term lengths (in words) in each thesaurus and the topic sets manually assigned to the corresponding collections. Agricultural terms are significantly shorter than both medical and physics ones. Two-word terms make up the majority of terms in all thesauri and corpora, and across all corpora, there is a trend towards the shorter topics in the vocabulary. In NLM-500 single words are much more common than in MeSH, whereas in CERN-290 two-word terms are the most popular topics. Although descriptors with 4 or more words are more common in the HEP thesaurus than in any other vocabulary, they are rarely chosen as topics. The maximum term length in the vocabularies ranges from 7 (Agrovoc) up to 15 words (MeSH). The maximum length in corpora is 5 words in FAO-780 and CERN-290 and 6 words in NLM-500.

4.1.2 Consistency of professional indexers

Knowing the performance of professional human indexers is important for evaluating an indexing algorithm. The quality of one indexer's topics can be evaluated by comparing them with topics assigned by several other people to the same document. Section 4.1.1 describes FAO-30, a data set with 30 documents each indexed by 6 professional indexers. In order to assess their indexing quality, we apply the inter-indexer consistency measure explained in Section 2.2. Although a larger document set would give more reliable results, such collections are expensive

	Min	Max	Mean	St.deviation
Indexer1	8	19	12.7	3.0
Indexer2	8	19	11.0	2.4
Indexer3	3	8	4.7	1.2
Indexer4	4	14	8.6	2.5
Indexer5	9	52	17.4	9.1
Indexer6	5	11	7.6	1.4
Average	6.2	20.5	10.3	3.3

Table 4.2 Number of topics assigned by indexers to each document in FAO-30

to construct and are rarely available. The experiment with 30 documents provides at least a rough idea of human performance, to which Maui can be compared.

Exhaustiveness and specificity

One of the indexing characteristics is *exhaustiveness*, which quantifies the number of topics identified for each document. The FAO indexers assigned between 3 and 52 index terms per document. Table 4.2 summarizes the statistics for each indexer. Indexer5 is unusually exhaustive, assigning 7.3 more terms to each document than the average indexer, while Indexer3 is less exhaustive than others, assigning 5.6 fewer terms than the average and four times less than the Indexer5.

Imagine that Indexer5 and Indexer3 have indexed two parts of the FAO repository of equal size and similar topic distribution. A search for a specific topic would retrieve a document indexed by Indexer5 four times as often as one indexed by his colleague. This, of course, would not reflect the true contents of the repository. While both indexers' choices may be correct, it is important for them to be similarly exhaustive. An algorithm should aim to approximate the average exhaustiveness of all indexers in the group.

The second criterion of indexing is *specificity*, which reflects how specific or general are the assigned topics. Generally speaking,[6] single words are more general than multi-word expressions (e.g. *Forest* ⊃ *Rain forest* ⊃ *Tropical rain forest*). Here again, agreement is important. If two documents about *Tropical rain forest* are

[6] Of course, not all single words are general and not all longer phrases are specific (e.g., *Invertebrate zoology* ⊃ *Arachnology*). Term length statistics presented here are approximate not exact indicators of specificity.

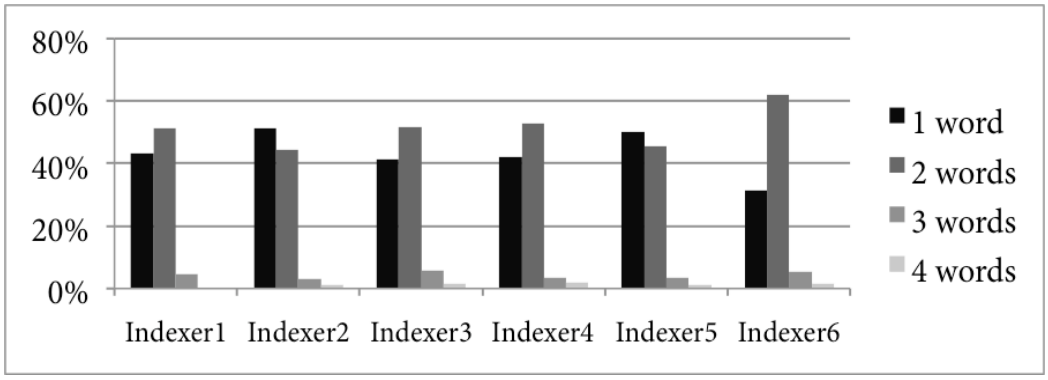


Figure 4.3 Length of terms assigned to documents in FAO-30

indexed with terms of different specificity, the user will struggle to find them using just one search term.

Figure 4.3 compares the specificity of the FAO indexers using the distribution of term length in their topic sets. Indexer1, Indexer3 and Indexer4 have similar distributions. Their profiles align with those computed for the FAO-780 corpus shown in Figure 4.2. Indexer2 and Indexer5 prefer to assign single-word terms, and Indexer6 is slightly more specific than the rest of the group, with a larger percentage of two-word or longer terms.

Inter-indexer consistency

Another important criterion of indexing quality is *inter-indexer consistency*. Unlike consistency in exhaustiveness and specificity, inter-indexer consistency measures agreement on the actual topics assigned to a document, not just agreement on their properties. In term assignment, each vocabulary term serves as a topic. It is important that human indexers agree on what terms describe a document, because the degree of agreement determines retrieval effectiveness (Section 2.2).

Inter-indexer consistency depends on the number of unique index terms assigned to each document. A merged topic set assigned by all indexers contains on average 32.2 terms, whereas the average size of a topic set assigned by just one indexer is 10.3 (Table 4.2). This means that approximately two-thirds of terms are unique, which indicates low agreement between professional indexers.

Table 4.3 summarizes the inter-indexer consistency values of the FAO-30 indexers, computed using the Rolling measure defined in Section 2.2. The overall

Indexers	1	2	3	4	5	6	Average
1		45	42	46	40	46	43.8
2	45		35	36	43	34	38.6
3	42	35		40	26	34	37.1
4	46	36	40		37	35	38.9
5	40	43	26	37		33	35.3
6	46	34	34	35	33		38.5
						Overall	**38.7**

Table 4.3 Inter-indexer consistency for FAO-30

consistency of this group is 38.7%. Indexer3 and Indexer5 exhibit the lowest pairwise consistency: 26% (line 3, column 5). Indexer1 has the highest average consistency with the group, 43.8% (line 1, column *Average*), while Indexer5 has the lowest, 35.3% (line 5, column *Average*). Interestingly, the exhaustiveness and specificity of Indexer1 match the overall statistics of the group better than those of other indexers, whereas Indexer5 shows the most irregular indexing behavior.

Idiosyncrasy

Indexing consistency depends on the overlap between topic sets assigned by different indexers to the same documents. In total, indexers assigned 1864 terms to all documents, of which 967 were unique. The majority of topics (550, or 57%) were assigned to documents by a solitary indexer, and only 38 topics (4%) were agreed on by all indexers.

To study the agreement of indexers on certain topics, we evaluated each indexer's predilection for assigning topics that no one else assigns to that document—index terms that are *idiosyncratic* to this particular indexer. The number of such terms varied between indexers from an average of one per document to more than four, and averaging over all indexers, a quarter (25.6%) of each indexer's topic choices were idiosyncratic. The least consistent indexer (Indexer5) is the most idiosyncratic one. However, high consistency does not mean low idiosyncrasy, or vice versa. Indexer3 shows low consistency with others (37.1 vs. 38.7 in Table 4.3), but only 10% of his index terms are idiosyncratic. Indexer3 assigned fewer terms per document than his colleagues, and low exhaustivity led to low idiosyncrasy.

Experiments assessing the quality of human indexing using controlled vocabularies report consistency values between 13% and 70%, with an average of 44.3% (Leininger, 2000). FAO's indexers achieve 38.7%, slightly lower than average. The analysis shows that the best performing indexers have higher exhaustiveness than the average of the group and assign fewer idiosyncratic terms.

4.2 Indexing with Wikipedia

Current topic indexing research has not yet addressed an important disadvantage of keyphrase extraction (as opposed to term assignment): the lack of consistency among the generated keyphrases. Adding consistency to keyphrase extraction requires a major modification: a vocabulary that defines equivalent terms. This vocabulary needs to be universally applicable to any document collection. Maui utilizes the online encyclopedia Wikipedia as such a vocabulary.

Indexing using topics from Wikipedia is a novel approach. A thorough analysis of the encyclopedia and of the indexing task is necessary to assess whether this approach is feasible. Does Wikipedia meet the requirements of controlled vocabularies used for indexing? Wikipedia's structure and content have to be compared to a traditional thesaurus used in term assignment. How well would a human perform indexing with topics from such a large vocabulary as Wikipedia? An experiment with human subjects is required to evaluate their performance, as well as to create experimental data for developing an algorithm for this task. To be considered human-competitive, an algorithm for indexing with Wikipedia should match the consistency of the human subjects.

4.2.1 Wikipedia as a controlled vocabulary

Wikipedia was launched in 2001 with the goal of building free encyclopedias in all languages. Today it outstrips all other encyclopedias in size and coverage, and is one of the most visited sites on the web. Out of more than three million articles in 125 different languages, one-third are in English, yielding an encyclopedia almost ten times bigger than the *Encyclopedia Britannica*, its closest rival. As an open source project, the entire content of Wikipedia is easily obtainable in the form of

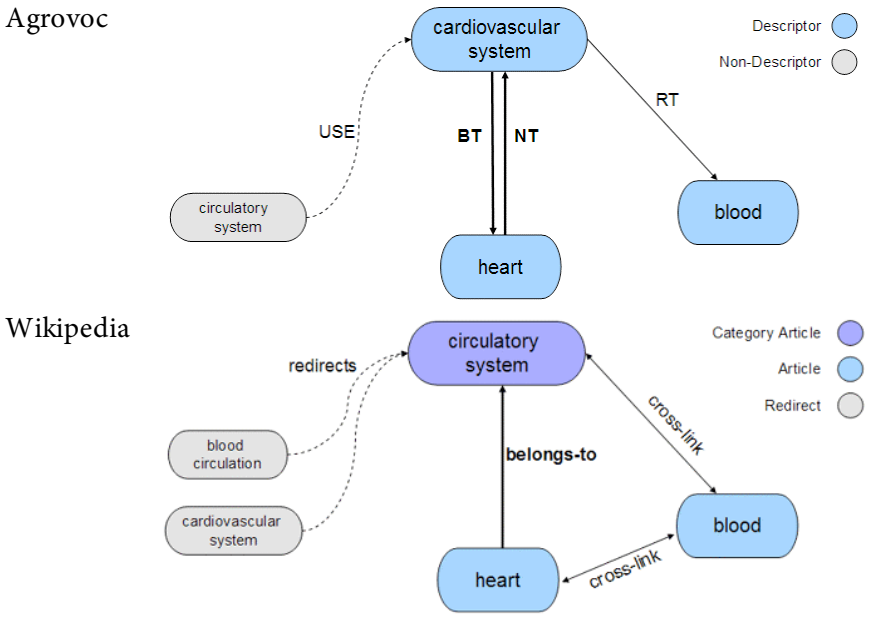


Figure 4.4 Structures in Agrovoc and Wikipedia

database dumps that are released periodically. The version used in this thesis was released on May 6, 2009.

In the last few years, Wikipedia was discovered as an alternative to the lexical resource WordNet for designing many natural language processing tasks (Medelyan *et al.*, 2009). One of the first experiments analyzing the structure and the coverage of Wikipedia showed it to be remarkably similar to a traditional indexing thesaurus, Agrovoc (Milne *et al.*, 2006). This section summarizes the findings of that study. Figure 4.4 begins the comparison with two excerpts from Agrovoc and Wikipedia. The structures are nearly identical, but a closer analysis reveals some differences.

Synonymy

The relation of equivalent meaning, or *synonymy*, is necessary to bridge variety of idiolects and terminology present in a document collection. Section 4.1.1 described non-descriptors and entry terms encoded in Agrovoc and MeSH thesauri to cover synonymous terms. Likewise Wikipedia ensures that there is a single article for each concept by using "redirects" to link equivalent terms to a preferred one, namely the article's title. Wikipedia's scheme copes with capitalization and spelling variations, abbreviations, synonyms, colloquialisms, and scientific terms.

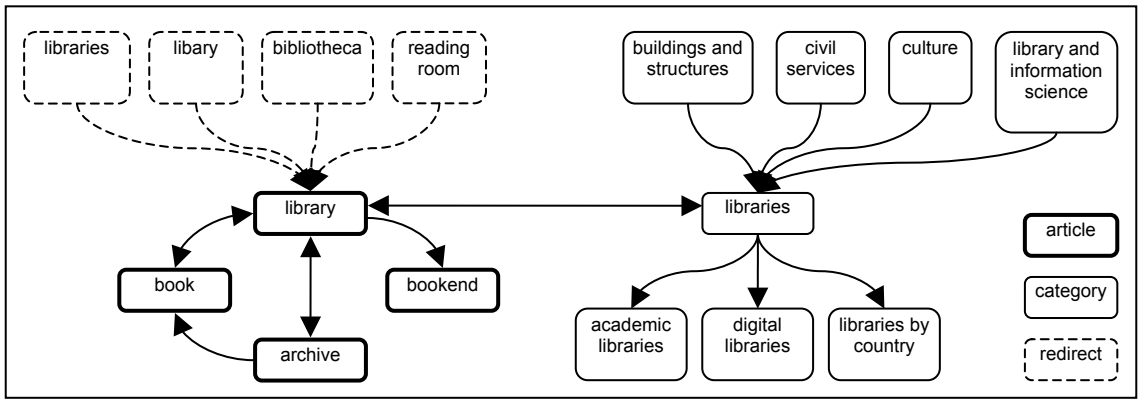


Figure 4.5 Representation of the term *Library* in Wikipedia

The top left of Figure 4.5 shows four redirects for the article *Library*: the plural *Libraries*, the common misspelling *Libary*, the technical term *Bibliotheca*, and a common variant *Reading room*.

Mapping Agrovoc's synonymy relation to Wikipedia shows that only 5% of Agrovoc's relations are absent from Wikipedia, and Wikipedia redirects cover 75% of those that are present. Examples from the remaining 25% indicate that Wikipedia separates such pairs into distinct articles rather than treating them as synonyms, e.g. *Aluminum foil* and *Shrink film* and *Spanish West Africa* and *Río de Oro*. Agrovoc judges these concepts to be "near enough" and does not list separate entries.

Hierarchical relations

The hierarchical organization of a thesaurus is reflected in Wikipedia's categorization structure. Authors are encouraged to assign categories to their articles, and the categories themselves can be assigned to other more general categories. The right-hand side of Figure 4.5 shows that the article *Library* (left) has a corresponding category *Libraries* (right), which contains several more specific subcategories and articles, such as *Academic libraries* and *Digital libraries*. Wikipedia's category structure does not form a simple tree-structured taxonomy but is a graph in which multiple organization schemes coexist.

Although 69% of Agrovoc's hierarchical relations are covered by Wikipedia, only 25% appear in the category structure: the remaining 44% were found in redirects and hyperlinks between articles. Coverage greatly improves if transitive relations

are considered. For example, a category relation *Oceania* ⊃ *American Samoa* is implied by the chain *Oceania* ⊃ *Oceanian countries* ⊃ *American Samoa*.

Associative relations

Hyperlinks in Wikipedia express relatedness between articles. For example, the lower left of Figure 4.5 shows hyperlinks between the article *Library* and those for *Book*, *Archive*, and *Bookend*, some of which link back. Articles are peppered with such connections, which can be explored to mine the associative relations that are present in thesauri.

Mapping of Agrovoc's related term relation (RT) to Wikipedia resulted in an overall coverage of 56%. Mutual links between articles were expected to match RT relations closely. However, only 22% were found in this way; the remaining 34% were found within one-way links or the category structure.

Terminology

One may argue that Wikipedia is not specific enough to reflect domain-specific terminology as well as conventional thesauri created by the experts in the field. Milne *et al.* (2006) quantify this assumption by analyzing how well Wikipedia covers all Agrovoc terms and those assigned to documents in FAO-780 collection.

Wikipedia covers approximately 50% of Agrovoc and misses many terms that are generally scientific or highly specific multi-word phrases such as *Margossa*, *Bursaphelenchus* and *Flow cytometry cells*. For general terms located at the top of Agrovoc's hierarchy, coverage is around 75%. Interestingly, analysis of appearances of Agrovoc terms in sample documents demonstrates exactly the same pattern: highly specific terms that are missed by Wikipedia rarely occur in documents. Out of 1560 distinct Agrovoc terms assigned to FAO-780 by professionals, 75% had corresponding Wikipedia articles—Wikipedia covers most Agrovoc terms that are likely to represent topics.

Ambiguous terms

Mapping of Agrovoc terms to Wikipedia has shown that one third of the terms found in both structures are ambiguous according to Wikipedia; they match multiple articles. For example, the Agrovoc term *Viruses* relates to articles like *Biologi-*

cal viruses and *Computer viruses*. Whereas in domain-specific thesauri ambiguity was not a big issue (Section 4.1.1), Wikipedia contains multiple meanings for most words and phrases.

For the convenience of its readers, Wikipedia provides *disambiguation pages* that present various possible meanings from which the intended article can be selected. The term *Library* (Figure 4.5) yields several options, including *Library*, a collection of books, and *Library (computer science)*, a collection of subprograms used to develop software. Section 5.1.2 describes how multiple matches can be disambiguated automatically.

Comparison of Wikipedia and Agrovoc shows that Wikipedia encodes many semantic relations between terms and it tends to cover terms that are more likely to represent topics. Given the sheer breadth and size of Wikipedia (and its rate of expansion), we assume that similar, if not better, coverage can be obtained for many other domains. Wikipedia is also a valuable machine-readable knowledge base of words and their meanings, concepts and their semantic relations. Section 5.2 describes how it can be mined for features useful for topic indexing.

4.2.2 Consistency of human indexing with Wikipedia

Professional indexers at FAO achieve an inter-indexer consistency of 37.8% when assigning Agrovoc terms to agricultural documents (Section 4.1.2). Assignment of topics from Wikipedia is not a conventionally performed task. Hence, it requires new experimental data. This section presents the results of a human study designed to create a collection of documents indexed by several people each.

The experiment was set up as an indexing competition to encourage high quality assignments. The test set included 20 technical research reports covering different aspects of computer science. Fifteen teams of graduate and undergraduate Computer Science students independently assigned topics to each report using Wikipedia article titles as the allowed vocabulary. Appendix C shows the instructions the students read before completing the task. Each team had two members who worked together in two 1½ hour sessions, striving to achieve high indexing consistency with the other teams; no collaboration was allowed. The best perform-

ing team, i.e. the one that was the most consistent with all other teams, received a prize.

In the following, the resulting data set is called WIKI-20. The topic sets are analyzed using the same criteria as in Section 4.1.2 and the results are compared to those achieved by professional indexers in the FAO-30 collection.

Exhaustiveness, specificity and idiosyncrasy

The teams were instructed to assign around 5 terms to each document. The topic sets ranged from 3 to 12 terms per document, with an average of 5.7. Unlike the professional indexing case described in Section 4.1.2, the exhaustiveness here did not vary much, except for one overly exhaustive team, which assigned 9.3 topics per document.

Around 31% of topics in WIKI-20 were single words, whereas the majority (53%) were two-words terms. 16% of terms contained three or more words. These statistics are similar to the CERN-290 corpus, whereas the topics in FAO-780 and FAO-30 were on average shorter. These findings indicate that computer science technical reports, like physics papers, require more specific index terms than agricultural documents.

In total, the students assigned 1722 Wikipedia topics to 20 documents. Out of these, 702 were unique and 389 idiosyncratic, i.e. assigned by just one team. Only 20 topics out of 702 (3%) were agreed by all teams, slightly less than in FAO-30, where 4% of topics were agreed on by all indexers.

Inter-indexing consistency

Each document received an average of 35.1 unique topics assigned by all teams. In FAO-780 professionals assigned 32.2 topics, but their individual topic sets were much larger (10.3 vs. 5.7). The students tend to assign more idiosyncratic topics than professionals, which indicates that their consistency as a group will be lower. However, the vocabulary used in FAO-30 is significantly smaller than Wikipedia (28,000 vs. two million concepts). The FAO indexers are more restricted in their choices and are expected to index more consistently.

Team rank	Native speaker?	Year	Consistency
1	yes	4	37.1
2	mixed	4	35.5
3	yes	4	33.8
4	mixed	3	32.4
5	yes	4	31.6
6	yes	3.5	31.6
7	yes	4	31.6
8	no	3	31.2
9	yes	3	31
10	mixed	3.5	30.8
11	yes	4	30.2
12	no	2.5	28.7
13	no	4	26.2
14	no	1	24.1
15	no	4.5	21.4
		Overall	30.5

Table 4.4 Indexing performance and characteristics for the human teams for WIKI-20

Table 4.4 shows the inter-indexer consistency of each team with the other 14 teams. The table also lists whether the team members are native English speakers, foreign students, or mixed, and gives their average study year. Consistency ranges from 21.1% to 37.1% with an average of 30.5% within this group. The values vary more than those computed for professional indexers in Section 4.1.2.

Factors affecting indexing performance

Knowing the language proficiency and the year of study of each team, we can investigate how these factors affect indexing consistency using the following subgroups:

Group A. 9 teams with senior students (at least in year 4), including non-native speakers;

Group B. 10 teams with at least one native speaker (*yes* or *mixed* in Table 4.4);

Group C. 7 teams with native speakers only (*yes* in Table 4.4);

Group D. 6 teams with senior students and at least one native speaker;

Group E. 6 randomly chosen teams.

Each group's consistency was re-computed using topics of the participating teams only.

The consistency of Group A was 31.2%, slightly better than that of the original 15 teams. The average consistency of Group B was significantly higher, 35%, and it did not improve as the teams were reduced to native speakers only in Group C. The best performing subgroup was D, with senior students and at least one native speaker per team. They achieved an indexing consistency of 38.3%, ranging from 35.2% to 43.9%. The performance of this subgroup is equivalent to that achieved by the 6 professional indexers in the FAO study. Consistency does not necessarily increase if the group is small: the improvement of Groups C and E over the original 15 teams was only 1.5%.

These experiments show that indexing quality in the first place depends on the indexers' fluency in the language of the documents and, in the second, on their familiarity with the area of the documents. Consistency does not seem to depend on the size of the controlled vocabulary. Native-speaking graduate students, who were choosing the topics from a vocabulary of millions terms, were as consistent with each other as professionals, who were choosing the topics from a much smaller thesaurus. This might have been due to their familiarity with Wikipedia and searching for topics in this resource, but also due to the competitive environment and team work.

4.2.3 Additional datasets

Indexing with terms from Wikipedia has been recently picked up by other researchers (Grineva *et al.*, 2009; Coursey *et al.*, 2009), who created their own data sets. Grineva *et al.*'s work was discussed in Section 3.4.2. They used a collection of 250 documents that include blog posts, news articles, research papers and websites. Their approach is shown to be stable to heterogeneous content and noise because it produces similarly good results on all documents in this set.

Two subsets of the full dataset have been made publicly available. The first contains 129 documents from different sources with on average 3 manually assigned topics from Wikipedia. The second consists of 86 blog posts with 7 topics per document. Five Computer Science graduates and undergraduates annotated the documents' topics, and only topics on which at least two annotators agreed were

included in the data sets. Section 7.3.3 uses these documents to test Maui's ability to handle heterogeneity and noise.

4.3 Tagging

Collaborative tagging is popular on websites hosting user-generated content because it adds structure to collections, thus making them more useful to other users. Most tagging schemes, however, rely on voluntary efforts. Although some might be experts in the area of focus, most are likely to be recreational web enthusiasts.

This section examines the quality of tagging on *CiteULike.org*, a platform for organizing academic citations. Co-tagging statistics and indexing consistency analysis are applied to extract a high-quality multiply indexed corpus from this data. The performance of CiteULike taggers is determined using methods traditionally used to assess professional indexing. This analysis provides insights into the quality of the tagging folksonomy and provides a guideline for developing an automatic tagging algorithm.

4.3.1 Extracting a multiply tagged corpus

CiteULike.org is a bookmarking service which concentrates on scholarly papers. Due to copyright laws, CiteULike does not replicate the content of tagged papers, but simply points to their URLs, where full text is not always freely accessible. When adding new citations, users are encouraged to assign tags. Automatic tag suggestions are not provided and users do not see other users' tags. Thus, there is no bias in tag assignments.

The CiteULike data set lists document IDs and tags assigned to them by individual users (identities are not provided). In the data snapshot used in this thesis,[7] 22,300 users have tagged 713,600 documents with 2.4M "tag assignments"— single applications of a tag by a user to a document. The two most popular tags, *bibtex-import* and *no-tag*, indicate an information source and a missing tag, respectively. Most other tags describe particular concepts relevant to the documents,

[7] Downloaded in June 2008 from *http://www.citeulike.org/faq/data.adp*

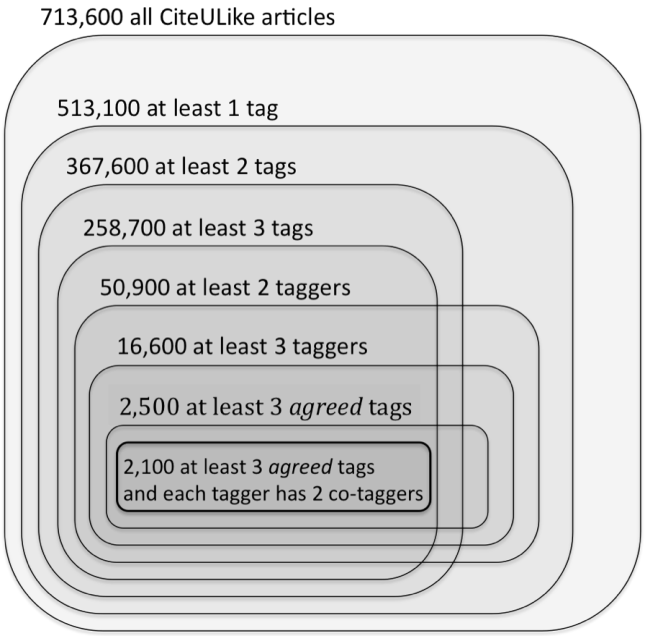


Figure 4.6 Imposing quality control on the CiteULike data

with rare exceptions (e.g. personal tags *to-read* or *todo*), which were removed. The data set does not contain any spam entries.

The experiments build on the assumption that while some taggers might assign poor tags others consciously produce meta-data of high quality. This is not unreasonable; CiteULike users supply tags voluntarily for their own benefit and are likely to assign tags of high quality. We identify such tags by looking at agreement between the users.

Co-tagging analysis

First, we identify a subset of sufficiently tagged citations—those that have been indexed with at least three tags on which at least two users have agreed. To measure tagging consistency, co-taggers are then identified. Two users are "co-taggers" if they have both tagged at least one common document. Furthermore, only taggers who have at least two co-taggers are included in the final data set.

Figure 4.6 shows the proportion of CiteULike documents that were discarded to produce a high quality data set. The final set contains only 2,100 documents (0.3% of the original). Because many documents are unavailable for download, e.g. book citations from *Amazon.com*, we restrict the attention to two sources, HighWire and Nature, which both provide easily-accessible PDFs of the full text.

Frequency	Tag	Frequency	Tag
124	network	39	transcription
120	evolution	35	protein
91	networks	35	human
58	expression	34	structure
51	bioinformatics	32	microarray
50	yeast	31	genomics
50	review	30	statistics
47	genome	30	gene expression
46	gene	28	systems biology
42	rna	26	regulation

Table 4.5 Top 20 tags assigned to documents in CiteULike-180

Multiply tagged data set

The above analysis results in a set of 180 documents indexed by 332 taggers, *CiteULike-180*, each document indexed with five tags on average. A total of 4,638 tags were assigned to documents in this set, but only 946 tags were agreed on by at least two users.

Table 4.5 shows the most popular tags in CiteULike-180. Most relate to the field of bioinformatics. To give an example, a document entitled *Initial sequencing and comparative analysis of the mouse genome* was tagged by eight users with a total of 22 tags. Four of them agreed on the tag *mouse* and one tagger used the broader term *rodents*. Three agreed on the tag *genome*, but one added the tag *genome paper*, and another used the more specific tag *comparative genomics*. There are also cases when tags are written together, e.g. *genomepaper*, or with a prefix *key genome*, or in a different grammatical form: *sequence* vs. *sequencing*. This example shows that many inconsistencies in tags are not caused by personalized tag choices as Chirita *et al.* (2007) suggest, but rather stem from lack of guidelines and uniform tag suggestions that a bookmarking service could provide. The Maui algorithm proposed in this thesis can be used for such suggestions.

4.3.2 Consistency of voluntary taggers

To compute the overall quality of tagging in the extracted subset of CiteULike, again the inter-indexer consistency measure described in Section 2.2 is applied. Although traditional consistency measures have not yet been applied to collaboratively tagged data, Xu *et al.* (2006) define an authority metric that assigns high

Tagger	Co-taggers	Documents	Consistency
1	1	5	71.4
2	1	5	71.4
3	6	5	57.9
4	6	6	51.0
5	11	12	50.4
6	2	5	50.1
7	4	6	48.3
8	8	8	47.1
9	13	16	45.4
10	12	8	44.4
11	7	6	43.5
12	7	6	41.7
13	8	5	40.9
14	7	6	39.7
15	9	13	38.8
16	4	5	38.4
17	12	9	37.3
18	4	14	36.1
19	9	8	35.9
20	10	11	33.7
21	7	6	33.1
22	6	5	33.0
23	7	10	32.1
24	11	16	31.7
25	8	13	30.6
26	6	8	30.6
27	9	6	29.8
28	10	12	29.0
29	8	6	28.8
30	9	10	27.9
31	10	8	26.7
32	8	7	26.3
33	10	5	25.6
34	8	7	21.0
35	9	9	18.3
36	3	6	7.9
average	7.5	8.1	37.7

Table 4.6 Consistency of the best taggers in CiteULike-180

scores to those users who match other users' choices on the same documents, in order to eliminate spammers.

Each of the 332 taggers in CiteULike-180 corpus indexed between 1 and 25 documents and has between 2 and 129 co-taggers, 18 on average. To compute the consistency of a given tagger, their tags are first compared to those of their co-taggers. The consistency is then averaged across documents.

Indexing consistency of professionals was similar among all six FAO's indexers (Section 4.1.2), whereas consistency of students varied depending on the language fluency and the study year (Section 4.2.2). In case of voluntary taggers, the distribution of per-user consistency resembles a power law with a few users achieving high consistency values and a long tail of inconsistent taggers.

The maximum consistency in this group is 92.3% and the average is 18.5%. The average consistency of the most prolific 70 indexers—those who have indexed at least five documents—lies in the same range, namely 18.4%.[8] This is significantly lower than the consistency of professionals on FAO-30 (38.7%) and of students on WIKI-20 (30.5%). Note that tagging is an example of *free* indexing, where no controlled vocabulary is used. Studies of consistency in free indexing report values between 4% and 67%, with an average of 27% depending on what aids are used (Markey, 1984).

Next, we identify the group of best taggers using two conditions:

1. taggers exhibit greater than average consistency with all others;
2. taggers are sufficiently prolific to have tagged at least five documents.

There are 36 such taggers; Table 4.6 lists their consistency within this group. The average consistency they achieve as a group is 37.7%, which is similar to consistency of professionals. Using inter-indexer consistency results we can determine a group of exceptionally well performing taggers from the large pool of CiteULike users. When assessing automatically assigned tags in Section 6.3, this data will help to identify the exact ranking of Maui's indexing performance.

4.4 Summary

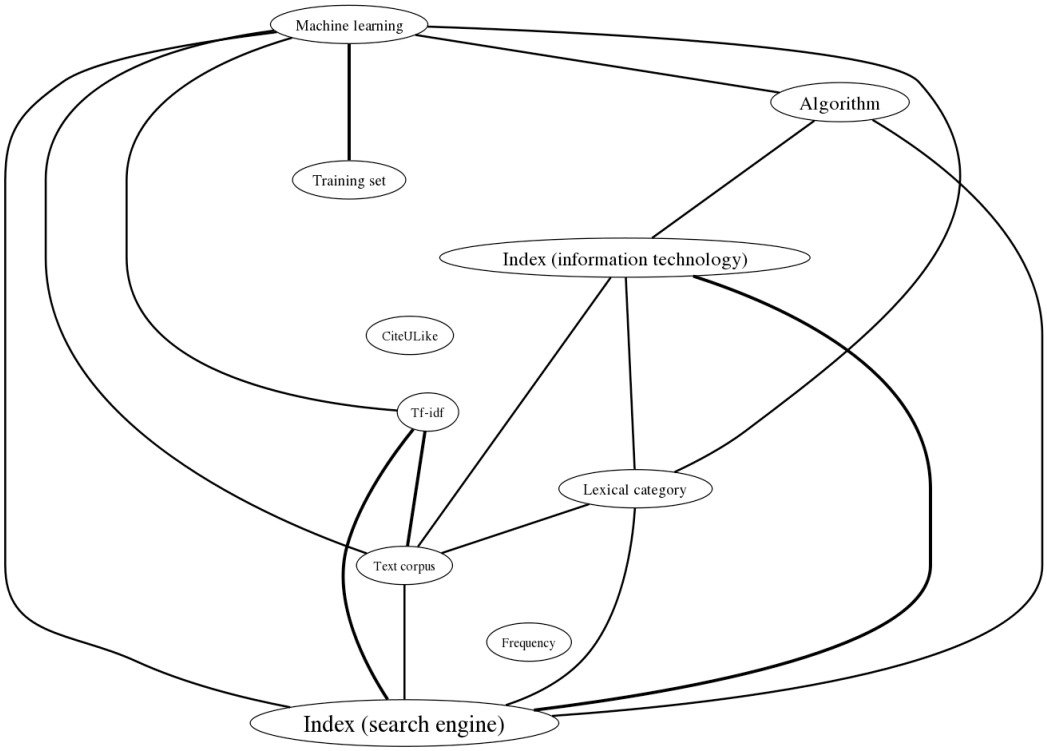
This chapter presented data sets three tasks: term assignment with a domain-specific thesaurus, keyphrase extraction with Wikipedia, and tagging. Each corresponds to a different way of indexing: controlled, somewhat restricted and free. Previous studies report that humans perform differently in free and controlled in-

[8] Interestingly, the average consistency of the proofreaders of this thesis, whose tags were shown as a tag cloud in Figure 2.3, Section 2.1.2, is 16.4%—only slightly worse than consistency achieved by CiteULike taggers.

dexing (Leininger, 2000). Experiments with multiply indexed collections presented in this chapter confirm this: the professional indexers produce topic sets of the highest quality; students perform slightly worse; and the consistency of taggers on the web is only half as good.

Comparison of exhaustiveness, specificity and indexing consistency between the collections identified several factors correlated with indexing quality: consistency in the size of assigned topic sets, language skills, familiarity with the area of the documents, indexing guidelines and the availability of a searchable vocabulary. Surprisingly, indexing experience and the size of the vocabulary are less important.

For the purposes of evaluating automatic tagging, a group of the best performing taggers can be extracted from collaborative tagging sites using co-tagging and indexing consistency analysis. The subset extracted from CiteULike is the largest multiply indexed collection in this thesis: 336 taggers and 180 documents. This set was created in a natural setting, as opposed to the other two data sets, which were results of controlled studies. All three data sets, FAO-30, WIKI-20 and CiteULike-180, help us to assess whether topics assigned automatically by Maui can be considered human-competitive.


Machine learning
Algorithm
Training set
Index (information technology)
CiteULike
Tf-idf
Lexical category
Text corpus
Frequency
Index (search engine)

Chapter 5
Candidate generation and filtering

Three ways of performing topic indexing were discussed in Chapter 4. *Term assignment* uses domain-specific thesauri as a source of topics. *Keyphrase extraction* traditionally uses phrases extracted from the document itself, but can be made more consistent by choosing topics with respect to Wikipedia. *Tagging* is the least restricted way of indexing; tags can be chosen freely.

This thesis proposes a two-stage topic indexing algorithm called Maui for all these tasks. In the first stage, *candidate generation*, Maui identifies candidate topics in a document by extracting its phrases and mapping them to vocabulary terms, if a vocabulary is available. In the second stage, *filtering*, it analyzes the properties of the candidate topics and discards the less significant ones.

Both stages have specific implementations depending on the task and what resources are accessible. Section 5.1 describes and evaluates Maui's candidate generation techniques proposed. Section 5.2 surveys features characterizing properties of typical topics and compares their performance on corpora from different tasks.

5.1 Candidate generation

This section describes the candidate generation methods and evaluates them by comparing against manually assigned topics using precision (P), recall (R) and F-measure (F), defined in Section 2.2. Recall is particularly important here, because it demonstrates how many manually assigned topics are automatically identified as

candidates. Precision is less important, because it will be further improved in the filtering stage.

5.1.1 Mapping documents to domain-specific thesauri

In term assignment, candidates are chosen by mapping document phrases to terms in a controlled vocabulary. All matching vocabulary terms serve as candidates. This indexing strategy imposes a limitation on the term assignment task: only phrases that appear in the actual text of the document are used to identify relevant vocabulary terms.

Previous studies have shown that approximately 80% of freely chosen keyphrases appear in the document text verbatim (Turney, 1999; Hulth, 2004). Such keyphrases are usually assigned by the author and are likely to match document phrases. An important question is how often topics assigned by a librarian and restricted by a vocabulary appear in a document. This section investigates this question on the data sets described in Section 4.1. The candidates are evaluated using topics assigned from domain-specific thesauri by professional human indexers.

Candidate generation steps

The input for Maui's candidate generation algorithm is the document text. The output is a set of candidates, i.e. vocabulary terms that match phrases appearing in this text. The algorithm operates in five steps:

1. Extract all *n*-grams up to a certain length, with should match the length of the longest term in the vocabulary.
2. Normalize both *n*-grams and vocabulary terms in order to ensure good coverage. Normalization techniques include conversion into lowercase, stopword removal, stemming and word re-ordering.
3. Search for vocabulary terms that match the extracted *n*-grams.
4. Apply semantic conflation: if the *n*-gram matches a non-descriptor, replace it by the linked descriptor.
5. Compute all possible senses of the given *n*-gram, where a *sense* is a vocabulary descriptor that represents the possible meaning of a phrase.

The last step implies that an *n*-gram may be ambiguous and match more than one descriptor. However, no disambiguation is performed during the mapping. Instead, it is performed implicitly in the filtering stage, which ranks candidates based on their properties. Ideally, candidates representing irrelevant senses should receive a low rank. This method is suitable for domain-specific thesauri, which contain few ambiguous terms. Alternatively, multiple matches could be disambiguated explicitly. A disambiguation algorithm would determine the context of the phrase, relate possible senses to this context, pick the best sense, and eliminate all others. This solution is used for mapping document phrases to terms in Wikipedia, as described in Section 5.1.2.

These steps produce a set of candidate topics—vocabulary terms that describe the content of the document. The mapping allows Maui to build a document's internal representation in terms defined for a given domain by the experts.

Evaluating the candidates

The candidate generation algorithm was tested on the three data sets discussed in Section 4.1: agricultural (FAO-780), medical (NLM-500) and physics (CERN-290) documents. These documents' phrases were mapped to terms in the corresponding domain-specific thesaurus: Agrovoc, Medical Subject Headings and High Energy Physics thesaurus.

The evaluation included several normalization configurations. Four stemming options were tested: no stemming, a simple *s*-removal stemmer that cuts off the *–s* and *–es* plural endings,[1] the Porter (1980) and the Lovins (1968) stemmer. With the additional alphabetic ordering of stems proposed by Paice and Black (2003) (Section 3.2.2), the four stemming options and two ordering options give eight possible configurations.

For each configuration and three data sets, Maui produced a set of candidate terms for each document. All candidates that match manually assigned topics, in the same normalized form, are considered correct and serve as positive examples.

[1] This is the first step in the Porter (1980) stemmer.

		FAO-780			NLM-500			CERN-290		
Ordering	Stemming	P	R	F	P	R	F	P	R	F
no	1. None	1.7	77.6	3.3	6.3	51.1	11.2	2.4	41.1	4.5
	2. *S*-removal	1.4	79.5	2.8	5.6	54.3	10.2	2.4	45.1	4.6
	3. Porter	1.1	82.2	2.2	4.4	56.8	8.2	2.1	48.6	4.0
	4. Lovins	0.9	81.5	1.8	3.4	57.4	6.4	1.4	47.9	2.7
yes	5. None	**1.7**	79.3	**3.3**	**6.4**	53.5	**11.4**	**2.6**	47.2	4.9
	6. *S*-removal	1.4	81.2	2.8	5.7	56.1	10.3	**2.6**	52.2	**5.0**
	7. Porter	1.1	**84.2**	2.2	4.5	58.7	8.4	2.3	**57.0**	4.4
	8. Lovins	0.9	83.7	1.8	3.5	**59.4**	3.5	1.6	56.1	3.1

Table 5.1 Candidate generation results for FAO-780, NLM-500 and CERN-290

	FAO-780		NLM-500		CERN-290	
Stemming	Candidates	Ambig.	Candidates	Ambig.	Candidates	Ambig.
None	520.5	4.9	130.1	7.4	145.7	4.3
S-removal	614.2	6.8	153.7	10.3	167.7	5.0
Porter	776.0	18.8	201.8	13.9	204.1	10.1
Lovins	932.8	25.8	261.5	17.2	279.1	16.6

Table 5.2 Number of candidate topics per document using different stemmers

All other candidates serve as negative examples.[2] The more positive candidates are identified, the higher the recall and the better the mapping technique.

Table 5.1 summarizes the results. Overall, alphabetic re-ordering improves recall by at least 2 percentage points in each configuration, without decreasing precision. The CERN-290 corpus shows the greatest improvement (up to 8.4 percentage points). The choice of stemmer also influences the results, Porter providing the greatest coverage of manually assigned terms in nearly all cases. In CERN-290 it improves recall by up to 9.8 percentage points (lines 5 and 7).

Using a stemmer generates more candidate terms, which in turn negatively influences precision. Table 5.2 compares the size of candidate sets when different stemmers are used. The first column for each corpus represents the average number of candidates per document; the second tells how many of them are ambiguous. Stemming improves recall by several percentage points (Table 5.1), but yields twice as many candidates per document (Table 5.2). Chapter 7 will investigate whether the filtering features are powerful enough to differentiate between posi-

[2] Because topic indexing is a subjective task, some negative candidates may be considered as positive according to some indexer, and vice versa.

tive and negative candidates, so that the size of the candidate set does not influence the quality of the final results.

The mapping algorithm results in sufficient coverage of manually assigned topics. In the FAO-780 corpus, 83.8% of topics are covered, similar to that reported in free keyphrase extraction (Turney, 1999; Hulth, 2004). The medical and physics domains are more challenging: only 58.9% and 56.6% of assigned topics are found in NLM-500 and CERN-290 respectively. Note that the recall achieved in the candidate generation stage cannot be improved in the filtering stage—it is the upper-bound recall performance for Maui—whereas ranking the candidates as discussed in Section 5.2 helps to identify the most significant topics, thereby improving precision.

5.1.2 Mapping documents to Wikipedia

Maui uses Wikipedia as a general vocabulary, which, like a domain-specific thesaurus, ensures consistency via redirect links that connect equivalent phrases to the same article. Due to Wikipedia's size and coverage, the main advantage of keyphrase extraction is preserved: Maui can be applied to any document collection, regardless of whether a domain-specific vocabulary is available.

Moving outside a specific domain and using a large vocabulary like Wikipedia presents significant challenges. There are over two million articles in the English Wikipedia, and a further several million alternative terms used to refer to these articles: redirect titles and anchor text used to link these articles within the text of Wikipedia. Almost every document phrase can be mapped to at least one article; most phrases map to several. A different mapping algorithm to the one described in Section 5.1.1 is necessary to prevent generating unneeded candidates and to disambiguate ambiguous matches.

Identifying important n-grams

Maui maps document phrases to Wikipedia article titles by first identifying important words and phrases, and then resolving these words and phrases to corresponding Wikipedia articles. The first step excludes words that contribute little to a document's content—that is, words that can be changed without affecting the

	Link frequency (L)	Total frequency (T)	Keyphraseness (L/T)
yacc	24	31	0.75
yacc is	0	0	0
yacc is well	0	0	0
well	269	304,303	0.0007
well established	1	1551	0.0006
…			
compiler-compiler	8	12	0.67

Table 5.3 Examples of keyphraseness values

meaning of the text. Mihalcea and Csomai (2007) propose the *keyphraseness* measure, which computes the probability of an *n*-gram appearing as a link in Wikipedia. The keyphraseness of an *n*-gram *a* is the number of Wikipedia articles in which this *n*-gram appears as a link (L), divided by the total number of articles in which it appears (T):

$$Keyphraseness(a) = \frac{L}{T}$$

Table 5.3 lists L, T, and keyphraseness values for *n*-grams in the sentence *Yacc is well established in the compiler-compiler field.* All phrases exceeding an empirically determined threshold are then selected, e.g. *yacc* and *compiler-compiler* for *keyphraseness* ≥ 0.01 in the example sentence.

Disambiguating the n-grams' meanings

Next, Maui links each selected phrase to a Wikipedia article that captures its meaning. For example, the word *tree* in a document about depth-first search should be linked to the article *Tree (Data structure)* rather than a biological tree or a part of a saddle.

Mihalcea and Csomai (2007) retrieve possible senses for an *n*-gram from link annotations in Wikipedia. If a phrase *bar* appears in links [[*bar (law)*|*bar*]] and [[*bar (establishment)*|*bar*]], the two Wikipedia articles *Bar (law)* and *Bar (establishment)* are possible senses. However, links are often made to hyponyms rather than synonyms of the anchor text. For example, the anchor *king* has 371 destinations, the majority of which are specific kings. Direct matching of *n*-grams against titles of Wikipedia articles and redirects avoids such irrelevant senses.

Before matching *n*-grams to Wikipedia titles, Maui case-folds both of them and strips parenthetical text from the titles (e.g., *Bar (establishment)* becomes *Bar*). If the *n*-gram matches an article, it is used as a sense. If it matches a redirect, the target article is used as a sense. If it matches a disambiguation page, all senses listed on that page are collected. The result is a set of possible senses for each significant document phrase.

If more than one article relates to a given *n*-gram, the algorithm needs to determine the intended sense. The best performing method proposed by Mihalcea and Csomai is a supervised approach that learns typical features of ambiguous mappings such as part-of-speech patterns and their context words. Maui implements an unsupervised disambiguation technique based on two properties of the mappings: the *commonness* of a sense and its *semantic relatedness* to the context. The context is a set of articles that result from mapping unambiguous phrases that appear in the same document. By default, Maui requires five such articles. If less than five are found, Wikipedia articles that map from ambiguous *n*-grams with high probability (see *commonness*, below) are used as context as well.

Given an ambiguous *n*-gram, Maui computes the average semantic relatedness of each possible sense to all context articles identified in a given document. The semantic relatedness of a pair of articles is computed from their incoming links (Milne and Witten, 2008). For each pair of articles x and y, the sets of articles X and Y that link to them are retrieved and their overlap ($X \cap Y$) is computed. Given N, the total number of articles in Wikipedia, the relatedness of x and y is:

$$relatedness_{x,y} = \frac{\log\left(\max\left(|X|,|Y|\right)\right) - \log\left(|X \cap Y|\right)}{\log\left(N\right) - \log\left(\min\left(|X|,|Y|\right)\right)}.$$

The disambiguation technique takes into account both the relatedness to context and the *commonness* of each sense: the extent to which they are well-known. The commonness of a sense T for an *n*-gram a is defined as:

$$commonness_{a,T} = P\left(T|a\right).$$

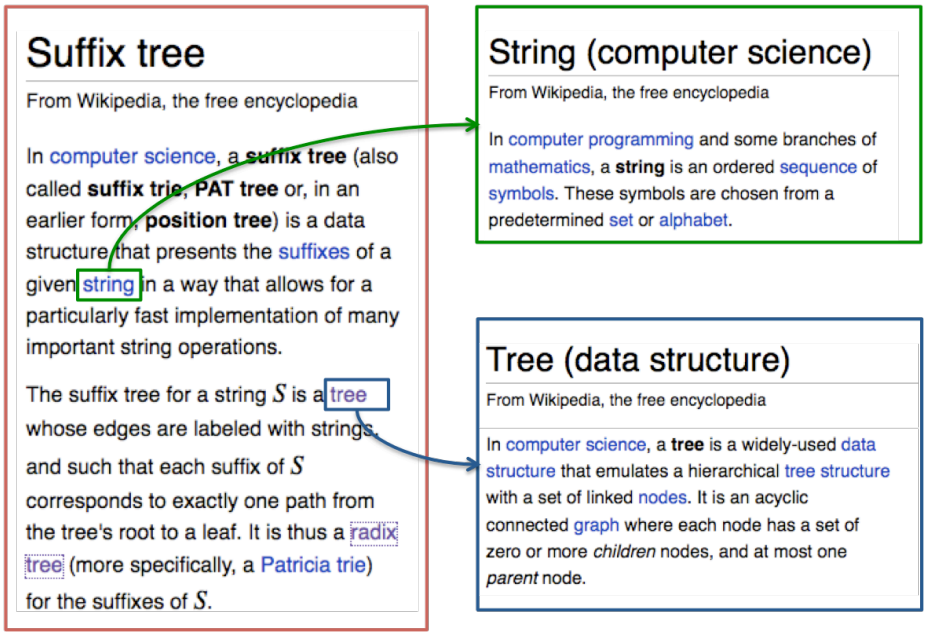


Figure 5.1 Link annotations in Wikipedia articles

For example, the word *Jaguar* appears as a link anchor in Wikipedia 927 times. In 466 cases it links to the article *Jaguar cars*; thus the commonness of this mapping is 0.5. In 203 cases it links to the description of *Jaguar* as an animal, a commonness of 0.22. Mihalcea and Csomai (2007) use this information for one of their baselines.

The score of a mapping of an n-gram a to an article T is the average relatedness of T to the context articles ($c \in C$), multiplied by its commonness given the n-gram a:

$$score(a,T) = \frac{\sum_{c \in C} SIM_{T,c}}{|C|} \times commonness_{a,T}$$

The highest-scoring article is chosen as the final mapping for the n-gram a.

Evaluating the disambiguation

The goal of the mapping is to express a document using the terminology of a given vocabulary. In traditional term assignment, human indexers perform mapping implicitly, in their mind, before choosing the most significant terms as topics (David *et al.*, 1995). In Wikipedia, human contributors explicitly identify all terms that are relevant for understanding a given article and link these to corres-

	Attempted	Correct	P (%)	R (%)	F (%)
Random	17,640	8,651	45.8	45.7	45.8
Most common	17,640	15,886	90.6	90.4	90.5
Relatedness-based	17,416	16,220	93.3	92.3	92.9

Table 5.4 Disambiguation results on 100 Wikipedia articles

	any team	at least two teams	at least three teams
Topics per doc	35.5	15.8	8.7
Recall (%)	53.3	65.6	70.5

Table 5.5 Candidate generation results on WIKI-20

ponding Wikipedia articles. Figure 5.1 shows how the author of the article *Suffix tree* linked the words *string* and *tree* to the articles *String (Computer Science)* and *Tree (Data structure)*. The ideal mapping algorithm should be able to match such annotations.

The method was tested on 100 randomly chosen Wikipedia articles, and Table 5.4 compares the results with two baselines. Given a document phrase and a set of possible senses, the first baseline chooses a sense at random. The second chooses the sense whose commonness value is greatest. The new relatedness-based disambiguation method covers almost as many candidates as the baselines (17,416 vs. 17,640).[3] The comparison also shows that this method is substantially more accurate than both baselines, achieving an F-measure of nearly 93%. The results are somewhat better than the ones reported in Mihalcea and Csomai (2007).

Evaluating the candidates

Section 4.2.2 presented the WIKI-20 corpus, a collection of documents with topics from Wikipedia assigned by 15 teams of Computer Science students. For each document the above method produced 160 candidate topics on average. Note that in term assignment (Section 5.1.1), many more candidates per document were generated—up to 932 terms for the FAO-780 corpus (see Table 5.2).

Table 5.5 summarizes how well candidates extracted from Wikipedia with the above method cover manually assigned topics in this experiment. Out of 35.5

[3] Because each candidate in the baseline settings originates from an anchor, but does not necessarily match the title of an article or a redirect, the baseline's coverage is slightly higher.

terms assigned by all 15 teams, only 53.3% appear in the candidate set (column 1). However, the percentage of correctly identified candidates increases to 65.6% and 70.5% respectively if only terms assigned by at least two and three out of the 15 teams are considered. The results show a general trend: the more teams agree on a topic, the more likely it will be covered.

Section 4.2.3 described two further collections with topics from Wikipedia. Each document was indexed with an average of 7.5 and 3 topics in each collection respectively. Maui successfully identified 74.1% of these topics.

5.1.3 Generating candidates in automatic tagging

Any automatic keyphrase extraction method discussed in Section 3.2 can be applied to generate tag suggestions. Surprisingly, there has been little research in this area. Related work on automatic tagging (Section 3.3) shows that researchers focus on personalized elements when generating tags. They either compute tags that a given user, or his co-taggers, assigned to similar documents (Mishne, 2007; Sood *et al.*, 2007), or compute significant phrases from similar documents owned by the same user (Chirita *et al.*, 2007). This thesis proposes an alternative approach based on supervised keyphrase extraction. This section describes how Maui generates candidate tags and evaluates their quality.

Extracting the candidates

Because generation candidate tags from documents is equivalent to the first stage of the keyphrase extraction algorithms described in Section 3.2.1, Maui simply adopts the technique used in the Kea algorithm (Witten *et al.*, 1999). The maximum length of a candidate phrase is the longest observed length of a tag in the training data. Tags are usually short. The CiteULike-180 corpus described in Section 4.3.1 only contains tags with three words or less, and most are single words.

Evaluating the candidates

Candidate tags generated for documents in CiteULike-180 were matched against the tag sets assigned by all taggers to these documents. Any candidate that appeared in at least one tag set was considered to be correct. Recall is computed as the percentage of automatically identified unique tags assigned by the users.

Order	Stemming	Column 1 length = 1 word P	R	F	Column 2 length ≤ 2 words P	R	F	Column 3 length ≤ 3 words P	R	F
no	1. None	0.8	60.7	1.6	0.6	68.0	1.2	0.4	68.4	0.8
	2. S-removal	0.8	61.5	1.6	0.6	69.2	1.2	0.4	69.8	0.8
	3. Porter	1.0	63.0	2.0	0.6	71.1	1.2	0.4	71.7	0.8
	4. Lovins	**1.0**	63.5	**2.0**	0.6	71.8	1.2	0.4	72.3	0.8
yes	5. None	0.8	60.7	1.6	0.6	68.2	1.2	0.4	68.8	0.8
	6. S-removal	0.8	61.5	1.6	0.6	69.4	1.2	0.6	69.4	1.2
	7. Porter	**1.0**	63.0	2.0	0.6	71.5	1.2	0.5	72.4	1.0
	8. Lovins	**1.0**	63.5	**2.0**	0.7	72.2	1.4	0.5	**73.0**	1.0

Table 5.6 Candidate generation results on CiteULike-180

Table 5.6 summarizes the results and compares the effects of different normalization techniques, discussed in Section 5.1.1, as well as the effect of the term length threshold. A more powerful normalization technique can increase recall by over 4 percentage points, from 68% to 72.2% (first column, rows 1 and 8). However, the effect of re-ordering the stems and increasing the term length threshold is marginal, because the vast majority of tags are single words or two-word phrases.

The number of candidates per document extracted in each configuration varies from 1150 (length = 1 word, Lovins stemmer, ordering) to 3500 (length ≤ 3 words, no stemming, no ordering). Rows 7 and 8 in column 1 of Table 5.6 show recall values on approximately 2000 candidates. Note that in term assignment, powerful stemming increases the number of candidates (Table 5.2), but the opposite happens here. If phrases are stemmed in term assignment, they can be mapped to more vocabulary terms. If phrases are stemmed in automatic tagging, they are conflated to a smaller number of candidates.

With the most powerful stemmer (Lovins) and a phrase length of three words, 73% of manually assigned tags were found in the documents (Row 8 and column 3 in Table 5.6). This shows that document phrases influence the users' tag choices and are a good source of tag suggestions.

5.1.4 Coverage of manually assigned topics

The effectiveness of candidate generation methods depends on how well the resulting candidate sets cover manually assigned topics. In term assignment, depend-

ing on the thesaurus, Maui identifies between 60% and 80% of manually assigned topics. In topic indexing with Wikipedia, mapping document phrases to Wikipedia article titles covers 53% of all manually assigned topics and 70% of those agreed by at least three teams. Finally, in automatic tagging, Maui detects 73% of manually assigned tags among document phrases.

In each task, the recall values represent the upper bound recall for the final results. Precision will be improved when the candidate sets are condensed to the most prominent topics for each document. This process is referred to as "filtering".

5.2 Filtering

Document topics can be obtained by ranking candidates according to their properties, or *features*. Features are chosen manually by observing characteristics of topics assigned by humans. Ideally, a feature should produce substantially different scores for positive and negative candidates. In the real world such situations are rare. One feature is usually insufficient to differentiate the candidates, and several must be combined to obtain best results. The final combination of features is either a fixed formula determined empirically using a sample corpus, or a flexible model created using a learned classifier that automatically adjusts the importance of each feature depending on the characteristics of the training data.

This section surveys features that are useful in topic indexing and assesses their performance on the three test collections FAO-780, WIKI-20 and CiteULike-180 (Section 4.1). The features are assessed in the following way. First, candidate topics are generated as described in Section 5.1. Next, feature values are computed individually for positive and negative candidates. The percentage of candidates for each value in each class is then plotted to demonstrate the strength of the feature. For space reasons, only plots for some document sets are shown, but the distributions of feature values look similar for other collections as well. Section 5.2.6 additionally compares the features using the area under the ROC curve statistic, which provides a more formal and less empirical assessment than the plots.

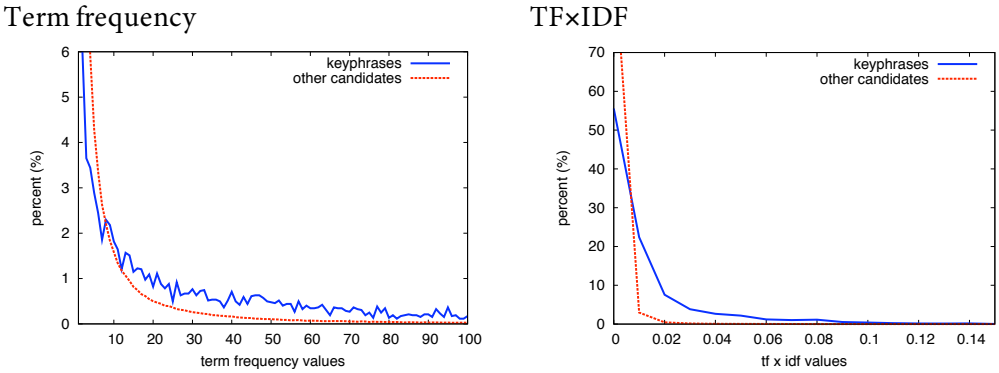


Figure 5.2 Distribution of term frequency and TF×IDF in FAO-780

5.2.1 Frequency statistics

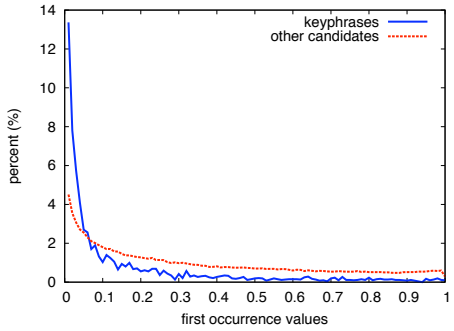
An intuitive way of determining a document's topics is by identifying candidate topics with the highest ***term frequency***—those that appear most frequently. However, term frequency might not be discriminative enough, because some terms are generally common and used repeatedly in most documents in the collection, like *Fishing* or *Agriculture* in FAO-780. To highlight candidates that are particularly frequent in a given document, ***inverse document frequency*** (IDF) is used. IDF computes the proportion of documents containing a given term in a reference corpus. The ***TF×IDF*** statistic combines term frequency and inverse document frequency in a single formula. Given a candidate term t in a document d, *TF×IDF* computes the following:

$$TF \times IDF = \frac{freq(t,d)}{\sum_i freq(t_i,d)} \times -\log_2 \frac{n_t}{N}$$

where $freq(t,d)$ is t's occurrence count in d, n_t is the number of documents containing term t and N is the total number of documents in the corpus. The first component in this expression is the term frequency, or the normalized frequency of term t in document d. The second is the negative logarithm of the inverse document frequency, which is larger for rarer phrases.

Figure 5.2 plots the distribution of the term frequency and TF×IDF values in the FAO-780 corpus. Positive examples are denoted by a solid line and negative ones by a dashed line. The values of negative examples are lower than that of positive in

First occurrence



Last occurrence

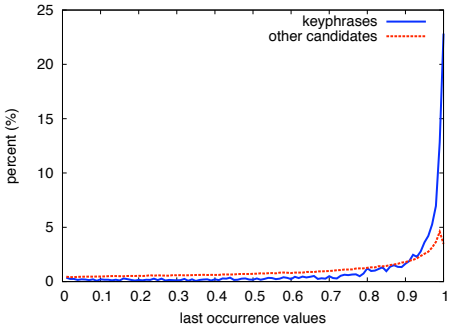
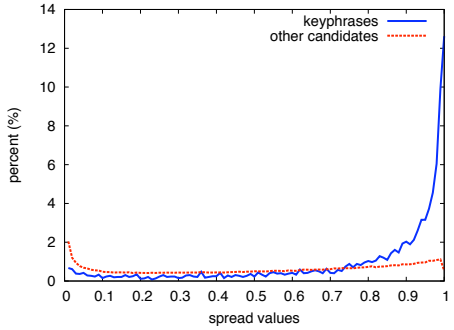


Figure 5.3 Distribution of *first* and *last occurrence* values in FAO-780

FAO-780



CiteULike-180

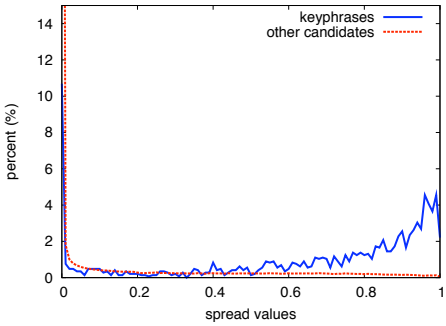


Figure 5.4 Distribution of occurrence spread in FAO-780 and CiteULike-180

both cases. However, the TF×IDF feature produces a larger gap between the two curves and is thus more informative than the term frequency.

TF×IDF has been applied in keyphrase extraction (Witten *et al.*, 1999) and automatic tagging (Brooks and Montanez, 2006). Alternatively, term frequency and inverse document frequency have been used as two independent features (Hulth, 2004, Csomai and Mihalcea, 2008). Document frequencies can be computed using the domain-specific training set or a general reference corpus like Wikipedia (Csomai and Mihalcea, 2008).

5.2.2 Occurrence positions

Some document parts are more important than others. Professional human indexers commonly focus on the opening—title, abstract, table of contents, introduction—or final sections—conclusion and reference list (David *et al.*, 1995). If such locations are marked in a document, they can be included into the indexing algorithm as one of the features. If not, Nguyen and Kan (2007) suggest

algorithm as one of the features. If not, Nguyen and Kan (2007) suggest classifying document's paragraphs into structural categories automatically.

A more generalizable approach, which is also adopted in this thesis, is to determine occurrence positions statistically. Witten *et al.* (1999) and Hulth (2004) use the ***position of the first occurrence*** feature, calculated as the candidate's distance in words from the beginning of the document, normalized by the document's word count. The result is the proportion of the document that precedes the candidate's first appearance. Candidates that have extreme (high or low) values for this feature are more likely to be valid topics. Figure 5.3 shows the distribution of feature values for both the first and the last position of the occurrences in FAO-780.

A variation of this feature is the ***occurrence spread***, calculated as the distance between the first and the last occurrences of a term in a document and normalized by the document's length in words. Terms that are mentioned both at the beginning and end of a document are characterized by high values. Figure 5.4 shows that a large percentage of topics have an occurrence spread close to 1, particularly in the FAO-780 corpus.

The curves in Figures 5.3 and 5.4 look somewhat similar. The different occurrence features are likely to repeat the same information rather than complement each other. Evaluation of the indexing performance using combinations of these features will identify possible redundancies.

5.2.3 Keyphraseness

Analysis of manually assigned keyphrases shows that many of them repeat because human indexers seem to prefer certain vocabulary terms (Sections 4.1.3). Bates (1996) writes that the human mind processes certain classes of terms differently from others, which explains the privileged use of some terms in comparison to others. Statistically, the likelihood of a candidate being a topic increases with the number of times it previously appeared in manually assigned topics sets.

Deriving keyphraseness from the training data

Frank *et al.* (1999) define the ***domain keyphraseness*** feature, which records the number of times a candidate appears in the training set as a keyphrase. This feature

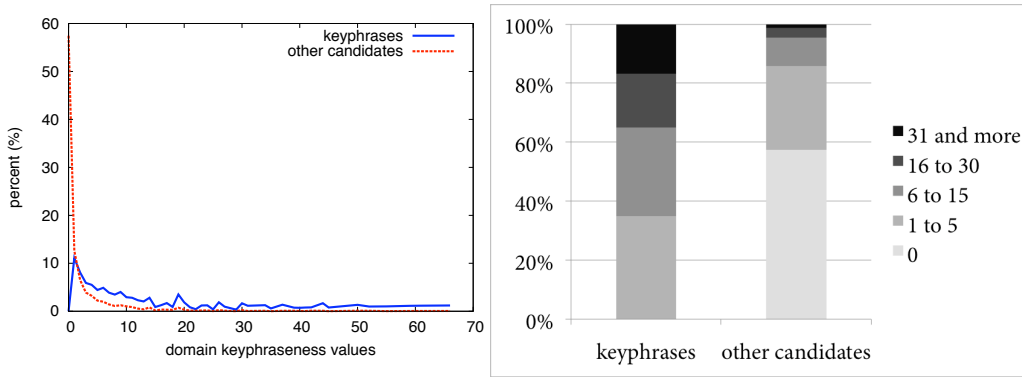


Figure 5.5 Distribution of domain keyphraseness in FAO-780

improves indexing performance on domain-specific documents and adds consistency to the extracted keyphrase sets. Automatic tagging methods utilize similar statistics: Mishne (2006) and Sood *et al.* (2007) automatically suggest tags previously assigned to similar documents. In Frank *et al.* the keyphraseness feature is just one component of the overall model. If a candidate never appeared as a keyphrase in the training corpus, it can still be assigned automatically, as long as its other feature values are significant enough. On the contrary, the algorithms in Mishne and Sood *et al.* never suggest unseen tags.

Figure 5.5 compares the distribution of the domain keyphraseness feature for positive and negative candidates in the FAO-780 corpus. Keyphraseness appears to be a strong feature, which is more obvious in the column histogram on the right, in which values are first manually grouped into five numeric ranges. In machine learning, the process of grouping feature values into more meaningful groups than individual numeric values is performed automatically using discretization (Section 6.4.1).

Deriving keyphraseness from Wikipedia

Wikipedia can be seen as a large collection of documents manually annotated with topics, because each article is hyperlinked to other articles that describe related topics. The anchors in the hyperlinks are similar to freely chosen keyphrases, and the hyperlinked articles are similar to index terms. For example, the description of *Library* in Wikipedia is annotated with the following anchors and articles:

- *books* → *Books*,
- *media* → *Data storage device*,

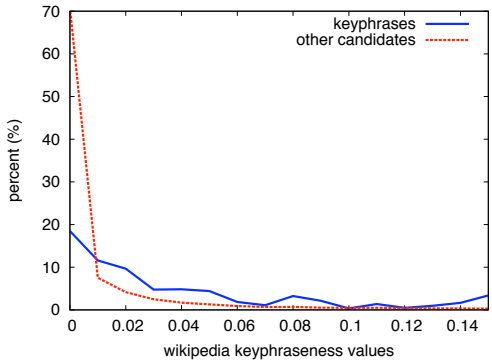


Figure 5.6 Distribution of Wikipedia keyphraseness in CiteULike-180

- *storing information* → *Recording*,
- *maps* → *Maps*,
- *documents* → *Documents*,
- *information* → *Information*
- *librarians* → *Librarian*.

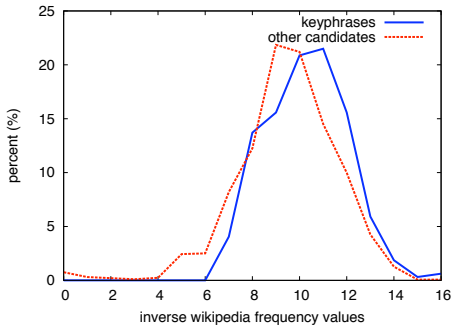
Statistics of such occurrences can be used as features in topic indexing because they represent semantic properties of phrases and concepts: Some phrases and concepts are more likely to be used in explanations. We can assume that these phrases and concepts are highly descriptive and thus likely to denote topics.

Wikipedia keyphraseness is the likelihood of a phrase *a* to appear as an anchor anywhere in the Wikipedia corpus.[4] It has been defined in Section 5.1.2 as *Keyphraseness(a)*, used for identifying meaningful phrases in document text. Csomai and Mihalcea (2008) show that this is one of the strongest features for the automatic construction of a back-of-the-book index. Figure 5.6 plots Wikipedia keyphraseness values for candidate *n*-grams in the CiteULike-180 corpus. Positive candidates assigned by human taggers exhibit much higher values than other candidates.

Instead of computing the keyphraseness of a phrase (e.g. *storing information)*, we can also compute the keyphraseness of a Wikipedia article (e.g. *Recording*). The keyphraseness of an article is its likelihood of being used in explanations in other articles. In other words, given an article *A*, the keyphraseness is the number of in-

[4] This feature can be generalized to the entire web, by computing the likelihood of a phrase appearing as an anchor anywhere on the web. An experiment applying this feature to indexing would be useful, but the scale and resource requirements are too demanding for this thesis.

Inverse Wikipedia frequency



Total Wikipedia keyphraseness

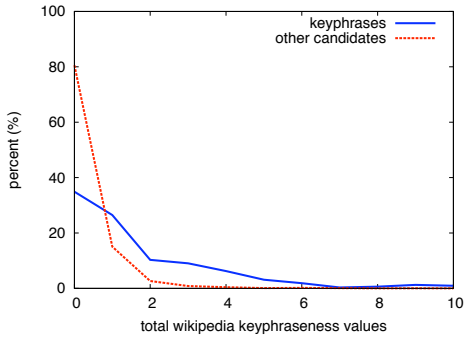


Figure 5.7 Distribution of inverse Wikipedia frequency and total Wikipedia keyphraseness in WIKI-20

coming links to this article *inLinksTo*(A) divided by the total number of articles in Wikipedia N. The result can be normalized in a manner similar to the inverse document frequency (Section 5.2.1) and expressed as the ***inverse Wikipedia frequency***:

$$IWF(A) = -\log_2 \frac{inLinksTo(A)}{N}$$

Based on its distribution of values in Figure 5.7 (left), this feature does not clearly differentiate between positive and negative topics in the WIKI-20 corpus.

The keyphraseness of a vocabulary term (as opposite to the keyphraseness of a phrase) can also be computed as the sum of the keyphraseness of all phrases that were mapped to this term in a document. Given candidate topic A in document d, A's ***total Wikipedia keyphraseness*** TWK is defined as

$$TWK(A_d) = \sum_{p \Rightarrow A} WK(p) \times freq(p,d)$$

where for each unique n-gram p mapped to candidate A the TWK value is increased by p's Wikipedia keyphraseness value (WK) times p's frequency in document d. Figure 5.6 (right) plots TWK's values in the WIKI-20 corpus. The distribution is similar to that in Figure 5.6, but the difference between positive and negative candidates is more obvious.

Four different but related features were presented in this section: domain keyphraseness, Wikipedia keyphraseness, inverse Wikipedia frequency and total Wikipedia keyphraseness. Domain keyphraseness can be applied in any topic in-

dexing task provided a training corpus of manually indexed documents is given. The other three features require Wikipedia as a reference. Wikipedia keyphraseness is a constant value that can be determined for any string. The inverse Wikipedia frequency is also a constant value, but if candidate topics are not Wikipedia articles, it can only be determined for those phrases that can be mapped to such articles. The total Wikipedia keyphraseness is document dependent: it combines Wikipedia keyphraseness and term frequency.

5.2.4 Semantic relatedness

When explaining a topic, authors naturally use terms that relate to each other semantically. The earlier example, the *Library* article, contains terms like *books*, *documents,* and *information*, which are all highly related. Therefore, semantic relatedness of terms can be useful in topic indexing. There are two basic approaches to determine semantic relatedness automatically: statistical and symbolic. Statistical approaches compute co-occurrence statistics. Symbolic approaches analyze the connectivity of terms in manually encoded knowledge structures.

Co-occurrence analysis requires a large corpus. Turney (2003) collects co-occurrence frequencies from the entire web by querying a search engine. He reports improved results, but admits that his method is computationally expensive and impractical. Sigurbjörnsson and Overell (2008) mine co-occurrences from thousands of tag sets manually assigned to photographs on *flickr.com*. Given a tag for a photograph, the co-occurrence statistics are used to generate related tags. This method can only be applied to previously assigned tags.

Two of the topic indexing tasks investigated in this thesis assign topics from manually defined knowledge structures. Term assignment uses a domain-specific thesaurus, and the new approach to keyphrase extraction uses Wikipedia. Both resources encode semantic relatedness, which can be retrieved automatically.

A simple semantic feature is the ***node degree***, which measures how richly the term is connected in the thesaurus graph structure. The degree of a term is the number of semantic links that connect it to other terms—for example, a term with one broader term and four related terms has degree 5. The node degree fea-

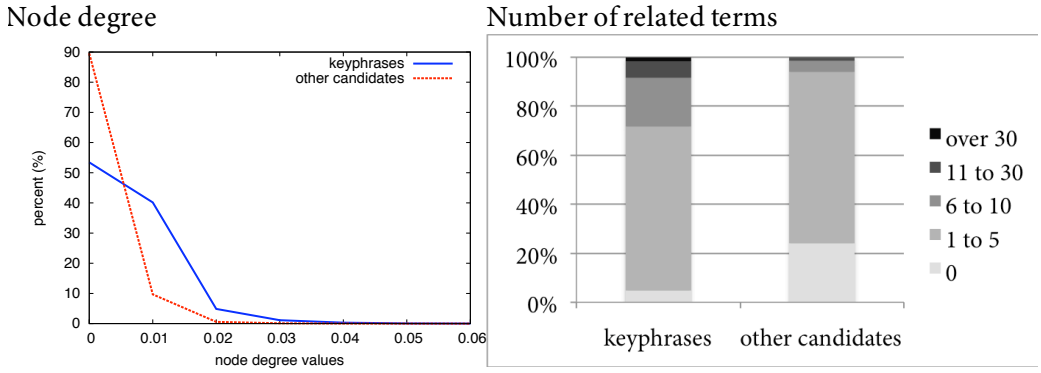


Figure 5.8 Distribution of node degree, and the number of related terms in FAO-780

ture represents the number of links that connect the term to other candidates identified in the same document, normalized by the total number of candidates. A document that describes a particular topic area will cover many terms from this area; therefore candidates with a high node degree are more likely to be significant. When Wikipedia is used as a controlled vocabulary and as a knowledge base, the node degree for a given article can be computed by counting the number of incoming links from other candidate articles.

Figure 5.8 plots the node degree values and the number of related terms for all candidates in the FAO-780 corpus. Positive candidates exhibit much higher values than negative ones. In the histograms, only 4.8% of keyphrases are not related to any other terms in the document, whereas this is true for nearly a quarter of non-keyphrases (24.1%).

Recent research on mining semantic meaning from Wikipedia developed more accurate methods for computing semantic relatedness than mere link counts. Gabrilovich and Markovitch (2007) apply explicit semantic analysis (ESA), to compute vector similarity between two Wikipedia articles. Milne and Witten (2008) propose a link-based approach that is more efficient than ESA and nearly as accurate. Section 5.1.2 contains a detailed description of this method, which is also applied in Maui for mapping documents phrases to Wikipedia articles. Given a candidate, its ***semantic relatedness*** feature value is the average semantic relatedness to all other candidates in this document. The higher the value is, the more likely this candidate is a significant topic.

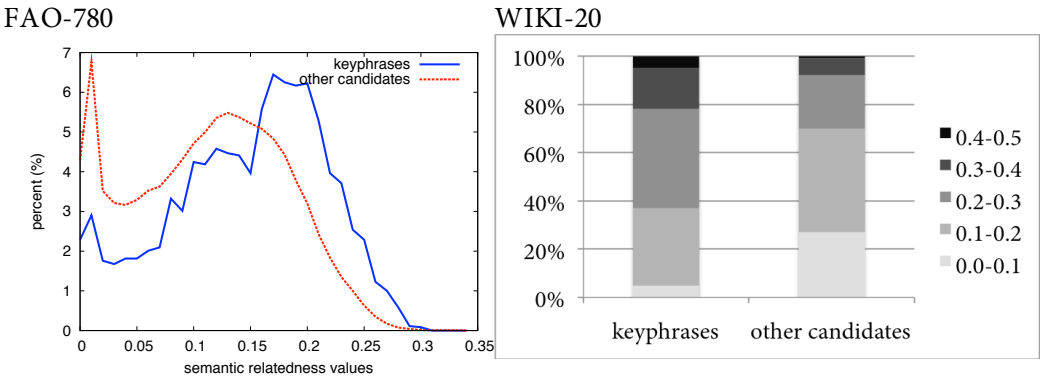


Figure 5.9 Distribution of semantic relatedness in FAO-780 and WIKI-20

In term assignment and automatic tagging, candidates are thesaurus terms and phrases, but they can be mapped to the most likely Wikipedia articles using the commonness measure described in Section 5.1.2. Then the semantic relatedness feature can be then computed in the same manner. Figure 5.9 shows the distribution of semantic relatedness values computed on the WIKI-20 and CiteULike-180 collections.

There are other, more complex, ways to compute semantic relatedness, which are not considered in this thesis. Latent semantic analysis (LSA) calculates term relatedness by subsequent decomposition of term-document matrixes (Landauer *et al.*, 1998). Csomai and Mihalcea (2008) apply LSA to determine term relatedness in the back-of-the-book indexing. Other methods explore the hierarchical structure in the lexical database WordNet (Fellbaum, 1998) and the category tree of Wikipedia (Strube and Ponzetto, 2006). This thesis does not use these methods because they are restrictive: LSA requires large reference corpora and WordNet is only available in a few languages.

5.2.5 Other features

Term length in words is a feature that highlights candidates of a typical length in the training corpus. Longer phrases are usually more specific (Section 4.1.3) and term length is an indicator of the indexing specificity in a given training corpus. The same kind of information can be determined using the hierarchy in a vocabulary. The higher up a concept, the more general it usually is. This can be captured with the ***generality*** feature. Wikipedia Miner (Milne, 2009) retrieves the gener-

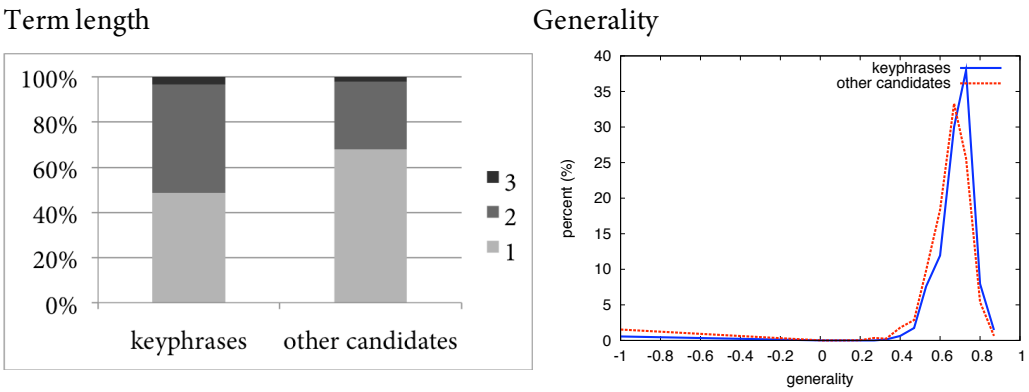


Figure 5.10 Distribution of term length and generality in FAO-780

ality of a Wikipedia article by analyzing its position in the category tree. Figure 5.10 shows that both term length and generality are less informative than other features in the FAO-780 corpus. Other collections exhibit similar statistics (Section 5.2.6).

Several other features for topic indexing have been suggested in the literature (Section 3.2.1). Among the best performing ones is the ***part-of-speech pattern*** feature. A part-of-speech tagging algorithm assigns a grammatical category to each word in a document. The patterns for each candidate are then collected. For example, a phrase *topic indexing* would receive a pattern *NN NN* (two consecutive singular nouns). Hulth (2004) and Nguyen and Kan (2007) use these patterns as nominal feature values in their algorithms. Csomai and Mihalcea (2008) convert each pattern into a numeric value that represents its likelihood of denoting a positive candidate. This feature is the strongest out of 14 features they suggest for the back-of-the-book indexing task.

The part-of-speech pattern feature was not applied in this thesis for two reasons. First, it would require an additional language-dependent tool, a part-of-speech tagger. Instead, this thesis proposes an algorithm that operates with a minimum of additional resources. Second, grammatical correctness of candidates—the main purpose of the part-of-speech pattern feature—is not an issue in term assignment and indexing with Wikipedia, where terms are defined in a controlled vocabulary.

Recent supervised approaches experiment with further features. Csomai and Mihalcea (2008) simulate ***short term memory*** effects motivated by cognitive pro-

cesses that take place when humans perform indexing. They extract noun phrases and apply PageRank and LSA to determine their re-occurrence, given a sequence of sentences. Re-occurrences are scored using three different weighting methods and added as features to the algorithm. These features all outperform TF×IDF weighted *n*-grams in the back-of-the-book indexing task. Adding these features to the algorithm proposed in this thesis would introduce complexity and language dependence that we try to avoid.

Nguyen and Kan (2007) experiment with ***named entity*** and ***acronym*** features. However, the individual contributions of these features are not visible from their experiments, and they also require additional language-dependent tools.

5.2.6 Feature comparison

All features presented in this section have a common characteristic: they generate numeric values. A machine learning algorithm examines these values in training data and obtains a model that separates the positive and negative candidate topics. The performance of the resulting classifier depends not only on the features but also on the threshold.

In order to assess the features independently of the threshold, performance can be calculated for every possible threshold. The algorithm scans through all possible thresholds and computes two sets of values: true positive rate (the percentage of manually assigned topics that were identified by the algorithm) and false positive rate (the percentage of topics that were not assigned manually, but classified as such by the algorithm). A graph plotting these for all possible thresholds forms the *receiver operating characteristics curve* or ROC curve (Witten and Frank, 2005). The *area under the ROC curve* (abbreviated *AUC*) is a single number that expresses the performance of the feature that created the given ROC curve. The AUC is the probability that a randomly chosen topic is ranked above a randomly chosen non-topic. AUC values close to 1 indicate perfect performance, whereas 0.50 indicates random performance.

By calculating the AUC for each feature individually we can fairly compare the ability of each feature to distinguish between topics and non-topics. Table 5.7

	Feature	FAO-780	WIKI-20	CiteULike-180
Frequency	Term frequency	0.841	0.753	0.918
	Inverse document frequency	0.510	0.446	0.504
	TF×IDF	0.866	0.764	0.887
Occurrence position	First occurrence	0.785	0.758	0.830
	Last occurrence	0.748	0.612	0.762
	Spread	0.807	0.765	0.891
Keyphraseness	Domain keyphraseness	0.879	0.929	0.587
	Wikipedia keyphraseness	0.527	0.867	0.756
	Inverse Wikipedia frequency	0.542	0.603	0.512
	Total Wikipedia frequency	0.531	0.794	0.723
Semantics	Node degree	0.799	0.632	0.645
	Semantic relatedness	0.549	0.703	0.525
Specificity	Term length	0.594	0.545	0.666
	Generality	0.531	0.508	0.515

Table 5.7 AUC values for each feature in FAO-780, WIKI-20 and CiteULike-180

compares the AUC values based on the three sample collections. Features are grouped according to their type: frequency, occurrence position, keyphraseness, semantics and specificity.

Among the frequency features, TF×IDF performs best on FAO-780 and WIKI-20, but it is outperformed by term frequency on CiteULike-180, where keyphrases are freely chosen. These findings support those reported by Hulth (2004) and Csomai and Mihalcea (2008), who chose term frequency over TF×IDF for the keyphrase extraction task. The IDF feature on its own performs poorly on all three corpora.

Occurrence features show high AUC values in all collections, with spread outperforming both the first and last occurrence, particularly on CiteULike-180. The conclusion is that, in any topic indexing task, the position of the candidate topic in the document determines its significance.

Domain keyphraseness outperforms all other features on WIKI-20, but performs worse than most other features on CiteULike-180. Wikipedia keyphraseness, in turn, is discriminative on CiteULike-180, but nearly random on FAO-780. This is perhaps due to the fact that only some Agrovoc terms used to index the FAO-780 collection could be mapped to Wikipedia articles. Analysis of other Wikipedia-based features on this collection shows that their performance is, in general,

weaker than on other corpora. The inverse Wikipedia frequency, and in fact all other Wikipedia-based features, except generality, perform best on WIKI-20, where all topics and candidates are Wikipedia articles.

The node degree feature is most useful on FAO-780, but it does seem informative on other collections as well. Using the semantic relatedness feature on FAO-780 and CiteULike-180 is not as straightforward as on WIKI-20, where it performs reasonably well. All candidates first need to be mapped to the most common Wikipedia article, but many cannot be mapped to any articles at all. The performance is particularly poor on CiteULike-180.

Finally, term length and generality, which measure the candidates' specificity, produce the weakest results. Length performs slightly better than generality, particularly on CiteULike-180 and FAO-780.

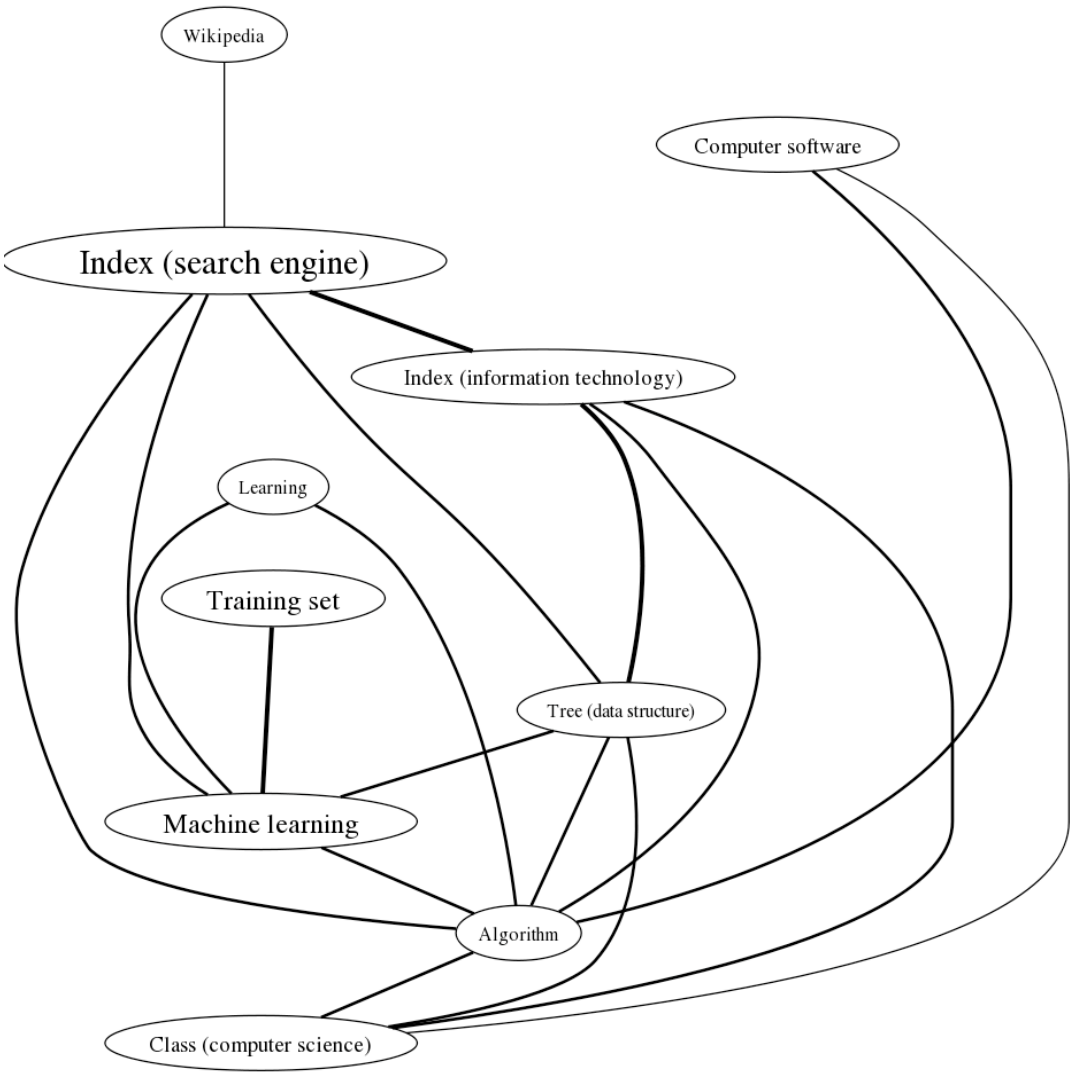
There are clear differences in the performance of the features depending on the corpus, i.e. the kind of indexing task. In the FAO-780 corpus, the best three performing features are domain keyphraseness, TF×IDF and term frequency; in WIKI-20 they are the domain keyphraseness, total Wikipedia frequency and spread; in CiteULike-20 they are term frequency, spread and TF×IDF. Many approaches apply a fixed formula that combine features which perform best on one particular corpus (Section 3.1.2 and 3.2.2). However, this analysis shows that the importance of the features varies depending on the corpus. These differences can be captured using machine learning.

5.3 Summary

This chapter described and evaluated two main stages in the topic identification process: candidate generation and filtering. The difficulties in the candidate generation depend on the task. In term assignment, the challenge is to bridge the gap between document and vocabulary terminology. Here, a combination of normalization techniques was explored. In keyphrase extraction with Wikipedia, the main problem is word sense ambiguity. Here, an unsupervised disambiguation algorithm was proposed and evaluated. Evaluation of the candidates shows that up to 80% of

manually assigned topics can be identified automatically, depending on the collection and the vocabulary.

Once generated, candidates are analyzed based on typical features of topics. We have shown how features relate to each other and how well they discriminate positive from negative candidates. Features perform differently depending on the topic indexing task, which needs to be captured using machine learning techniques. The next chapter explains how candidate generation and filtering methods are combined into Maui, a single generally applicable algorithm that uses machine learning to optimize performance of topic indexing.


Wikipedia
Computer software
Index (search engine)
Index (information technology)
Learning
Training set
Tree (data structure)
Machine learning
Algorithm
Class (computer science)

Chapter 6
The Maui
topic indexing algorithm

Chapter 5 discussed two important stages in automatic topic indexing: candidate generation and filtering. This chapter explains how these stages are integrated into the Maui algorithm, and how this algorithm performs topic indexing. Section 6.1 describes Maui's components and gives a general overview of the main steps in the indexing process. Subsequent sections present each step in detail, along with supporting examples. Finally, Section 6.6 explains how one actually uses Maui: either from the command line or integrated directly into the code.

6.1 Components

Maui contains four open-source software components.

Kea. Maui builds on the keyphrase extraction algorithm Kea (Witten *et al.*, 1999) by adopting its two-stage indexing process and inheriting some of its components. Kea's phrase filtering and *n*-gram extraction were introduced into Maui without major modifications. Others components such as feature computation, were extended with new elements. Whereas the original Kea was restricted to one kind of topic indexing—keyphrase extraction—Maui performs many related tasks.

Weka. Also inherited from Kea is the machine learning toolkit Weka for creating topic indexing models and applying them to new documents (Witten and Frank, 2005). However, Kea contained only a few of Weka's classes, whereas Maui plugs in the complete library. This gives an opportunity for experienced users to tailor Maui's code and optimize its performance for particular collections.

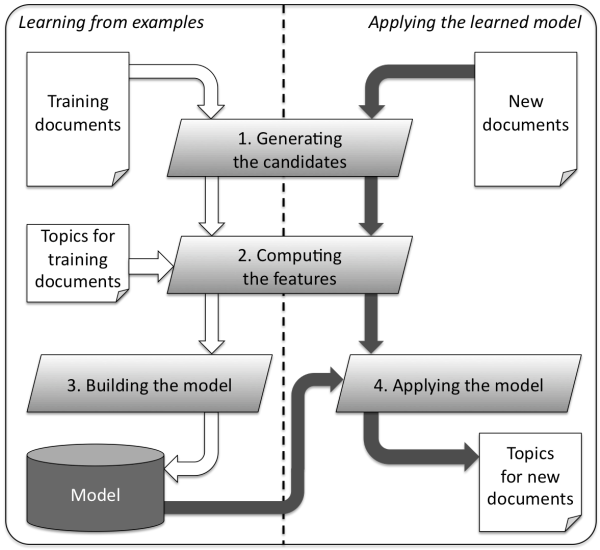


Figure 6.1 Operation of Maui

Jena. The Jena software library allows Maui to incorporate externally-produced controlled vocabularies (McBride, 2001). Using Jena, Maui reads RDF-formatted thesauri and stores them in memory for quick access.

Wikipedia Miner. Maui uses Wikipedia Miner to access Wikipedia data (Milne 2009). This tool converts Wikipedia dumps to MySQL database format and provides object-oriented access to parts of Wikipedia. It also computes semantic relatedness between articles, which Maui uses to disambiguate document phrases to Wikipedia articles and to compute semantic features.

These four software components are combined with classes created specifically for Maui to form a single topic indexing algorithm. Figure 6.1 shows its four main steps:

1. Generating candidate topics
2. Computing their features
3. Building the topic indexing model
4. Applying the model

The flow chart depicts two activities required for the indexing task: *learning the indexing model* from manually assigned topics (left, white arrows) and *applying the learned model* to compute topics for unseen documents (right, dark arrows). Maui implements a supervised machine learning approach, where a small training set

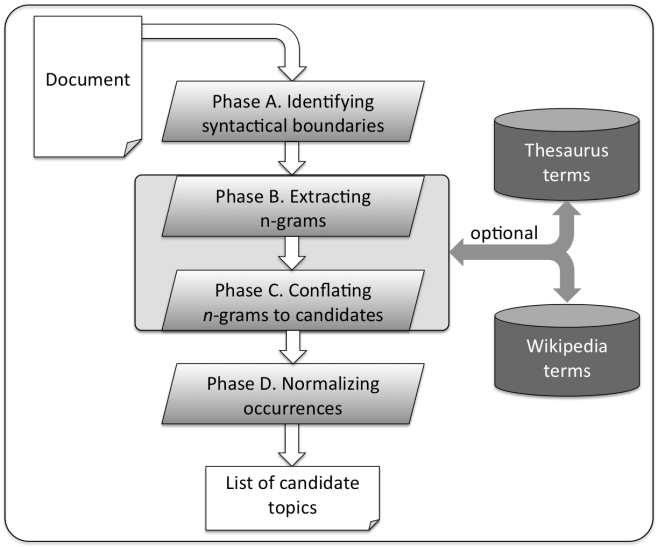


Figure 6.2 Maui's candidate generation

provides a model that can be used for fresh documents that had not been seen at training time.

6.2 Generating candidate topics

Maui generates candidate topics individually for each input document, based on its text, and outputs a list of candidate topics that is then passed onto the next step, which computes the features. This section presents the candidate generation part of the algorithm and shows how it applies to different kinds of topic indexing using example documents.

6.2.1 Algorithm

Figure 6.2 details the four phases in Maui's candidate generation step, while Figure 6.3 summarizes the actual algorithm.

Phase A. First the document text is analyzed to identify initial syntactic boundaries (Figure 6.3, lines 2–3). For this Maui uses Kea's PhraseFilter algorithm:

- Punctuation marks and numbers are replaced by boundary symbols;
- Apostrophes are removed;
- Hyphenated words are split in two;
- Any tokens that do not contain letters are deleted.

```
1   generateCandidates(documentText)
2       apply PhraseFilter to documentText
3       store the results in documentSegments

4       initialize allCandidates, a list in which to place the candidates

5       for each documentSegment in documentSegments
6           compute word n-grams up to a predefined length

7           if no vocabulary is used
8               if n-gram does not begin or end with a stopword
9                   normalize n-gram
10                  add(n-gram, allCandidates)
11          if vocabulary is a thesaurus
12              retrieve possible thesaurus terms for this n-gram
13              for each thesaurus term
14                  add(thesaurus term, allCandidates)
15          if vocabulary is wikipedia
16              if wikipedia keyphraseness of n-gram > 0.01
17                  add(n-gram, allCandidates)

18      if vocabulary is wikipedia
19          disambiguate each n-gram in allCandidates to a wikipedia article

20      for each candidate in allCandidates
21          normalize candidate's frequency and occurrence positions

22  return allCandidates
```

```
23  add(phrase, allCandidates)
24      if allCandidates contains phrase
25          retrieve candidate for phrase from allCandidates
26          update candidate's total frequency
27          update candidate's last occurrence
28          if no vocabulary is used
29              record the phrase's full form

30      else
31          create a new candidate
32          record this candidate's total frequency
33          record this candidate's first occurrence as current occurrence
34          record the last occurrence as current occurrence
35          if no vocabulary is used
36              record phrase's full form
37          if vocabulary is wikipedia
38              record anchor information for this phrase

39          add candidate to allCandidates
```

Figure 6.3 Algorithm for generating candidates in Maui

Task	Topics	What *n*-grams?	What conflation method?
Automatic tagging	Document phrases	Those that neither begin nor end with a stopword	Each word is stemmed. The most frequent full form of a stemmed *n*-gram is the topic's title.
Term assignment	Controlled vocabulary descriptors	Those that match vocabulary terms	Vocabulary terms and *n*-grams are both normalized to pseudo-phrases and all matching descriptors are retrieved.
Indexing with Wikipedia	Wikipedia articles	Those whose Wikipedia keyphraseness exceeds a threshold	*N*-grams are matched to Wikipedia articles and disambiguation is applied to determine their intended meaning.

Table 6.1 Details of candidate generation for each topic indexing task

The result is a set of textual segments (full sentences or their parts), each being a sequence of word tokens containing at least one letter.

Phase B. Maui extracts all subsequences of tokens of length *n* (*n*-grams) in each line. The value of *n* ranges between lower and upper limits defined by the user. For each *n*-gram Maui then determines whether it is a suitable candidate (Figure 6.3, lines 5–17). A different test is applied to the *n*-grams depending on the indexing task (see Table 6.1). In automatic tagging, candidate topics are sequences that do not begin or end with a stopword (Figure 6.3, lines 7–10). In term assignment, Maui accepts all sequences that match terms listed in a controlled vocabulary (lines 11–14). When indexing with from Wikipedia, Maui identifies *n*-grams with Wikipedia keyphraseness value over a given threshold (lines 15–17), i.e. those that are likely to appear as anchors in Wikipedia.

Phase C. The *n*-grams are conflated to a set of candidate topics as shown in lines 23 to 38 of Figure 6.3. A different conflation strategy is applied depending on the indexing task, because in each task a topic is a different kind of entity (Table 6.1). In *automatic tagging*, *n*-grams are conflated based on their matching normalized forms, as described in Section 5.1.3. For example, applying the Porter stemmer to unigrams (*1*-grams) *indexing*, *index* and *indexed* conflates them to the stem *index*. The most frequent full form in the document is used as the title for this candidate.

In *term assignment*, Maui maps *n*-grams to vocabulary terms by transforming both into pseudo-phrases and replacing matching non-descriptors with the equivalent descriptors. Jena library is used for accessing vocabularies in RDF format.

In *indexing with Wikipedia*, a topic is a Wikipedia article. Here Maui uses the Wikipedia Miner to retrieve matching Wikipedia articles and disambiguate them to those articles that are most similar to the unambiguous context (Figure 6.3, lines 18–19), as described in Section 5.1.2.

Phase D. When the last candidate has been identified, the document's length and the total number of candidate occurrences are known. Before outputting the candidates, Maui normalizes the occurrence positions by document length and the occurrence frequencies by the number of candidates (Figure 6.3, lines 20–21). These values are stored for future processing and the candidate list is passed to the next stage: computing the features (Section 6.3). The following sections demonstrate candidate generation for each task on sample documents, and shows the intermediate output of each phase.

6.2.2 Candidates from document text

As discussed in Section 5.1.3, the candidate generation for automatic tagging in Maui's implementation replicates the one used in Kea. Figure 6.4 illustrates each phase given an abstract from a sample document in the CiteULike-180 corpus (Section 4.3.1).

In Phase A, the text is transformed into several textual segments. Figure 6.4a shows the output for the abstract's first sentence. Next, in Phase B, Maui extracts a set of *n*-grams: Figure 6.4b lists all *n*-grams for $1 \leq n \leq 3$. Finally, in Phase C, Maui conflates the *n*-grams based on their normalized forms and computes statistics such as occurrence count, term frequency, and first and last occurrence position. Figure 6.4c lists all candidate topics, that appeared at least twice (Total count ≥ 2). The full form of each topic that is the most frequent in that document is used as the final answer (e.g. *alignment* rather than *aligning* for the candidate *align*).

CiteULike users have agreed on five topics (tags) for the example document: *rna*, *secondary structure*, *alignment*, *clustering* and *structure*. Four candidates' stems match the stems of these tags (*align*, *rna*, *cluster* and *structur*) and are shown in bold (Figure 6.4c). During training these candidate topics serve as positive examples, while all other candidates serve as negative examples.

Torarinsson, Havgaard and Gorodkin. 2006. **Multiple structural alignment and clustering of RNA sequences**. *Bioinformatics* 23(8), pp. 926. CiteULike: *http://www.citeulike.org/article/1133633*

ABSTRACT Motivation: An apparent paradox in computational RNA structure prediction is that many methods, in advance, require a multiple alignment of a set of related sequences, when searching for a common structure between them. However, such a multiple alignment is hard to obtain even for few sequences with low sequence similarity without simultaneously folding and aligning them. Furthermore, it is of interest to conduct a multiple alignment of RNA sequence candidates found from searching as few as two genomic sequences. Results: Here, based on the PMcomp program, we present a global multiple alignment program, FOLDALIGNM, which performs especially well on few sequences with low sequence similarity, and is comparable in performance with state of the art programs. …

a)

- An apparent paradox in computational RNA structure prediction is that many methods
- in advance
- require a multiple alignment of a set of related sequences
- when searching for a common structure between them

b) ABSTRACT, Motivation, ABSTRACT Motivation, apparent, paradox, apparent paradox, computational, paradox in computational, RNA, computational RNA, structure, RNA structure, computational RNA structure, prediction, structure prediction, RNA structure prediction, methods, advance, require, multiple, require a multiple, alignment, multiple alignment, set, related, set of related, sequences, related sequences, searching, common, structure, common structure

c)

Candidate ID, stems	*N*-grams' full forms (and their counts)	Total count	Term frequency	First occurrence	Last occurrence
sequenc	sequences (5), sequence (4)	9	0.0274	0.16	0.81
align	**alignment (5), aligning (1)**	**6**	**0.0183**	**0.12**	**0.86**
align multipl	multiple alignment (4)	4	0.0122	0.12	0.58
program	program (3), programs (1)	4	0.0122	0.55	0.94
low sequenc	low sequence (2), sequences with low (2)	4	0.0122	0.3	0.66
multipl	multiple (4)	4	0.0122	0.12	0.58
structur	**structure (3)**	**3**	**0.0091**	**0.05**	**0.83**
similar	similarity (3)	3	0.0091	0.32	0.83
perform	performance (1), performs (1)	2	0.0061	0.61	0.7
base	based (2)	2	0.0061	0.52	0.8
search	searching (2)	2	0.0061	0.17	0.46
result	results (1), Results (1)	2	0.0061	0.51	0.9
foldalignm	FOLDALIGNM (2)	2	0.0061	0.6	0.98
rna	**RNA (2)**	**2**	**0.0061**	**0.05**	**0.44**
sequenc similar	sequence similarity (2)	2	0.0061	0.32	0.66
base sequenc	sequences based (1), based on sequence (1)	2	0.0061	0.79	0.8
low	low (2)	2	0.0061	0.31	0.66
cluster	**cluster (2)**	**2**	**0.0061**	**0.79**	**0.88**
low sequenc similar	low sequence similarity (2)	2	0.0061	0.31	0.66

Figure 6.4 Candidate generation for a sample document in CiteULike-180

Mabugu, Milne and Campbell. 1998. **Incorporating fuelwood production and consumption into the national accounts. A case study for Zimbabwe.**
http://www.fao.org/docrep/005/AB603E/AB603E00.htm

Natural resource accounting methods are applied in a case study of fuelwood consumption in Zimbabwe. The study estimates values of economic depreciation of timber stocks from fuelwood consumption from 1990 to 1996. Fuelwood is an appropriate variable to study because of the country's high dependency on wood for energy, particularly in rural areas where most of the population lives. There is substantial criticism of the linkage between the environment and national accounts in most countries including Zimbabwe. Traditional national income data such as Gross Domestic Product (GDP) do not fully capture the total economic value of natural resource stocks such as forests.

a)
- Natural resource accounting methods are applied in a case study of fuelwood consumption in Zimbabwe
- …
- Traditional national income data such as Gross Domestic Product
- GDP
- do not fully capture the total economic value of natural resource stocks such as forests

b) Natural resource (Natural resources), accounting (Accounting), methods (Methods), case (Casings), case study (Case studies), fuelwood (Fuelwood), consumption (Consumption), Zimbabwe (Zimbabwe), economic (Economics), values of economic (Economic value), depreciation (Depreciation), timber (Wood), stocks (Handle stocks, Stocks), fuelwood (Fuelwood), consumption (Consumption), Fuelwood (Fuelwood), wood (Wood), energy (Energy), wood for energy (Wood energy), rural areas (Rural areas), population (Political systems), environment (Environment), accounts (Accounting), national accounts (National accounting), Zimbabwe (Zimbabwe), income (Income), national income (National income), data (Data), Domestic (Domestication), Product (Products, Productivity, Production), Domestic Product (Domestic production), Gross Domestic Product (Gross national product), economic (Economics), economic value (Economic value), natural resource (Natural resources), stocks (Handle stocks, Stocks), forests (Forestation, Forests, Foresters)

c)

Candidate ID	Descriptor title	*N*-grams' full forms (and their counts)	Total count	Term frequency	First occur.	Last occur.
3137	**Fuelwood**	**fuelwood (2), Fuelwood (1)**	**3**	**0.067**	**0.11**	**0.3**
62	**Accounting**	**accounts (1), accounting (1)**	**2**	**0.044**	**0.02**	**0.7**
8516	**Zimbabwe**	**Zimbabwe (2)**	**2**	**0.044**	**0.14**	**0.74**
35691	Economic value	economic value (1), values of economic (1)	2	0.044	0.18	0.91
3484	Handle stocks	stocks (2)	2	0.044	0.24	0.96
28772	Stocks	stocks (2)	2	0.044	0.24	0.96
5091	Natural resources	Natural resource (1), natural resource (1)	2	0.044	0.00	0.94
8421	Wood	wood (1), timber (1)	2	0.044	0.23	0.44
2481	Economics	economic (2)	2	0.044	0.2	0.91
1827	Consumption	consumption (2)	2	0.044	0.12	0.27

Figure 6.5 Candidate generation for a sample document in FAO-780

6.2.3 Candidates from controlled vocabularies

Mapping documents to terms in a controlled vocabulary was already discussed in Section 5.1.1. Given the abstract from a sample document in FAO-780 and the Agrovoc thesaurus (Section 4.1.1), Figure 6.5 shows the intermediate output of each phase. Once the text segments are identified (Figure 6.5a), Maui extracts *n*-grams and computes all matching descriptors in the vocabulary. These are shown in parentheses after each *n*-gram in Figure 6.5b. The *n*-gram set contains ambiguous cases, e.g. *Product* was mapped to *Products*, *Productivity* and *Production* because they all share the stem *product*. Without stemming, other matches would not have been possible, such as *case study* to *Case studies*. Some descriptors were found through non-descriptors, e.g. *timber* was correctly mapped to *Wood* via *Timber*. There is one erroneous mapping: the term *population* and the non-descriptor *Populism* were stemmed to *popul*, and *Populism* was then mapped to *Political systems*.

Conflation transforms the set of *n*-grams into a set of candidate terms, each uniquely identified by the descriptor id. Occurrence counts and their position values are computed for each candidate. Figure 6.5c shows candidates with Total count ≥ 2. Three out of ten topics assigned by professional indexers to this document (*Fuelwood*, *Accounting* and *Zimbabwe*) were identified in the abstract and are shown in bold. The remainder of the document (not shown) covers all other topics except one, *Statistical data*.

6.2.4 Candidates from Wikipedia

Section 5.1.2 explained Maui's approach to generating candidates using Wikipedia as a vocabulary. The individual steps are demonstrated in Figure 6.6 on the abstract from a WIKI-20 document (Section 4.2.2). Once the segments are identified (Figure 6.6a), Maui extracts *n*-grams. Because nearly all *n*-grams match an article title in Wikipedia, a pre-defined keyphraseness threshold is used to identify meaningful ones. Figure 6.6b lists all *n*-grams with keyphraseness > 0.01 (the actual values are shown in parentheses).

Bowen and Breuer. 1992.
Occam's Razor: The cutting edge of parser technology. In Proceedings of TOULOUSE'92: 5th Intern. Conf. on Software Engineering and its Applications, Toulouse, France, December 1992.
CiteSeer: *http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.18.7729*

Yacc is well established in the compiler-compiler field, but is beginning to show its age. Issues which were important when hardware resources were more scarce are now less critical. Precc is a new compiler-compiler tool that is much more versatile than yacc, whilst retaining efficiency of operation on modern computers. It copes with the context-dependent BNF grammar descriptions and higher order meta constructions that are naturally encountered in semi-formal concrete syntax specifications, building fast and efficient infinite-lookahead tools in the form of ANSI-compliant C code. This paper provides a demonstration of this state-of-the-art compiler-compiler technology using the programming language occam as an example. The parsing of occam is particularly difficult compared to some programming languages since the indentation is an integral part of the language. However the precc tool allows a natural implementation of an occam parser that follows the syntax very closely.

a)
- Yacc is well established in the compiler compiler field
- but is beginning to show its age
- …

b) Yacc (1), compiler (0.317), compiler compiler (1), field (0.011), hardware (0.086), scarce (0.012), tool (0.065), much more (0.011), yacc (1), efficiency (0.032), computers (0.037), copes (0.011), the context (0.071), BNF (0.891), grammar (0.072), BNF grammar (0.25), order (0.014), meta (0.14), semi formal (0.25), concrete (0.141), syntax (0.255), concrete syntax (0.125), infinite (0.042), lookahead (0.308), ANSI (0.545), C (0.14), code (0.012), C code (0.044), paper (0.036), demonstration (0.021), state (0.03), art (0.031), state of the art (0.052), technology (0.042), programming (0.036), language (0.036), programming language (0.463), occam (0.717), parsing (0.269), languages (0.013), programming languages (0.134), indentation (0.123), integral (0.039), parser (0.325)

c)

Candidate ID	Article title	N-grams' full forms (and their counts)	Total count	Term freq.	First occur.	Last occur.
5739	**Compiler**	**compiler (6)**	**6**	**0.105**	**0.04**	**0.67**
17524	Language	languages (1), language (2)	3	0.053	0.71	0.88
22660	**Occam (programming language)**	**occam (3)**	**3**	**0.053**	**0.71**	**0.95**
70097	**Compiler-compiler**	**compiler compiler (3)**	**3**	**0.053**	**0.04**	**0.67**
23015	**Programming language**	**programming language (1), languages (1), programming languages (1)**	**3**	**0.053**	**0.7**	**0.81**
34358	**Yacc**	**yacc (1), Yacc (1)**	**2**	**0.035**	**0**	**0.28**
6021	**C (programming language)**	**C code (1) C (1)**	**2**	**0.035**	**0.58**	**0.58**
5311	Computer programming	programming (2)	2	0.035	0.7	0.8
310015	**Parsing**	**parsing (1), parser (1)**	**2**	**0.035**	**0.75**	**0.95**
30677	Tool	tool (2)	2	0.035	0.24	0.9
26860	**Syntax**	**syntax (2)**	**2**	**0.035**	**0.48**	**0.98**
62247	**Backus–Naur Form**	**BNF (1), BNF grammar (1)**	**2**	**0.035**	**0.38**	**0.38**

Figure 6.6 Candidate generation for a sample document in WIKI-20

An *n*-gram that matches just one article is unambiguous, and is used for disambiguating other *n*-grams. There are two unambiguous *n*-grams in our example: *yacc* and *compiler compiler*, matching *Yacc* and *Compiler-compiler* respectively. Additional context articles are collected from highly likely mappings of ambiguous *n*-grams, those with a sense commonness of at least 0.9, e.g. *Backus–Naur Form* (from the *n*-gram *BNF grammar*), *Abstract syntax* (from *concrete syntax*) and *Lookahead* (from *lookahead*). The resulting five articles are used as context for disambiguation.

Tool is one example of an ambiguous *n*-gram. Its most common sense, according to Wikipedia, is *Tool (band)* with a probability of 0.49. However, this sense is discarded because its similarity to the context articles is zero. The less likely senses are *Tool* in its broad definition, with a probability of 0.48, *Programming tool* with 0.005 and *Tool (insult)* with 0.003. The last one is discarded because its commonness is below the threshold (0.05). The remaining two fit to this context with semantic relatedness ≈ 0.1 (computed as described in Section 5.1.2), but only *Tool* (the broad term) reaches the disambiguation score threshold and is accepted as a mapping.

Figure 6.6c lists all candidates (with Total count ≥ 2) generated from the given abstract. Bold font indicates positive terms—those that match a topic assigned by any of the 15 teams that indexed this documents (Section 5.2.2). There are many more positive candidates here than in the examples of automatic tagging and term assignment because more manually assigned topics are available for this document—34, as opposite to 5 in the tagging example (Section 6.2.2) and 10 in the term assignment one (Section 6.2.3).

6.3 Computing the features

Section 5.2 surveyed features that reflect the significance of candidate topics in topic indexing. Some of the feature values are determined during the candidate generation step:

- *term frequency* – the occurrence count for each candidate relative to the total occurrence frequency of all candidates

- *first occurrence* – the position of the first occurrence for each candidate relative to the number of words in the document
- *last occurrence* – the position of the last occurrence for each candidate relative to the number of words in the document

The remaining ones are computed once the candidate topics are identified (Figure 6.1, step 2). In the training stage, first, a dictionary containing global frequencies is created, in which Maui records how many documents contain each candidate topic (n_t), and the total number of documents (N). Also in the training stage, manually assigned topic sets are used to construct a dictionary giving the frequency of each topic that appears in these sets (m_t).

Given these pre-computed statistics, access to a controlled vocabulary and Wikipedia data, all features can now be computed:

- *inverse document frequency* $= -\log_2 \frac{n_t}{N}$
- *TF×IDF = term frequency × inverse document frequency*
- *spread = last occurrence – first occurrence*
- *domain keyphraseness* is 0 if the candidate topic never appears in a manually assigned topic set and m_t otherwise
- *Wikipedia keyphraseness* involves matching candidate title against anchors appearing in the Wikipedia corpus. If Wikipedia is used as a vocabulary, values are pre-computed during candidate generation
- *inverse Wikipedia frequency* is computed by retrieving the most likely Wikipedia article for the current candidate (unless the candidate is a Wikipedia article itself) and counting the number of its incoming links
- *total Wikipedia keyphraseness* is the sum of *Wikipedia keyphraseness* values over all *n*-grams that were mapped to the Wikipedia article corresponding to the given candidate
- *node degree* is computed for candidates whose semantic relations are described in a vocabulary, and is the number of their related candidates in the given document divided by the total number of all candidates

- *semantic relatedness* is the total relatedness of the Wikipedia article representing the candidate to Wikipedia articles identified for all other candidates computed using Wikipedia Miner
- *term length* is the number of words in the candidate topic's name
- *generality* is computed for candidates that were mapped to Wikipedia articles, and is the distance between the category corresponding to the article and the root of the category tree, normalized by the tree depth
- *class value*, which is only known for training documents, is 1 if the candidate has been assigned manually, and 0 otherwise

Feature values are stored as double-length floating point numbers and passed to the next step. During training the next step is *Building the model* (Figure 6.1, step 3), and the feature values are used to create a topic indexing model (Section 6. 3). Otherwise, the next step is *Applying the model* (Figure 6.1, step 4), and the feature values are used in the model created during the training process. Based on this comparison each candidate receives a probability of being a topic (Section 6.4).

6.4 Building the model

Section 5.2.6 shows that the performance of features depends on the indexing task. Machine learning allows the algorithm to capture such dependencies. During training Maui uses a set of documents whose topics are known. For each one, the candidates are identified and their feature values computed as described above. To reduce the size of the training set all candidate topics that occur in a document only once may be discarded. This is normally not required when a vocabulary is used because that automatically restricts the candidate set.

Each candidate topic receives a class value indicating whether it is a positive example, matching a topic that has been manually assigned to this document, or a negative one. The class value is used by the machine-learning scheme to train a model that predicts the class for candidates extracted from unseen documents using the other feature values.

Maui includes the entire Weka toolkit (Witten and Frank, 2005), and any classifier can be chosen to build the model. Experiments with different topic indexing

tasks and corpora have shown that *Naïve Bayes* and *bagged decision trees* outperform other classifiers (Section 7.2). The discussion below is restricted to these two.

6.4.1 Naïve Bayes with discretization

Like Kea, Maui can be used with Naïve Bayes, a simple but powerful classifier (Domingos and Pazzani, 1997). To apply Naïve Bayes, Maui's numeric features are converted to nominal form. This process, called *discretization*, involves automatic detection of numeric ranges for each feature based on analysis of the training data (Witten and Frank, 2005). A discretization table records the numeric ranges and is used to replace the actual values by the corresponding range identifier. The same ranges are used to discretize features for unseen documents.

Table 6.2 shows Weka's output of Naïve Bayes classifier generated using the WIKI-20 corpus. Table 6.2a summarizes the discretization boundaries for each feature. For example, the discretized values for the *Domain keyphraseness* feature fall into three ranges: candidates that never appear as topics in the training set (≤ 0.5), those that appear up to seven times (0.5–6.5), and more frequently (> 6.5). Weka's Naïve Bayes performs discretization internally using the supervised method of Fayyad and Irani (1993).

The Naïve Bayes model also contains *conditional probabilities* (Table 6.2b). These are feature weights learned from positive and negative examples in the training data. For example, P[FirstOccurrence=2|yes] is the proportion of positive examples that have a discretized *FirstOccurrence* value of 2. In this case 13.2% of the positive instances have values ranging from 0.017 to 0.043, and therefore the resulting 0.132. The distribution of values for each feature gives an approximate picture of its usefulness. For example, the distribution for P[*Length*|yes] and P[*Length*|no] are nearly identical, whereas for P[*TWK*|no] most instances are in range 1 (0.562), and for P[*TWK*|yes] they are mostly in range 4 (0.351).

The final component of the model are the *prior probabilities* computed using the number of positive and negative examples in the training data (Table 6.2c). These are the probabilities of a candidate being a topic, in the absence of any other information. Section 6.4.1 describes how this model is applied to unseen documents.

a)

Features	Discretization ranges			
	1	2	3	4
Term Frequency	≤ 0.002	(0.0012-0.003]	(0.003-0.011]	> 0.011
TFxIDF	≤ 0.002	(0.002-0.007]	(0.007-0.026]	> 0.026
First occurrence	≤ 0.017	(0.017-0.043]	(0.043-0.19]	> 0.19
Last occurrence	≤ 0.82	> 0.82		
Spread	≤ 0.043	(0.043-0.708]	(0.708-0.943]	> 0.943
Domain keyphraseness	≤ 0.5	(0.5-6.5]	> 6.5	
Length	≤ 1.5	> 1.5		
Generality	≤ 0.594	>.594		
Node degree	≤ 1.5	(1.5-5.5]	(5.5-23.5]	> 23.5
Semantic relatedness	≤ 0.017	(0.017-0.169]	(0.169-0.274]	> 0.274
Wikip. keyphraseness	≤ 0.059	(0.059-0.225]	(0.225-1.05]	> 1.05
Inverse Wikipedia freq	≤ 5.916	(5.916-12.808]	> 12.808	
Total Wikipedia keyphr	≤ 0.169	(0.169-1.192]	(1.192-2.63]	> 2.63

b)

Features	Values	Discretization ranges			
		1	2	3	4
Term Frequency	P[*TermFreq*|no]	0.695	0.155	0.113	0.037
	P[*TermFreq* |yes]	0.230	0.185	0.266	0.319
TFxIDF	P[*TFxIDF*|no]	0.602	0.291	0.090	0.017
	P[*TFxIDF*|yes]	0.187	0.264	0.364	0.185
First occurrence	P[*FirstOccurrence*|no]	0.050	0.050	0.183	0.717
	P[*FirstOccurrence*|yes]	0.348	0.132	0.211	0.309
Last occurrence	P[*LastOccurrence*|no]	0.573	0.427		
	P[*LastOccurrence*|yes]	0.305	0.695		
Spread	P[*Spread*|no]	0.615	0.263	0.097	0.025
	P[*Spread*|yes]	0.195	0.264	0.296	0.245
Domain Keyphraseness	P[*DomainKeyphr*|no]	0.889	0.095	0.016	
	P[*DomainKeyphr*|yes]	0.299	0.476	0.225	
Length	P[*Length*|no]	0.741	0.259		
	P[*Length*|yes]	0.621	0.379		
Generality	P[*Generality*|no]	0.178	0.822		
	P[*Generality*|yes]	0.074	0.926		
Node degree	P[*NodeDegree*|no]	0.260	0.252	0.407	0.081
	P[*NodeDegree*|yes]	0.040	0.145	0.528	0.288
Semantic relatedness	P[*SemanticRel*|no]	0.148	0.626	0.212	0.013
	P[*SemanticRels*|yes]	0.005	0.462	0.441	0.092
Wikipedia keyphraseness	P[*WikipKeyphr*|no]	0.470	0.267	0.248	0.015
	P[*WikipKeyphr* |yes]	0.132	0.206	0.467	0.195
Inverse Wikip frequency	P[*IWF*|no]	0.040	0.920	0.040	
	P[*IWF*|yes]	0.005	0.992	0.003	
Total Wikip keyphr	P[*TWK*|no]	0.562	0.358	0.054	0.025
	P[*TWK*|yes]	0.124	0.335	0.190	0.351

c)

Class	Training instances	Prior probability
no	6105	P(no) = N/(Y+N) = 0.942
yes	375	P(yes) = Y/(Y+N) = 0.058

Table 6.2 Output of the Naïve Bayes classifier

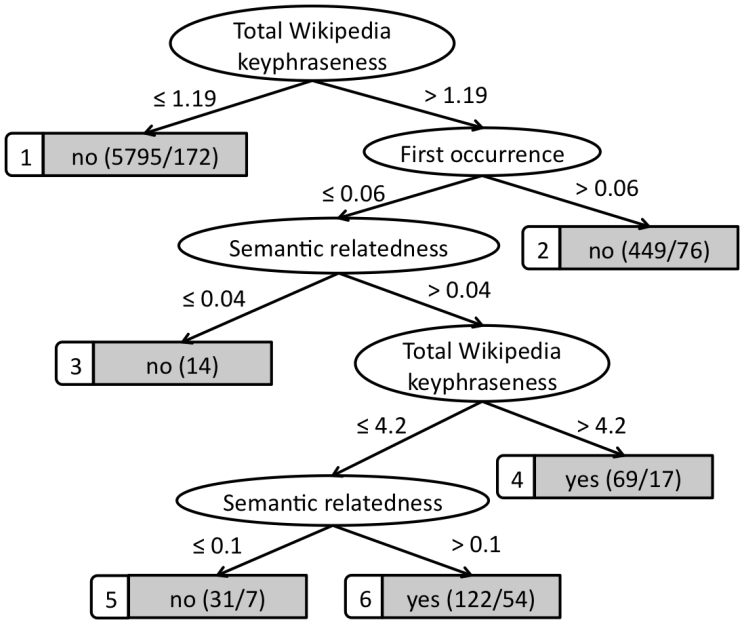


Figure 6.7 Decision tree generated using WIKI-20 documents and three features

6.4.2 Decision trees with bagging

One limitation of the Naïve Bayes classifier is that it implicitly assumes that the features are independent of each other, given the classification. This is not an issue in Kea, which uses only three features: *TF×IDF*, *first occurrence* and *domain keyphraseness*. In contrast, many of Maui's *features* are related, e.g. *first occurrence* and *spread*, or *node degree* and *semantic relatedness*. It is advantageous for the classifier to be able to handle such dependencies.

A *decision tree* is an example of such a classifier. A decision tree is a graph built automatically based on the distribution of feature values in the training data. Figure 6.7 shows a tree created using the FAO-780 corpus: for ease of demonstration only the three features—*total Wikipedia keyphraseness*, *semantic relatedness* and *first occurrence*—are used in this example. The nodes represent features, with the most powerful one at the root node (*total Wikipedia keyphraseness*). The edges are feature values. The leaf nodes are class values that the tree will assign to new candidates: they also show how many training examples were correctly and incorrectly classified with the given label. For example, node 1 [*no (5795/172)*] means that 5795 candidates exhibit *Total Wikipedia keyphraseness* ≤ 1.19, but 172 candidates are positive. A node with just one number (e.g., node 3) is a pure node that

contains only training instances of the assigned class. Maui uses *J48 decision trees* implemented in Weka based on the principles of the C4.5 algorithm (Quinlan, 1993). Proportions at leaf nodes can be used as class probability estimates.

The performance of decision trees can be improved using *bagging*, which learns an ensemble of classifiers and uses them in combination, thereby often achieving significantly better results than the individual classifiers (Breiman, 1996). Hulth (2004) also found that bagging improves the performance of keyphrase extraction. Maui uses *bagged decision trees*, which are generated by sampling from the original dataset with replacement. Like Naïve Bayes, bagged trees yield probability estimates—Weka's bagging averages probability estimates obtained from individual trees—that can be used to rank candidates.

6.5 Applying the model

To generate topics for a new document, Maui first determines candidates and their feature values and then applies the model built during training. The class feature is, of course, undetermined. The overall probability that each candidate is a topic is determined according to the model, and those candidates that yield the highest probability values are chosen as topics.

6.5.1 Applying Naïve Bayes to new documents

Suppose Naïve Bayes is used with just the two features *TF×IDF* (*t*) and *First occurrence* (*f*). For each candidate, Maui first determines feature values for *t* and *f* and then computes two joint probabilities, $P[yes,t,f]$ and $P[no,t,f]$:

$$P[yes,t,f] = \frac{Y}{Y+N} P[t|yes] P[f|yes]$$

(and similarly for $P[no,t,f]$). Here, Y and N are the number of positive and negative candidates in the training set. The Laplace estimator is used to avoid zero probabilities. This simply replaces Y and N by Y+1 and N+1. The overall probability that the candidate is a topic is then calculated as:

$$P[yes|t,f] = \frac{P[yes,t,f]}{\left(P[yes,t,f] + P[no,t,f]\right)}$$

	TFxIDF				*First occurrence*			
	feature values		condit. prob.		feature values		condit. prob.	
	Value	Range	P[*t*|yes]	P[*t*|no]	Value	Range	P[*f*|yes]	P[*f*|no]
Yacc	0.05	4	0.185	0.017	0.003	1	0.348	0.050
Language	0.009	2	0.264	0.291	0.16	3	0.211	0.183

Table 6.3 Feature values and conditional probabilities for *Yacc* and *Language*

Yacc	$P[\text{yes},t,f] = 0.058 \times 0.185 \times 0.348$	$= 0.0037$
	$P[no,t,f] = 0.942 \times 0.07 \times 0.05$	$= 0.0034$
	$P[\text{yes}\|t,f] = 0.0037 / (0.0037 + 0.0034)$	$= \mathbf{0.5211}$
Language	$P[\text{yes},t,f] = 0.058 \times 0.264 \times 0.21$	$= 0.0032$
	$P[no,t,f] = 0.942 \times 0.291 \times 0.183$	$= 0.0502$
	$P[\text{yes}\|t,f] = 0.0032 / (0.0032 + 0.0502)$	$= \mathbf{0.0599}$

Figure 6.8 Computing final class probabilities for *Yacc* and *Language*

Take as an example two candidate topics *Yacc* and *Language* generated for a WIKI-20 document "Occam's Razor: The Cutting Edge for Parser Technology" that was excluded from the training set. Table 6.3 lists feature values, discretization ranges and conditional probabilities for both terms. The *conditional probabilities* ($P[t|yes]$, $P[f|yes]$, with corresponding values for *no*) are copied from Table 6.2 (b). The *prior probabilities* (the first elements in the $P[t]$ and $P[f]$ formulas) were listed in Table 6.2 (c): 0.058 for the positive and 0.942 for the negative class. Given these values, the probabilities of *Yacc* and *Language* being topics for this document are computed as shown in Figure 6.8. *Yacc* receives a probability of 0.5211, and *Language* 0.059. Consequently, *Yacc* will appear higher in the ranked candidate list and is more likely to be chosen as a topic in the final step.

6.5.2 Applying bagged decision trees to new documents

Returning to the example decision tree shown in Figure 6.7, given the feature values for candidates *Yacc* and *Language* listed in Table 6.4, the tree maps the candidates to nodes 6 (*yes*) and 1 (*no*) respectively. At each leaf node, the probability of the class values is computed using the distribution of training instances. At leaf node 1 (*no, 5795/172)* the probabilities are:

$$P[yes|TWK] = \frac{172}{5795} = 0.0297 \qquad P[no|TWK] = \frac{5795 - 172}{5795} = 0.9703$$

	Total Wikipedia keyphraseness	First occurrence	Semantic relatedness
Yacc	23	0.003	0.17
Language	0.26	0.16	0.11

Table 6.4 Additional feature values for *Yacc* and *Language*

P[*yes*|*TWK*] refers to the probability of the candidate being a topic and is used for ranking. Its value for *Yacc* is 0.6932. Thus both Naïve Bayes and the decision tree predict the same ranking for the two candidates.

As noted in Section 6.2.1, Maui uses bagged decision trees where not just one but a set of trees is generated from different samples of the training data. The final probability of a candidate being a topic is computed by taking the average probability over all trees.

6.5.3 Specifying the final answer set

Candidates are now ranked according to their final probability values. When Naïve Bayes is used, TF×IDF (in its pre-discretized form) serves as a tiebreaker if two candidates have equal probability. From the resulting ranked list, the top k candidates with the greatest individual probabilities are returned, where k is the number of topics determined by the user (e.g. $k = 5$). Of course, in practice human indexers assign different numbers of topics to different documents. However, our data sets demonstrate that while the number of manually assigned topics does differ from document to document, there is little variation when these values are averaged over all indexers.

Maui can also be used for semi-automatic indexing by setting the threshold to generate a long list of potential topics, from which a human indexer selects the most appropriate ones. A joint list of candidate topics generated for the entire document collection can also be used for generating a back-of-the-book index or extracting terminology discussed in a collection.

6.6 Usage examples

The complete Maui code can be downloaded from two sources:

- Google Code**:** *http://maui-indexer.googlecode.com/*

General command:

java maui.main.MauiModelBuilder (or maui.main.MauiTopicExtractor)
–l *directory* –m *model* –v *vocabulary* –f {skos|text} –w *database@server*

Examples with experimental data supplied in the Maui package:

1. Automatic tagging
MauiModelBuilder –l *data/automatic_tagging/train/* –m *tagging_model*
MauiTopicExtractor –l *data/automatic_tagging/test/* –m *tagging_model*

2. Term assignment
MauiModelBuilder –l *data/term_assignment/train/* –m *assignment_model* –v *agrovoc* –f *skos*
MauiTopicExtractor –l *data/term_assignment/test/* –m *assignment_model* –v *agrovoc* –f *skos*

3. Topic indexing with Wikipedia
MauiModelBuilder
–l *data/wikipedia_indexing/train/* –m *indexing_model* –v *wikipedia* –w *enwiki@localhost*
MauiTopicExtractor
–l *data/wikipedia_indexing/test/* –m *indexing_model* –v *wikipedia* –w *enwiki@localhost*

Figure 6.9 Using Maui from the command line

- SourceForge**:** *http://maui-indexer.sourceforge.com/*

The package includes the code, libraries and sample data for three topic indexing tasks: term assignment, topic indexing with Wikipedia and tagging. If Wikipedia is used for topic indexing or for computing encyclopedic features, Wikipedia Miner should be installed first. The installation steps are detailed in Appendix E.

Once Maui is installed, there are two ways of using it: from the command line and from the Java code. Either way, the input data needs to be prepared first. Each document should be stored individually in text form, in a file with extension *.txt*. Maui takes as an input the name of the directory containing these files. If a model is created first, the same directory should contain manually assigned topics for each document, stored in individual files, one topic per line, named as the document text but with extension *.key*. If Maui is used to generate main topics for new documents, it will create *.key* files for each document in the input directory. If topics are generated but the output directory already contains *.key* files, the existing topics are used to evaluate the automatically extracted ones.

```
// Location of the data
String trainDir = "data/term_assignment/train_fr";
String testDir = "data/term_assignment/test_fr";

// Name of the file for storing the model
String modelName = "french_model";

// Language specific settings
Stemmer stemmer = new FrenchStemmer();
Stopwords stopwords = new StopwordsFrench();
String language = "fr";
String encoding = "UTF-8";

// Vocabulary to use for term assignment
String vocabulary = "agrovoc_fr";
String format = "skos";

MauiModelBuilder modelBuilder = new MauiModelBuilder();
MauiTopicExtractor topicExtractor = new MauiTopicExtractor();
Wikipedia wikipedia = new Wikipedia("localhost", "enwiki_20090306", "root", null);

// Settings for the model builder
modelBuilder.setDirName(trainDir);
modelBuilder.setModelName(modelName);
modelBuilder.setVocabularyFormat(format);
modelBuilder.setVocabularyName(vocabulary);
modelBuilder.setStemmer(stemmer);
modelBuilder.setStopwords(stopwords);
modelBuilder.setDocumentLanguage(language);
modelBuilder.setEncoding(encoding);
modelBuilder.setWikipedia(wikipedia);

// Which features to use?
modelBuilder.setFrequencyFeatures(false);
modelBuilder.setBasicWikipediaFeatures(true);

// Run model builder
modelBuilder.buildModel(modelBuilder.collectStems());
modelBuilder.saveModel();

// Settings for the topic extractor
topicExtractor.setDirName(testDir);
topicExtractor.setModelName(modelName);
topicExtractor.setVocabularyName(vocabulary);
topicExtractor.setVocabularyFormat(format);
topicExtractor.setStemmer(stemmer);
topicExtractor.setStopwords(stopwords);
topicExtractor.setDocumentLanguage(language);
topicExtractor.setWikipedia(wikipedia);

// Run topic extractor
topicExtractor.loadModel();
topicExtractor.extractKeyphrases(topicExtractor.collectStems());
```

Figure 6.10 Configuring Maui for French documents

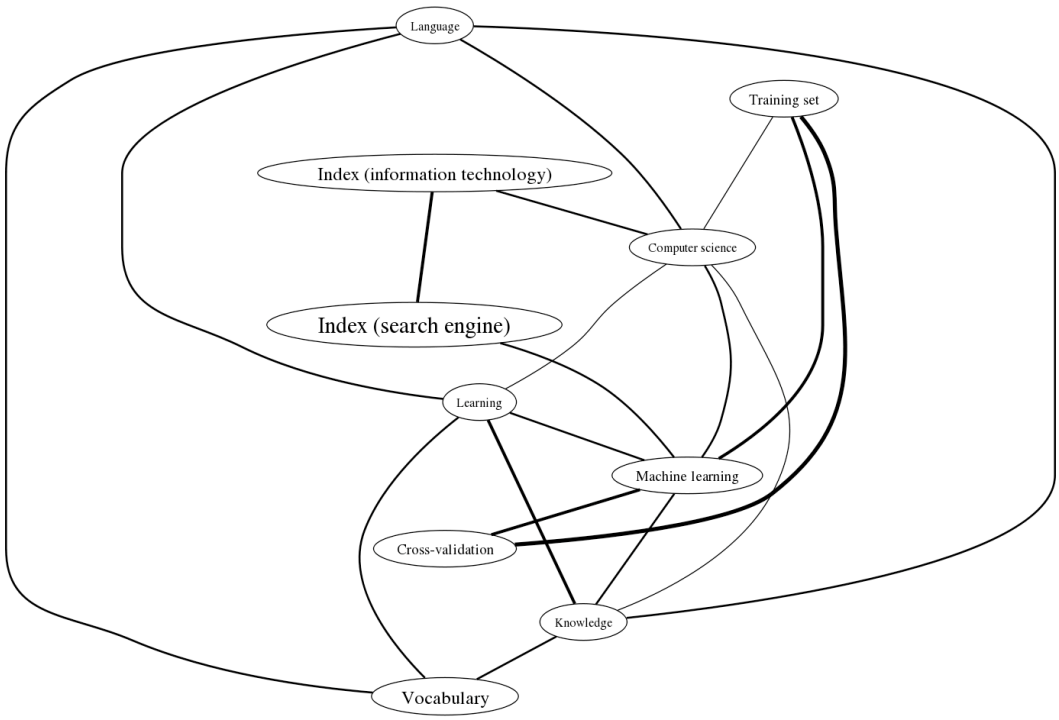
Maui also supports graphical visualization of the topics. If graphical output is selected, a *.gv* file is created for each document in the input directory. This file contains a plain-text description of a graph in the format used by the GraphViz software.[1] Maui produces a topic graph in which nodes represent topics and edges represent semantic relations between them. The font size indicates the significance of the topic, the edge thickness corresponds to the strength of the semantic relation, determined by Wikipedia Miner (Milne, 2009). GraphViz (or any other graph rendering program, e.g. OmniGraffle[2]) can be used to visualize the graph. The diagrams at the beginning of each chapter of this thesis show examples. They were generated from the top ten topics of each chapter, automatically computed by Maui after training on 20 manually indexed computer science articles (WIKI-20 corpus).

To create a model, Maui takes the name of the directory with *.txt* and *.key* files and the name of the output file for the model. If a controlled vocabulary is used, its name and its format need to be specified: it should be placed in Maui's main folder, in the directory *data/vocabularies*. If Wikipedia is used as a vocabulary, the name and the location of the database containing Wikipedia data are required. Figure 6.9 provides examples of command line arguments for the three types of topic indexing supported in Maui. There are many additional arguments, listed in Appendix F.

Figure 6.10 demonstrates how to use Maui directly from the Java code, given a specific use case: assignment of terms from the Agrovoc thesaurus to French documents. The data sets with sample files are provided in the download package. Only non-default settings are included for brevity. For example, by default Maui uses the frequency features, but not the basic Wikipedia features like *Wikipedia keyphraseness*. In this example, the opposite settings are chosen (Figure 6.10, lines 28-29). Other feature settings and additional use case scenarios are available in the class *maui.main.Examples* distributed as a part of Maui's download package.

---

[1] *http://www.graphviz.org/*
[2] *http://www.omnigroup.com/applications/OmniGraffle/*


Language
Training set
Index (information technology)
Computer science
Index (search engine)
Learning
Machine learning
Cross-validation
Knowledge
Vocabulary

Chapter 7
Evaluation of Maui

This thesis claims that automatic topic indexing can produce human-competitive results when given access to domain and background knowledge. Chapter 6 presented Maui, an algorithm that combines features reflecting different types of knowledge. This chapter evaluates Maui, and the knowledge behind it, on a series of topic indexing tasks that span several domains and languages. In each task, human performance is used as gold standard against which Maui is compared.

The evaluation strategy is explained in Section 7.1, and the remaining sections are dedicated to different topic indexing tasks. Section 7.2 evaluates Maui's performance in term assignment, where topics originate from domain-specific thesauri in agricultural, medical and physics domains, and on documents written in English, French and Spanish. It also compares the algorithm's topics with those assigned by humans specializing in indexing agricultural documents. Section 7.3 evaluates Maui's ability to choose topics from a controlled vocabulary of almost unprecedented size, Wikipedia. Section 7.4 investigates the quality of automatic tagging, where no vocabulary is used and tags are chosen directly from document text. In this task Maui is evaluated against 300 human taggers. The experiments demonstrate that Maui is applicable to a wide range of tasks, and automatically identifies document topics as well as people do.

7.1 Evaluation strategy

Three standard machine learning techniques are used to estimate Maui's performance: 10-fold cross-validation, leave-one-out and random sampling (Witten and

Frank, 2005). While human performance serves as the gold standard, the algorithm was also tested against several baselines.

7.1.1 Experimental settings

For *10-fold cross-validation*, the document collections are partitioned randomly into ten disjoint sets. Testing is performed on one set and the rest are used for training. The procedure is repeated ten times so that each set, and each document, is used nine times for training and once for testing. For example, with 780 documents Maui is trained on 702 documents and tested on the remaining 78, for each of ten runs. The results are averaged over all documents and runs. Cross-validation helps mitigate variance in random splits.

Leave-one-out evaluation is *n*-fold cross-validation where *n* is the number of documents in the corpus. Each document in turn is held out and the remainder are used to create a training model, which is tested on the selected document. This method uses the greatest possible amount of data for training, and therefore squeezes the maximum information from small datasets.

Random sampling is used in an experiment that evaluates Maui's dependence on training documents. Training data—i.e., manually indexed documents—is usually difficult to obtain, and it is interesting to know how much is necessary for good results. Frank *et al.* (1999) show that for keyphrase extraction with Kea, performance does not improve once the training set reaches 50 documents. We repeat the experiment on a topic indexing task that uses a controlled vocabulary, by using random samples of different sizes from the training set. For each size, ten samples are taken and the results are averaged over all runs.

7.1.2 Baselines

To provide a baseline for comparison, a simplified topic indexing approach was implemented: first identify candidate topics using the same strategy as in Maui (Section 5.1), and then select as topics those candidates with the greatest TF×IDF values. Section 5.2.6 demonstrates that TF×IDF is one of Maui's strongest fea-

tures, and this baseline is powerful and difficult to beat. Any improvement in performance reflects the contribution of additional features and the advantage of using machine learning techniques to incorporate this new information, which also depends on the volume of training data.

Frank *et al.*'s (1999) "domain keyphraseness" feature (referred to as *keyphraseness* throughout this chapter) could be used as an alternative baseline, but it does not in general outperform TF×IDF. It performs best on highly focused collections like medical (NLM-500) and science (CiteULike-180) documents. Appendix D.1 summarizes the results when topics were generated from candidates with the highest keyphraseness, computed over all other documents in the same collection.

Maui builds on the success of the keyphrase extraction algorithm Kea (Witten *et al.*, 1999, Frank *et al.*, 1999) and its extension Kea++, described in my Masters thesis (Medelyan, 2005). Both algorithms are used as additional baselines.

Where possible, Maui is compared directly to competitive systems. For term assignment we use the Medical Text Indexer, developed at the National Library of Medicine (Gay *et al.*, 2005), and BibClassify, available through the Center for Nuclear Research (Pepe and Yeomans, 2008). For topic indexing with Wikipedia, we use an algorithm developed at the Russian Academy of Sciences (Grineva *et al.*, 2009).

Results are evaluated using the measures introduced in Section 2.2. When only single topic sets per documents are available, precision (P), recall (R) and F-measure (F) are computed. When multiple topic sets for each document are known, Rolling's (1991) inter-indexer consistency measure is applied. F-measure and average inter-indexer consistency values provide a basis for comparing different settings and systems.

Stemming deserves a separate mention. In Chapter 5 four settings were evaluated: the Lovins (1968), Porter (1980) and *s*-removal stemmers, and no stemming. In most collections the Porter stemmer produced candidate sets that covered the greatest number of manually assigned topics. Previous evaluation of conflation rate of different stemmers has also shown that Porter produces the best re-

		P	R	F	P	R	F
1	TF×IDF baseline	16.1	17.5	16.8			
		Naïve Bayes			Bagged decision trees		
2	TF×IDF	21.4	23.2	22.2	6.2	6.9	6.5
3	+ First Occurrence	23.2	25.7	24.4	22.6	25.0	23.7
4	+ Keyphraseness	27.2	29.3	28.2	27.6	29.7	28.6
5	+ Node degree	28.8	30.9	29.8	27.7	30.3	29.0
6	+ Length	**30.9**	**33.2**	**32.0**	31.7	34.4	33.0
7	+ Frequency features	29.3	31.4	30.3	31.6	34.3	32.9
8	+ Occurrence features	21.0	22.8	21.9	30.7	33.2	31.9
9	+ Wik. keyphraseness	21.3	23.0	22.1	32.3	35.0	33.6
10	+ Total Wik. keyphraseness	17.6	18.8	18.2	31.7	34.3	32.9
11	+ Inverse Wik. frequency	17.0	17.9	17.4	32.7	35.5	34.0
12	+ Generality	16.4	17.3	16.8	**32.9**	**35.6**	**34.2**

Table 7.1 Sequential addition of features on FAO-780

sults (Fuller and Zobel, 1998). Therefore this stemmer was used in all experiments reported in this chapter except topic indexing with Wikipedia, where no stemming was used. Re-evaluating the final results using other stemmers showed that either the Porter stemmer produced better topics, or there was no significant difference between the settings.

7.2 Quality of term assignment

Here we test Maui's performance on the data sets described in Section 4.1, beginning with the agricultural domain. The performance of individual features and their combination is compared to the TF×IDF baseline and to the Kea++ algorithm. Next, Maui's domain-independence and language-independence are tested by applying it to medical and physics documents and to agricultural documents in French and Spanish. Finally, we provide a direct comparison with human indexers employed by the FAO specifically to assign term to agricultural documents.

7.2.1 Combining the features

Using the agricultural FAO-780 corpus described in Section 4.1.1, a 10-fold cross validation experiment was performed. In each run Maui was trained on 702 documents and tested on the remaining 78. FAO's indexers assign on average

eight terms to each document from the Agrovoc thesaurus. These were compared to the top eight terms that Maui derived from the same vocabulary.

Table 7.1 begins with the TF×IDF baseline (row 1) and the TF×IDF feature, used in conjunction with the two classifiers Naïve Bayes and bagged decision trees (row 2). Naïve Bayes (left) performs better than pure TF×IDF ranking by performing discretization of feature values (Section 6.4.1) and deriving additional information about typical TF×IDF values from the training data. Bagged decision trees (right) perform much worse: they do not handle a single feature well due to pruning.

The following rows (3–12) show the effect of adding successive features, combined with either Naïve Bayes or bagged decision trees. Each row uses the features in previous rows plus the new feature or features (e.g. both last occurrence and spread are deemed "occurrence features" and added in row 7). The order is determined by when the feature was first published in keyphrase extraction research literature, starting from old, commonly-used features like TF×IDF and first occurrence (e.g. Turney, 1999; Witten *et al.*, 1999) and ending with the new Wikipedia-based features proposed in this thesis.

Up until row 6 in Table 7.1, Naïve Bayes integrates new information added by the new features in a way that improves the performance, achieving a maximum F-measure of 32% with TF×IDF, first occurrence, keyphraseness (all used in Kea) and node degree, length (introduced in Kea++). With these features bagged decision trees perform slightly better than Naïve Bayes, with an F-measure of 33%. These results represent the best-performing combination of non-Wikipedia-based features on this collection.

Adding further features decreases the performance of Naïve Bayes. Results for bagged decision trees vary: they improve substantially only when Wikipedia keyphraseness (row 9) and inverse Wikipedia frequency (row 11) are included. Both represent general background knowledge concerning the likelihood that certain phrases and concepts are used in explanations and definitions (Section 5.2.3). Combining all features with bagged decision trees yields an F-measure of 34.2% (row 12), whereas Naïve Bayes' performance drops back to the baseline. This dem-

		P	R	F	Difference
1	All features	32.9	35.6	34.2	
2	- Keyphraseness	29.1	31.9	30.4	-3.8
3	- Wikipedia keyphraseness	32.7	35.2	33.9	-0.3
4	- Generality	32.7	35.5	34.0	-0.2
5	- First occurrence	33.1	35.1	34.1	-0.2
6	- Inverse Wikip. frequency	32.8	35.6	34.1	-0.1
7	- TF×IDF	33.0	35.8	34.4	-0.1
8	- IDF	32.9	35.6	34.2	0.0
9	- Length	33.1	35.6	34.3	+0.1
10	- Total Wikip. keyphraseness	33.4	36.2	34.8	+0.5
11	- Spread	33.3	36.2	34.7	+0.5
12	- Last occurrence	33.4	36.1	34.7	+0.5
13	- Node degree	34.0	36.7	35.3	+1.1
14	- Term frequency	34.6	37.4	36.0	+1.8
15	Best performing features only	**37.2**	**39.9**	**38.5**	**+4.3**

Table 7.2 Feature elimination with bagged decision trees on FAO-780

onstrates that the Naïve Bayes classifier does not handle feature dependencies well: they require a more powerful classifier like bagged decision trees.

Because performance fluctuates with the feature set, it is instructive to perform feature elimination. All features are first combined using bagged decision trees, and then each is eliminated in turn. Table 7.2 ranks features based on their contribution to the all-feature combination (row 1). The results show interesting dependencies. The contribution of the length feature is 3 percentage points when it is combined with TF×IDF, first occurrence and node degree using bagged decision trees (Table 7.1, row 4), but it is counter-productive when other features are added (Table 7.2, row 9). More surprisingly, eliminating strong features like term frequency and node degree that individually exhibit ROC values of 0.841 and 0.799 respectively (Section 5.2.6) improves the F-measure by 1.8 and 1.1 percentage points respectively. The damage they cause is due to overfitting: spurious fluctuations in the data cause inferior features to be incorporated in the trees.

Row 15 in Table 7.2 shows results for a new setting that excludes all those features whose elimination improved for the overall result. Here, Maui combines the three features used in the original Kea (Frank *et al.*, 1999)—TF×IDF, first occurrence and keyphraseness—and three Wikipedia-based features introduced in this

		P	R	F
Kea++'s candidate generation & Naïve Bayes				
1	Kea's two features (Witten et al., 1999)	20.5	19.7	18.7
2	Kea's three features (Frank *et al.*, 1999)	n/a	n/a	n/a
3	Kea++'s four features (Medelyan, 2005)	25.3	23.5	22.6
Maui's candidate generation & Naïve Bayes				
4	Kea's two features	23.2	25.7	24.4
5	Kea's three features	27.2	29.3	28.2
6	Kea++'s four features	25.5	28.0	26.7
Maui's candidate generation & bagged decision trees				
7	Kea's two features	22.6	25.0	23.7
8	Kea's three features	27.6	29.7	28.6
9	Kea++'s four features	28.2	31.2	29.6
10	Maui's all features	32.9	35.6	34.2
11	Maui's best feature combination	37.2	39.9	38.5

Table 7.3 Comparison of Kea, Kea++ and Maui on FAO-780

thesis—Wikipedia keyphraseness, inverse Wikipedia frequency and generality. This combination yields the best results: an F-measure of **38.5**%, which improves over the combination of all features (Table 7.2, row 1) by 4.5 points—and a total improvement of 21.7 points over the TF×IDF baseline (Table 7.1, row 1). However, it should be noted that this estimate is likely to be overoptimistic because the test data was used to determine which features to choose.

7.2.2 Improvement over Kea and Kea++

The same document collection (FAO-780) was used in (Medelyan, 2005). Table 7.3 repeats Kea++'s results reported there (rows 1–3) and compares them to new results achieved by Maui (rows 4–9).

Row 1 combines the two features used for keyphrase extraction—TF×IDF and position of first occurrence (Witten *et al.*, 1999)—that Kea++ applied to term assignment. The results for the keyphraseness feature used in a different experiment with Kea (Frank *et al.*, 1999), were not reported for Kea++. Row 3 adds node degree and term length, improving the F-measure from 18.7% to 22.6%.

Maui's generates candidates differently from Kea++ in that it handles multiple senses per vocabulary term (see Section 5.1). Re-evaluating the same features with

this new candidate generation method (rows 4 and 6) shows that it is more accurate: the same features yield an F-measure of 26.7% vs. the original 22.6%.

The three features proposed in Frank *et al.* (1999)—TF×IDF, first occurrence and domain keyphraseness—perform slightly better (row 5) than Kea++'s four features (row 6) when Naïve Bayes is used, whereas the opposite picture is observed for bagged decision trees (rows 8 and 9). Combining these features into a single filtering approach produces the best-performing combination of non-Wikipedia-based features and yields an F-measure of 32% and 33% for each classifier (rows 6 in Table 7.1). The new features proposed in this thesis give an additional improvement. Compared to Kea++, Maui improves the F-measure very substantially, from 22.6% to 34.2% (row 2 vs. row 10). Even better is the figure of 38.5% in row 11, but we do not use it for comparison because of the caveat noted at the end of the last subsection.

7.2.3 Domain independence

As noted earlier, Maui can be used with any controlled vocabulary in SKOS format. Many vocabularies are freely available in this format (see Appendix G). This section evaluates Maui's performance on medical documents indexed with MeSH terms (NLM-500) and physics documents indexed with HEP terms (CERN-290). Both collections and vocabularies were described in Section 4.1.1.

In both cases 10-fold cross-validation was applied, so the training was performed on 450 and 261 documents respectively. The precision, recall and F-measure values in Table 7.4 were averaged over all documents and test runs. For the evaluation the top 10 and top 25 terms are used in order to provide a basis for comparison with other systems. The results are compared to several baselines:

- TF×IDF;
- filtering using TF×IDF and first occurrence features (Witten *et al.*, 1999);
- filtering using a third feature, keyphraseness (Frank *et al.*, 1999);
- filtering using Kea++'s four features (Medelyan, 2005): TF×IDF, position of the first occurrence, node degree and term length;
- Medical Text Indexer developed specifically for indexing documents in the medical domain (Gay *et al.*, 2005);

		NLM 500			CERN-290		
		P	R	F	P	R	F
1	TF×IDF baseline	13.8	10.9	12.2	5.2	7.6	6.2
Naïve Bayes							
2	TF×IDF & First occurrence	22.4	18.0	19.9	11.0	16.0	13.0
3	as above & Keyphraseness	49.9	37.5	42.8	27.3	38.7	32.0
4	Kea++ (four features)	26.3	20.9	23.3	26.0	36.9	30.5
5	Maui (all features)	41.8	32.0	36.3	25.0	35.0	29.2
Bagging decision trees							
6	Kea++ (four features)	29.8	23.2	26.1	29.5	41.7	34.5
7	Maui (non-Wik. Features)	52.0	39.1	44.6	36.2	51.0	42.4
8	Maui (all features)	**55.4**	41.7	**47.6**	**38.4**	54.3	**45.0**
9	Maui (all features), *top 25*	32.2	58.6	41.6	22.1	**75.3**	34.2
Competitors							
10	Medical Text Indexer	31.0	**60.0**	40.9			
11	BibClassify				15.4	24.3	18.8

Table 7.4 Performance of Maui and competitors on NLM-500 and CERN-290

- the BibClassify algorithm developed for physics domain (Pepe and Yeomans, 2008).

Additionally, two classifiers, Naïve Bayes and bagged decision trees, are compared on features used in Kea++ and Maui.

Assigning MeSH terms to medical documents

Table 7.4 summarizes the results for both collections. For NLM-500, the TF×IDF baseline performs relatively poorly, achieving an F-measure of only 12.2% (row 1). Adding the first occurrence and keyphraseness features increases this to 19.9% (row 2) and 42.8%. Incorporating keyphraseness makes filtering twice as accurate, demonstrating that many of NLM-500's topics tend to repeat. Kea++ does not use the keyphraseness feature, but improves over Kea's original two features by 3.4 points (row 2 vs. row 4).

Combining all Maui's features with the Naïve Bayes classifier does not outperform the three features used by Frank *et al.* (1999) (row 3 vs. row 5). However, changing the classifier to bagged decision trees improves Maui's performance to an F-measure of **47.6%**, the maximum achieved in these experiments (row 8). To quantify the advantage of using Wikipedia-based features, they were eliminated

and the results re-evaluated. The F-measure dropped by 3 percentage points to 44.6% (row 7 vs. row 8), which shows that Wikipedia's contribution is substantial.

It is interesting to compare Maui's performance (55.4% precision and 41.7% recall) with that of other systems that were specifically developed for indexing with medical terms. Gay *et al.* (2005) use the NLM-500 documents to test their Medical Text Indexer (Section 3.1.2) and report precision of 31% and recall of 60% on the top 25 terms. For comparison, Maui's results computed for the top 25 topics are shown in Table 7.4, row 9. Maui's recall is slightly lower, while its precision is slightly higher. The F-measure comparison (row 9 vs. row 10) shows that the systems perform equally well. However, Medical Text Indexer was trained on the entire PubMed—millions of manually indexed documents—whereas Maui was trained on only 450 documents. It is likely that Maui's performance will further improve with larger training sets (see Section 7.2.6).

Assigning HEP terms to physics document

The right-hand part of Table 7.4 evaluates Maui's performance on physics documents with the baselines. A similar pattern emerges. The TF×IDF baseline is the weakest, and combining it with first occurrence, keyphraseness, node degree and length improves the results. In particular, node degree and length help a lot. However, the keyphraseness feature is not as strong on this collection as on NLM-500: presumably topics do not repeat so much. Maui combines further features using bagged decision trees and improves indexing performance to an F-measure of **45**% (row 8), compared with Kea++'s 30.5% (row 4). The Wikipedia-based features contribute an F-measure improvement of 2.6 points (row 7 vs. row 8), similar to the improvement for the NLM-500 corpus.

The results can be compared to an existing system for assigning terms from the HEP thesaurus. BibClassify is a module of CDS Invenio, a digital library system developed at CERN, and was developed specifically for topic indexing on physics documents. It combines term frequency statistics with the compound relation defined in the HEP thesaurus. BibClassify achieved an F-measure of 18.8% on 280

out of the original 290 documents[1] (Table 7.4, row 11). It performs better than the TF×IDF baseline (F-measure of 6.9%, row 1) and filtering using the original two features proposed by Witten *et al.* (1999), TF×IDF and position of the first occurrence (F-measure of 13%, row 2). However, it is completely outclassed by both Kea++ and Maui, which achieve F-measures of 30.5% and 45% respectively. Note, however, that bibClassify does not require any training data. Its results could also be improved using semantic conflation—currently it does not replace non-descriptors that appear in the document by their preferred labels.

Despite not being modified in any way before being applied to these data sets, Maui outperforms systems developed specifically for these domains—even ones trained on thousands of documents. Interestingly, its results here are even better than those obtained in the agricultural domain (Section 4.2.1) on which it was developed. Appendix D.2 shows Maui's topics for one example document from each collection.

7.2.4 Language independence

Another experiment was performed on the collections of 67 French and 47 Spanish agricultural documents described in Section 4.1.1. Prior to applying Maui the following modifications were made:

- The stemmer was set to French and Spanish stemmers, respectively.[2]
- The stopword list was set to the one in the corresponding language.
- The encoding was set to UTF-8, which was the encoding of the documents.
- The vocabulary language was set to *fr* and *es*, respectively, corresponding to Agrovoc's language tags.

Because both collections are small, the leave-one-out technique was applied to create a model from the maximum possible number of documents: 66 and 46 respectively. The model was tested on the remaining document, and the procedure

[1] Format conversion errors prevented bibClassify from producing results for the remaining ten documents. Kea's and Maui's performance on these 280 documents was within 1.5% of that for the full document set.

[2] These stemmers are explained at *http://snowball.tartarus.org/*, a website dedicated to stemming in different languages maintained by Martin Porter, the author of the English stemmer used in this thesis (Porter, 1980).

		French			Spanish		
		P	R	F	P	R	F
1	TF×IDF baseline	13.3	13.8	13.5	15.7	16.9	16.3
2	Kea++ (four features)	31.2	29.7	30.4	20.0	22.3	21.1
3	Maui (all features)	**34.5**	**31.8**	**33.1**	**24.7**	**26.9**	**25.7**

Table 7.5 Performance on French and Spanish documents

was repeated 67 and 47 times until Maui had assigned topics to every document. For each document the top ten terms were extracted, which was the average number of manually-assigned terms in each collection.

Table 7.5 compares Maui's performance on these sets with the baseline; the results can be compared to those for English documents in Tables 7.1 and 7.3. The performance of the baseline (row 1) in both collections is better than for English: the F-measure is 13.5% for French and 16.3% for Spanish documents versus 12.2% for English documents (Table 7.1, row 1). Row 2 in Table 7.8 shows the performance of Kea++'s four features: TF×IDF, first occurrence, node degree and term length. The F-measure of 30.4% for the French collection is better than that for the English collection (26.7%, row 4 in Table 7.3), whereas the Spanish results are weaker (F-measure of 21.1%). Maui, with its larger set of features and bagged decision trees, outperforms Kea++ on both collections, increasing the F-measure from 30.4% to **33.1**% on French and from 21.1% to **25.7**% on Spanish documents (row 3 in Table 7.8). The French results are comparable to those for English documents given the same settings but a larger training set (F-measure of 34.2%, Table 7.3, row 6), whereas the Spanish ones are worse. Appendix D.3 shows example topics extracted for sample documents in French and Spanish.

Examination of the manually assigned topics shows that on average only 66% of the Spanish ones actually appear in the documents. Therefore Maui cannot possibly achieve greater recall than this. This partially explains why recall is significantly lower than on English and French documents, where 81% and 74% of keyphrases respectively appear in the document text.

Because Wikipedia Miner is not yet available in languages other than English, the feature computation was restricted. Agrovoc descriptors in each language were still matched to anchors and article titles in the English Wikipedia, which in some

		Inter-indexer consistency		
		minimum	average	maximum
1	FAO's professional indexers	**26.0**	**38.7**	**46.0**
2	TF×IDF baseline	15.3	18.8	21.5
3	TF×IDF & First occurrence	16.5	19.7	23.1
4	Kea++ (four features & Naïve Bayes)	21.3	26.8	30.2
5	as above & keyphraseness	22.8	27.6	32.9
6	Maui (all features & bagged decision trees)	22.8	27.0	31.8
7	Maui (best features & bagged decision trees)	**26.0**	29.6	34.2

Table 7.6 Consistency with professional indexers on FAO-30

cases (especially named entities) yielded successful mappings. But in many cases no mapping was found. Thus the full potential of Wikipedia-based features for topic indexing in foreign languages cannot yet be explored.

7.2.5 Consistency with professional indexers

In the FAO-30 collection six FAO indexers independently assigned terms to 30 agricultural documents. This collection was used to determine inter-indexer consistency when people perform term assignment (Section 4.1.2). This section compares these indexers' performance with Maui's.

The first evaluation was performed with the leave-one-out method: training on 29 documents and testing on the remaining one. Table 7.6 gives the results. Human performance ranges from 26% to 46% with an average of 38.7% (row 1), also shown in able 4.4. The TF×IDF baseline is approximately half as good, with average consistency 18.8% (row 2). Adding the Kea and Kea++ features and combining them using Naïve Bayes successively improves consistency up to 27.6% (row 5). Interestingly, whereas keyphraseness significantly improved results for other collections, here it increases performance by less than one percent (row 4 vs. row 5). It seems that FAO-30 contains documents from different areas of agriculture, and topics do not repeat as much as in other collections. Its small size is not by itself the reason: the contribution of keyphraseness is more substantial on the even smaller WIKI-20 collection (Section 7.3.1).

Combining all features proposed in this thesis using bagged decision trees does not improve Maui's performance over Kea++. The average consistency Maui achieves is 27% (row 6). However, using the best combination of features deter-

Indexers	1	2	3	4	5	6	Average	Maui
1		45	42	46	40	46	43.8	33.2
2	45		35	36	43	34	38.6	34.2
3	42	35		40	26	34	37.1	26.8
4	46	36	40		37	35	38.9	28.2
5	40	43	26	37		33	35.3	29.3
6	46	34	34	35	33		38.5	26.0
						Overall	**38.7**	29.6

Table 7.7 Inter-indexer consistency comparison of professional indexers and Maui

mined on the FAO-780 corpus resulted in an average consistency with professionals of **29.6**% (row 7), which is a sound estimate because the optimization of performance was performed using a different corpus.

Table 7.7 compares Maui's consistency values directly with those of professional indexers. Maui agrees with Indexers 3 and 5 more than they agree with each other (26.8% and 29.3% vs. 26%); it also agrees as much with Indexers 2 and 6 as they do with each other (34%).

Inter-indexer consistency is computed per document in Table 7.8. The "Indexers" column gives the average consistency of the six professional indexers with each other, while the "Maui" column shows their consistency with Maui on the same document. The final column shows the difference between the two, and indicates how much worse (or better) Maui performed than to the humans.

Per-document consistency for indexers varies from 23.4% to 55.5%, most (14 documents) lying in the 35–45% range. Maui's values are lower: from 7.4% to 50.8%, most (12 documents) grouped around the 25-35% range. On four documents Maui's consistency is below 20% On seven (shown in bold), it agreed with the indexers on the main topics more than they agree with each other. In additional five cases, the difference lies within 5 percentage points.

Maui's performance is clearly weaker than the average achieved by the professionals, but the above analysis shows that

- in several cases Maui agrees with an indexer more than he or she does with a colleague;

Document rank	Indexers	Maui	Difference
1	40.3	50.8	**–10.5**
2	52.3	43.0	9.3
3	42.3	40.5	1.8
4	40.2	39.2	1.0
5	35.7	38.3	**–2.6**
6	55.5	37.7	17.8
7	35.9	37.5	**–1.7**
8	44.6	35.4	9.1
9	28.8	35.4	**–6.6**
10	51.9	33.6	18.3
11	27.7	32.7	**–5.0**
12	35.3	31.8	3.5
13	43.2	31.0	12.2
14	53.0	30.1	22.9
15	34.7	29.3	5.4
16	44.5	29.2	15.3
17	42.3	28.0	14.3
18	39.4	27.4	12.0
19	29.3	26.9	2.4
20	35.4	26.8	8.6
21	25.2	25.6	**–0.4**
22	53.7	24.8	28.9
23	23.4	24.1	**–0.6**
24	37.3	23.6	13.8
25	30.9	21.8	9.0
26	33.7	20.8	13.0
27	24.9	17.9	7.0
28	43.7	15.4	28.4
29	48.3	13.5	34.8
30	34.7	7.4	27.3
Average	38.9	29.3	9.5

Table 7.8 Per-document consistency on FAO-30

- for approximately one-third of the documents, Maui's consistency with the indexers is nearly as high or higher as their consistency with each other.

7.2.6 Effect of training set size

The FAO-780 collection originates from the same organization and was used for training in an experiment with FAO-30. Because 780 documents are available, we can investigate how the size of the training set affects the results. Frank *et al.* (1999) report that keyphrase extraction performance improves steadily up to 20 documents, makes smaller gains until 50 documents, and changes little thereafter. Their experiment was performed without using the keyphraseness feature.

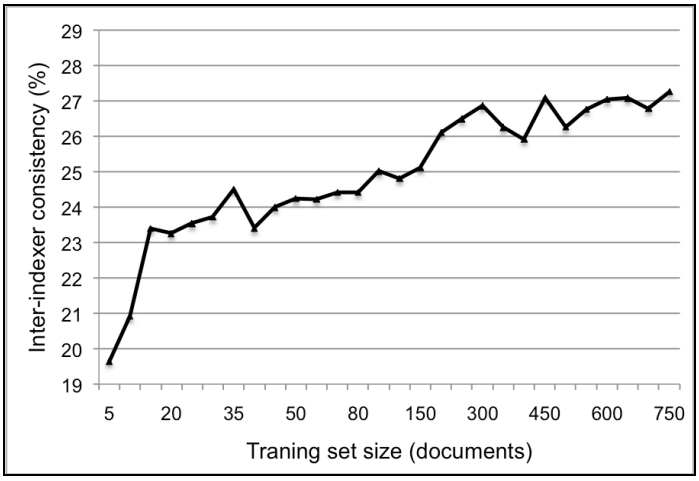


Figure 7.1 Effect of training set size on performance

We take ten different-sized subsets of training documents containing from 5 to 750 documents, use them for training, and report the average consistency on the FAO-30 corpus for each size. For simplicity, all features and bagged decision trees were used. Figure 7.1 shows the effect of increasing training data on the consistency that Maui achieves with humans. A steady increase in both precision and recall can be observed up to 300 documents. Beyond that, performance is erratic: values vary between 26% and 27%, with a very slow growth.

Training on 29 documents during the leave-one-out runs using FAO-30 gives a consistency of 27%, whereas training on 30 FAO-780 documents yields only around 24% and 400 documents are required to approach 27%. This is due to several factors. First, the joint topic sets in FAO-30 assigned by six people contain up to six times more positive examples than the single topic sets in FAO-780. Second, FAO-30 is more heterogeneous in both length and topics because it contains web pages from a variety of sources rather than FAO's own documents. We conclude that when creating a training collection for a particular set, it is better to provide multiple judgments from several people for each document, but if that is not possible, quantity can substitute for quality.

7.2.7 Examples and error analysis

Analysis of multiply indexed documents supplied by professional indexers at FAO provides insight into the quality of Maui's term assignment. Appendix D.4 compares topics that most indexers agree on with those from Maui's leave-one-out

Excellent performance. Maui 51% vs. Indexers 40%
Doc 1. The dynamics of sanitary and technical requirement assisting the poor

Topics by 6 professionals indexers	Topics assigned by Maui
Food safety (5)	Food safety (5)
Livestock (5)	Livestock (5)
Standards (5)	Standards (5)
Poverty (5)	Developing countries (4)
Developing countries (4)	Food chains (4)
Food chains (4)	Animal health (2)
Phytosanitary measures (4)	FAO (2)
Animal production (2)	Risk management (2)

Average performance. Maui 29% vs. Indexers 35%
Document 15. Climate change and the forest sector

Topics by 6 professionals indexers	Topics assigned by Maui
Climatic change (6)	Climatic change (6)
International agreements (5)	Forests (4)
Forests (4)	Greenhouse gases (3)
Greenhouse effect (4)	Forest management (3)
Legislation (4)	Property (0)
Forestry policies (4)	Climate (0)
Pollution control (4)	Land use (0)
Greenhouse gases (3)	Forest products (0)

Poor performance. Maui 7% vs. Indexers 35%
Document 30. Phosphorus limitation of microbial processes in tropical forests

Topics by 6 professionals indexers	Topics assigned by Maui
Tropical rain forests (6)	Carbon (2)
Phosphorus (6)	Costa rica (2)
Soil chemicophysical properties (3)	Tropical forests (0)
Soil fertility (3)	Respiration (0)
Soil microorganisms (3)	Rain forests (0)
Soil biology (2)	Primary productivity (0)
Microorganisms (2)	Forests (0)
Biodegradation (2)	Soil (0)

Figure 7.2 Example topics assigned to FAO-30

predictions. Each document has several topics on which five or six indexers agree. In only 6 out of 30 cases does Maui fail to identify at least one of these topics, and in 4 of these 6 it assigns similar ones—*Tropical forests* and *Rain forests* instead of *Tropical rain forests* or *International trade* instead of *Import* and *Export*.

Figure 7.2 gives three example documents reflecting different levels of performance. The numbers in brackets indicate how many people assigned this topic to the

		Inter-indexer consistency		
		minimum	average	maximum
1	TF×IDF baseline	5.7	8.3	14.7
Naïve Bayes				
2	TF×IDF & First occurrence	9.1	16.0	23.8
3	as above & Keyphraseness	13.5	20.2	25.8
4	KEA++'s four features	15.5	22.6	27.3
5	Maui – all features	22.6	29.1	33.8
Bagging decision trees				
6	Maui (all features)	25.4	30.1	38.0
7	Maui (best features)	23.6	31.6	37.9
Competitor				
8	Grineva *et al.* (2009)	18.2	27.3	33.0

Table 7.9 Performance on WIKI-20

same document. In the first, Maui successfully identified three out of four topics on which most people agreed. It missed *Poverty*, but the term *Developing countries* partially covers this topic. In the second document, some of the important topics are identified while others are missed. Topics such as *Climate* and *Forest products* that were not chosen by indexers still relate to the document's overall theme. In the last document Maui failed to identify *Phosphorus*, despite it being chosen by all indexers. Such errors generally reflect the fact that the corresponding terms rarely or never appear in the document text. If they do, Maui usually is able to identify them within the top 25 ranked candidates.

7.3 Quality of indexing with Wikipedia

Using Wikipedia as a controlled vocabulary for topic indexing is a new idea introduced in this thesis. A new collection, WIKI-20 (discussed in Section 4.2.2), was created in which 20 documents were indexed independently by 15 teams of graduate students. Here we compare how Maui performs on the same documents with human indexers. In each setting the leave-one-out technique was applied, so the training set contained 19 documents, and Maui's five top scoring terms were matched against those assigned by the teams.

7.3.1 Consistency with graduate students

Table 7.9 lists inter-indexer consistency under several experimental conditions. In each case, consistency was computed by matching automatically computed topics

	Inter-indexer consistency			Difference
	minimum	average	maximum	to average
All features	25.4	30.1	68.0	
– First occurrence	22.0	27.8	33.9	–2.3
– Generality	23.3	28.0	34.0	–2.1
– Length	19.9	28.3	32.3	–1.8
– Wikipedia keyphraseness	23.2	28.3	35.1	–1.8
– Keyphraseness	24.2	28.4	35.0	–1.7
– Inverse Wikip. frequency	23.7	28.7	33.1	–1.4
– Semantic relatedness	23.8	28.7	36.9	–1.4
– Node degree	25.1	28.9	37.0	–1.2
– Last occurrence	24.2	29.1	36.0	–1.0
– Spread	24.6	29.6	35.0	–0.5
– Term frequency	25.4	29.7	37.0	–0.4
– TF×IDF	25.4	29.7	37.0	–0.4
– IDF	25.4	30.1	38.0	0.0
– Total Wikip. keyphraseness	23.6	**31.6**	37.9	**+1.5**

Table 7.10 Feature elimination on WIKI-20

to those assigned by the 15 teams. The average consistency of the TF×IDF baseline (row 1) is quite low, at 8.3%, but improves rapidly as other features are added. Position of the first occurrence doubles consistency to 16% (row 2); adding keyphraseness increases it to 20.2% (row 3). Kea++ adds node degree and term length, but not keyphraseness, and improves the consistency by 6.6 percentage points (row 2 vs. row 4). Finally, Maui combines all features proposed in this thesis and achieves an average consistency of 29.1% with Naïve Bayes (row 5) and a slightly better value of 30.1% (row 6) with bagged decision trees.

Table 7.10 ranks the features in terms of Maui's performance after removing each one individually and re-evaluating consistency. The three strongest features are *first occurrence*, *generality* and *length*. It is not surprising that keyphraseness is not among the strongest, as it was on the FAO-780 corpus (Section 7.2.1), because the WIKI-20 training set contains only 19 documents in each run. The total Wikipedia keyphraseness feature repeats information already captured by Wikipedia keyphraseness and term frequency, and excluding it improves the result by 1.5 percentage points, achieving the overall consistency of 31.6%. This figure is higher than the average consistency of the human teams with each other, namely 30.5%.

Team rank	Native speaker?	Year	Inter-indexer consistency	
			with other teams	with Maui
1	**yes**	**4**	**37.1**	32.9
2	**mixed**	**4**	**35.5**	33.6
3	**yes**	**4**	**33.8**	28.9
4	mixed	3	32.4	37.9
5	**yes**	**4**	**31.6**	29.4
6	yes	3.5	31.6	31.3
7	**yes**	**4**	**31.6**	31.4
8	no	3	31.2	33.7
9	yes	3	31	34.4
10	mixed	3.5	30.8	30.2
11	**yes**	**4**	**30.2**	29.2
12	no	2.5	28.7	34.7
13	no	4	26.2	26.7
14	no	1	24.1	35.5
15	no	4.5	21.4	23.6
		Overall	30.5	31.6

Table 7.11 Inter-indexer consistency comparison of student teams and Maui

The last row of Table 7.9 shows the result of the topic indexing system described by Grineva *et al.* (2009). It achieves an average consistency with humans of 27.3%, which is 4.3% lower than that of Maui. The results are still impressive, because their approach is unsupervised and does not require training. Interestingly, the consistency between Grineva *et al.*'s algorithm and Maui is the highest: 42.9%. This is likely due to the fact that both systems derive candidate topics from document text, whereas human indexers are able to abstract from the textual content and find relevant Wikipedia articles regardless of whether they are explicitly mentioned in the document.

Table 7.11 compares the average inter-indexer consistency for each of the 15 student teams, as discussed in Section 4.2.2, with the consistency of Maui. The six teams containing senior students and at least one native speaker are shown in bold. Three of these teams outperform Maui, while two others exhibit the same average consistency and one performs slightly worse than Maui. All these teams outperform Grineva *et al.*'s algorithm, which at 27.3% would rank between the 12th and 13th student team, whereas at 31.6% Maui ranks 5th.

Table 7.12 compares Maui's consistency with human indexers on a per-document basis. The comparison is more favorable than the one with professional

Document	Indexers	Maui	Difference
12049	41.1	52.1	**–10.9**
7183	46.8	48.5	**–1.7**
43032	28.4	43.9	**–15.5**
7502	20.4	41.4	**–20.9**
20782	37.6	41.3	**–3.6**
18209	39.0	40.6	**–1.6**
39955	31.6	39.2	**–7.6**
287	41.4	39.1	2.4
39172	29.1	35.7	**–6.6**
19970	31.3	34.6	**–3.2**
40879	31.1	31.7	**–0.6**
10894	52.8	29.7	23.1
9307	26.6	29.0	**–2.5**
23267	27.7	27.7	0.0
23507	28.9	24.4	4.5
23596	22.6	23.5	**–0.9**
37632	24.1	22.9	1.1
13259	17.3	17.5	**–0.2**
25473	15.3	10.1	5.2
16393	37.5	9.4	28.0
Average	31.5	32.1	–0.6

Table 7.12 Per-document consistency on WIKI-20

indexers (Table 7.8). Student teams outperform Maui by a large margin on only two documents (10894 and 16393). On the remaining 18, Maui performs better than (the bold-face difference values) or similar to human indexers. We conclude that Maui performs at least as well as human indexers on this task.

7.3.2 Examples and error analysis

Multiply-indexed documents provide a good basis for comparing Maui with human indexers. Ideally, the algorithm should assign those topics on which most humans agree, and the inter-indexer consistency measure quantifies the agreement. However, it is instructive to analyze cases where Maui missed a highly relevant topic or assigned an incorrect one.

For each of the 20 documents Appendix 5 lists the top five most frequent topics as identified by the 15 teams and the five topics assigned by Maui. In 15 cases Maui assigned the topic most frequently chosen by the teams. In only four cases did it assign more than one topic that was not picked by any of the teams.

Most frequent topics by 15 teams	Topics assigned by Maui
Excellent performance. Maui 52.1% vs. Indexers 42.1%	
12049. Occam's razor: The cutting edge for parser technology	
Yacc (13)	Yacc (13)
Parsing (12)	Parsing (12)
Compiler-compiler (9)	Compiler-compiler (9)
Backus Naur form (9)	Compiler (6)
Compiler (6)	Programming language (4)
Average performance. Maui 29.7% vs. Indexers 52.8%	
10894. A Safe, Efficient Regression Test Selection Technique	
Regression testing (15)	Software maintenance (13)
Software maintenance (13)	Algorithm (7)
Control flow graph (10)	Test suite (2)
Software testing (9)	Computer software (1)
Algorithm (7)	Control flow (0)
Poor performance. Maui 17.5 vs. Indexers 17.3	
13259. Cone trees in the UGA graphics system	
Hierarchical model (7)	Computer graphics (3)
3D computer graphics (7)	Visualization (2)
Visualization (graphic) (6)	PARC (company) (2)
Tree (data structure) (5)	Visual display unit (0)
Computer graphics (3)	Graphics (0)

Figure 7.3 Example topics assigned to WIKI-20

Figure 7.3 shows three examples from Appendix D.5, each demonstrating a different level of performance. In the first (document 12049), Maui identified four out of five most frequent topics, and the fifth one (*Programming language*) was chosen by four of the 15 teams. This indicates excellent performance—and Appendix D.5 contains many more similar examples. The second example (document 10894) is one of the two on which the teams outperformed Maui by a large margin. It is obvious why: Maui failed to extract *Regression testing*, agreed upon by all teams, among the top five terms. Closer analysis shows that *Regression testing* received rank 7, so it was not completely missed. The second most important term, *Software maintenance*, was identified, whereas for the third most important, *Control flow graph*, Maui chose a related term *Control flow*.

The third example (document 13259) is chosen from the bottom three documents, on which Maui performed the worst. Three of the terms match topics on which at least two teams have agreed (*Computer graphics, Visualization and PARC*

	129 mixed documents			86 blog posts		
	P	R	F	P	R	F
TF×IDF baseline	18.3	33.4	23.7	12.4	27.3	17.0
Maui	44.2	46.6	45.4	41.6	46.7	44.0
Grineva *et al.* (2009)	34.6	36.4	35.5	39.0	38.1	38.5

Table 7.13 Performance on heterogeneous data sets

(company)), but important topics like *Hierarchical model* and *3D computer graphics* are omitted. Instead two topics not chosen by any other teams are added: *Graphics*, which is more generic than *3D computer graphics*, and *Visual display unit*, which is the same as *Computer display*, a topic chosen by one of the teams, but was not identified as such in Maui's version of Wikipedia. Interestingly, this document is the one on which teams disagreed the most: less than 50% of teams agreed on the most frequently chosen topics. The students seemed to struggle with this document as much as Maui did.

7.3.3 Performance on heterogeneous documents

Section 4.2.3 discussed two data sets with documents from heterogeneous sources, which are subsets of the original test set used by Grineva *et al.* (2009), who used them to demonstrate that their approach, described in Section 3.4.2, can handle noisy data. They report good results, which outperform several systems that they re-implemented for indexing with Wikipedia.

Table 7.13 summarizes the results of our experiments on the two data sets. Maui was trained using the leave-one-out method, on 128 and 86 documents in each case. It outperforms both the TF×IDF baseline and Grineva *et al.*'s algorithm, particularly on the larger collection, where more training documents were available. Maui has not been tweaked for these datasets in any way, but still achieves, on the larger collection, an F-measure of 45.4%, which is 10 percentage points better than the competitive system. Grineva *et al.*'s approach is unsupervised, which may explain its relatively poor performance.

7.4 Quality of automatic tagging

Now we apply Maui to a third topic indexing task: tagging using keyphrase extraction techniques. Here candidates are document phrases; no controlled vocabulary

		P	R	F
1	TF×IDF baseline	14.4	16.0	15.2
2	Brooks and Montanez, 2006 (single-word TF×IDF)	16.8	17.3	17.0
3	TF×IDF & First occurrence	20.4	22.3	21.3
4	– as above & Keyphraseness	41.1	43.1	42.1
5	Kea++ (four features & Naïve Bayes)	33.6	35.7	34.6
6	Maui (all features & Naïve Bayes)	37.6	39.6	38.6
7	Maui (all features & bagged decision trees)	**45.7**	**48.6**	**47.1**

Table 7.14 Performance on CiteULike-180

is used. Filtering uses the same strategy as in term assignment and indexing with Wikipedia. Automatic tagging can be seen as a keyphrase extraction task because the majority of tags assigned by humans match document phrases (Section 5.1.3).

This section evaluates Maui's performance on the collaboratively tagged collection CiteULike-180 (described in Section 4.3.1). Each document contains tags assigned by at least two human taggers, and the algorithm's goal is to match these tags. Again, 10-fold cross validation is applied: in each run Maui is trained on 162 documents and tested on the remaining 18. The top five extracted tags are compared to those assigned by the taggers and are considered correct if they match a tag assigned by at least two users. As before, precision, recall and F-measure are computed, and inter-indexer consistency analysis is applied in order to compare Maui's performance directly with that of people.

7.4.1 Evaluation of assigned tags

As well as the TF×IDF baseline used in previous experiments, Maui is compared with Brooks and Montanez's (2006) automatic tagging method, which is also based on ranking TF×IDF values. Here only single words are allowed—whereas for the TF×IDF baseline candidates are multi-word phrases. The first two rows of Table 7.14 show that using multi-word phrases is less accurate than using single words, which gives an overall F-measure of 17%. Multi-words have higher TF×IDF values, but single words dominate the users' tags. As shown below, the length feature applied in Maui captures this characteristic without compromising the ability to assign correct multi-word tags.

Features	F-measure	Difference
All Features	47.1	
– Keyphraseness	30.2	–16.9
– Wikipedia keyphraseness	43.1	–4.0
– Length	45.0	–2.1
– Inverse Wikipedia linkage	45.1	–2.0
– Semantic relatedness	45.4	–1.7
– First occurrence	45.6	–1.5
– Total Wikipedia keyphraseness	45.8	–1.3
– Node degree	46.0	–1.1
– Spread	46.4	–0.7
– Generality	46.5	–0.7
– Last occurrence	46.6	–0.5
– TF×IDF	46.8	–0.3
– Term frequency	46.9	–0.2
– IDF	47.1	0.0
Non-Wikipedia features	41.7	–5.4

Table 7.15 Feature elimination on CiteULike-180

Adding a second feature—the position of the first occurrence—and using the Naïve Bayes model to learn conditional distributions improves the results by 5 percentage points (row 3). Adding the keyphraseness feature (row 4) nearly doubles the F-measure, from 21.3 to 42.1%. This shows that CiteULike users tend to re-assign existing tags.

The remainder of Table 7.14 compares further combinations of features. The four features used in Medelyan (2005)—TF×IDF, first occurrence, node degree and term length—yield an F-measure of 34.6%. They outperform Kea's original features (row 5 vs. row 3), but do not improve on TF×IDF and first occurrence used with keyphraseness (row 4). Combining all features with the features proposed in this thesis using the Naïve Bayes classifier yields an F-measure of 38.6% (row 6), and the results can be further improved to F-measure of **47.1**% (row 7) by employing bagged decision trees. This shows that Maui matches nearly half of all tags on which at least two human taggers have agreed.

Next, we eliminate features one by one and rank them by the resulting degeneration in performance. The difference reflects each feature's individual contribution to the overall result. Table 7.15 compares the performance using bagged decision trees. Surprisingly, TF×IDF, one of the strongest features, is the one that contributes the least when all features are combined, while the contribution of key-

		Consistency with other taggers			Consistency with Maui		
		min	average	max	min	average	max
1	332 taggers & 180 docs	3.2	18.5	92.3	0.0	23.8	80.0
2	36 taggers & 143 docs	7.9	37.6	71.4	11.5	35.0	56.0

Table 7.16 Inter-indexer consistency comparison of taggers and Maui

phraseness, the strongest feature, is, as expected, the highest, at 16.9 points. The second most important feature is Wikipedia keyphraseness, contributing 4 percentage points to the overall result.

Some of the features in the best performing combination rely on Wikipedia as a knowledge source. The last row of Table 7.15 excludes Wikipedia-based features, and the F-measure drops by 5.4 points. Therefore Wikipedia's contribution is significant.

7.4.2 Consistency with human taggers

Section 4.3.2 discussed the indexing consistency of taggers in the CiteULike-180 corpus and found that the consistency of the 332 taggers with each other is 18.5%. Given the tag sets obtained by Maui during the cross-validation, when all features and bagged decision trees are used (Table 7.10, row 7), we analyze Maui's inter-indexer consistency with each human user, based on whatever documents that user tagged. The results are averaged to obtain the overall consistency with all users.

Row 1 in Table 7.16 compares the consistency of 332 human taggers to Maui's consistency with these taggers. Maui's values range from 0 to 80%, with an average of **23.8**%—5 points higher than an average CiteULike tagger (18.5%). The only incidences of very low consistency are those where the human has assigned just a few tags per document (one to three), or exhibits some idiosyncratic tagging behavior (for example, one tagger prepended the word *key* to most tags).

In Section 4.3.2 we identified a small group whose consistency is above average and are most prolific. These 36 taggers tagged a total of 143 documents with an average consistency of **37.6**% (row 2 in Table 7.13). Maui's consistency with them ranges from 11.5% to 56%, with an average of 35%. This places it only 2.6 percentage points behind the average performance of the best CiteULike taggers. In fact, it outperforms 17 of them (cf. Table 4.7 in Section 4.3.2).

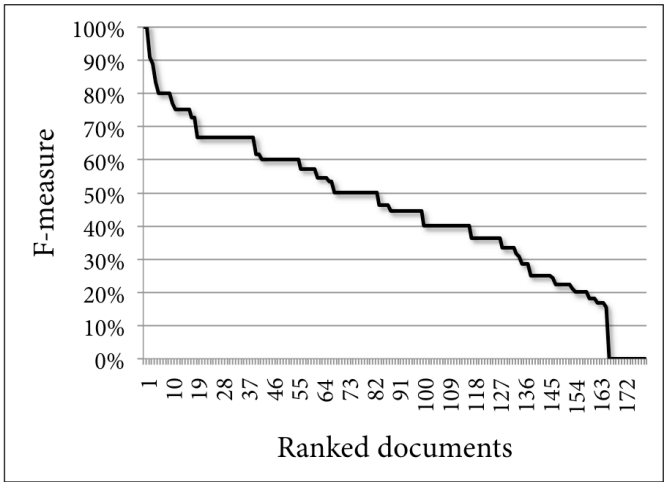


Figure 7.4 Distribution of F-measure on CiteULike-180

7.4.3 Examples and error analysis

Unlike FAO-30 and WIKI-20, in CiteULike-180 every tagger has assigned topics to a different set of documents. In many cases, judgments of only two taggers per document are available, so computing per-document consistency is not meaningful. Instead, Figure 7.4 plots a histogram of the F-measure values that Maui achieves across all documents, computed using as the gold standard tags agreed by two CiteUlike users. In 12 of the 180 documents none of the tags were matched, while in 10 an F-measure of at least 80% was achieved.

Figure 7.5 provides examples of excellent, average, and poor performance. It also shows the tags chosen by at least two taggers, which were used to compute the F-measure, and those chosen by other taggers, with their number in brackets. Boldface tags in Maui's column were assigned by at least two people; those matching a single tagger's choice are underscored. Judging by the titles, all Maui's tags—even those in the document with "poor" performance—relate to the document's content. For each document at least one of Maui's tags that was judged incorrect was picked by a human tagger (*basal ganglia*, *global* and *tropical*). In the document with "poor" performance, Maui picked tags that are more specific than human ones (e.g. *drought sensitivity* rather than *drought; tropical forests* rather than *vegetation*). The automatically assigned tags still capture all the main topics discussed in this document.

Excellent performance (F-measure **80%**)
101973. Different time courses of learning-related activity in the prefrontal cortex and striatum. Pasupathy and Miller (2005). *Nature* 433 (7028)

At least two taggers	One tagger	Maui
learning (5)	basalganglia, rt, dlpfc,	**learning**
striatum (4)	prefrontal, caudate, pfc,	**striatum**
monkey (3)	stimulusresponseassociation,	**monkey**
prefrontal cortex (2)	neurophysiology, dynamics,	**prefrontal cortex**
reversal (2)	striatothalamocortical	basal ganglia

Average performance (F-measure **46%**)
1322886. Global and regional drivers of accelerating CO2 emissions. Raupach et al. (2007). *Proceedings of National Academy of Sciences (PNAS)*

At least two taggers	One tagger	Maui
co2 (3)	global, scenarios,	**co2**
emissions (3)	sres, china	**emissions**
carbon (2)		**carbon**
ipcc (2)		global
economics (2)		energy
climate (2)		
projections (2)		
regional (2)		

Poor performance (F-measure **0%**)
1272477. Drought sensitivity shapes species distribution patterns in tropical forests. Engelbrecht et al. (2007). *Nature* 447 (7140)

At least two taggers	One tagger	Maui
precipitation (2)	tropical, tropic	tropical
drought (2)		tropical forests
ecology (2)		drought sensitivity
vegetation (2)		species
climate (2)		regional

Figure 7.5 Example tags assigned to CiteULike-180

7.4.4 Comparison with other autotagging approaches

The results of several previously published automatic tagging approaches, presented in detail in Section 3.3, can be compared indirectly with Maui's. For each paper, we compute Maui's results in settings closest to the reported ones.

Brooks and Montanez (2006) extract terms with the greatest TF×IDF values as tags for posts on *technorati.com*. They do not report their system's precision and recall, but re-implementing it and testing on the CiteULike-180 collection yielded precision of 16.8% and recall of 17.3% for the top five assigned tags. Maui's addi-

tional features and training improves these figures to 45.7% and 48.7% respectively.

Mishne (2006) uses TF×IDF-weighted terms as full-text queries to retrieve posts similar to the one being analyzed. Tags assigned to these posts are clustered and ranked heuristically to identify the best ones; tags assigned by the given user receive extra weight. Manual evaluation of the top ten tags assigned to 30 short articles results in precision and recall of 38% and 47% respectively. Automatic evaluation of Maui's top ten terms against tags assigned to 180 documents gave precision and recall of 44% and 29% respectively. Note that manual evaluation would likely give a far more favorable assessment, because human evaluators can assess conceptual correctness, which is always greater than terminological correctness (Section 2.2.6).

Chirita *et al.* (2007) use an unsupervised approach that scores candidate tags based on their term frequency and the position of the first occurrence. It yields a precision of 80% for the top four tags assigned to 30 large web pages (32 Kbytes), again evaluated manually. Maui achieves only slightly lower values (66%–80%) on CiteULike-180's documents (47 Kbytes on average) when evaluated automatically against user-assigned tags. (The above caveat regarding automatic and manual assessment applies here too.)

The only reported automatic evaluation of tags was found in Sood *et al.* (2007), where TagAssist was tested on 1000 blog posts. This algorithm is similar to Mishne's (2006), but uses centroid-based clustering. Exact matching of TagAssist's tags against existing ones yielded precision and recall of 13.1% and 22.8% respectively, substantially lower than Maui's 45.75% and 48.7% (Table 7.8).

These indirect comparisons do not reveal the true ranking of approaches, because the task definitions and test sets are different. It would be interesting to compare other systems on the multiple tagger set described in this thesis in order to more objectively assess the performance of these algorithms.

7.5 Summary

This chapter has investigated the quality of topics assigned automatically by Maui to several document collections, and shows that the two-stage approach to topic indexing based on candidate generation and filtering is applicable to several different tasks.

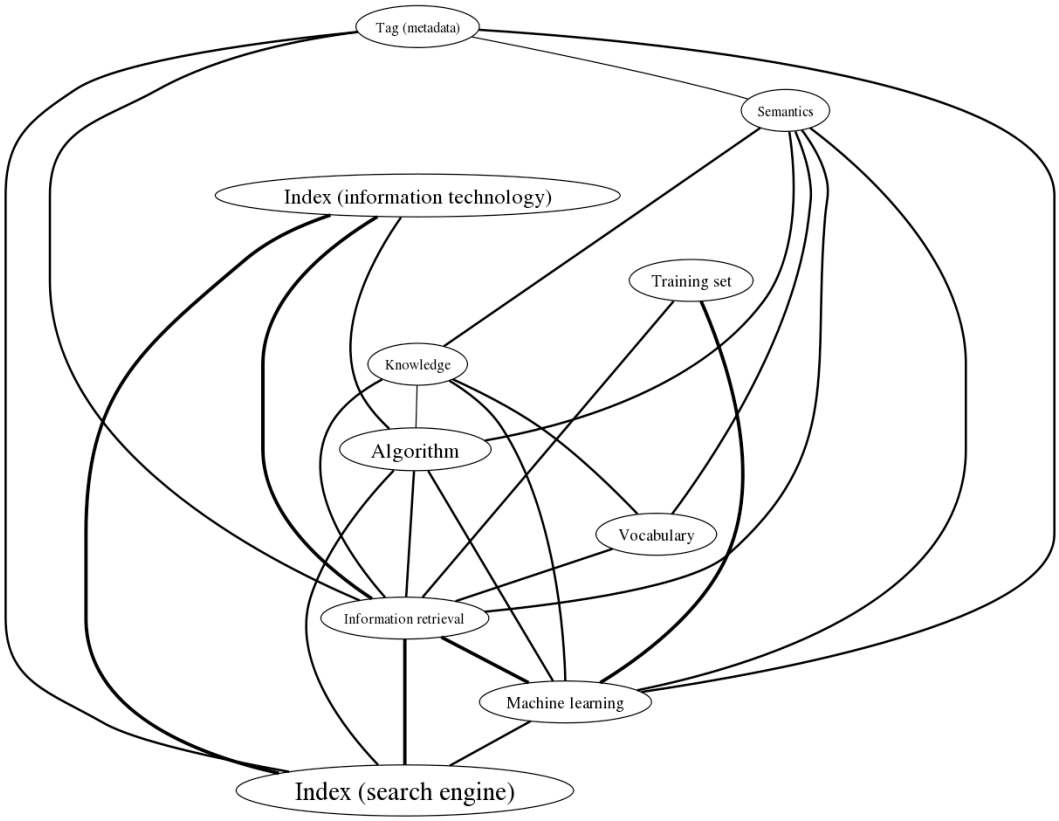
For term assignment, Maui was tested on agricultural documents in English, French and Spanish, and on medical and physics documents. On all collections, Maui outperformed the baseline by a large margin. Where a training set of over 50 documents was available, it yielded an impressive improvement over Kea++. On medical documents, Maui performed as well as special-purpose systems trained on thousands of documents. On physics documents, it produced a three-fold improvement over an unsupervised system developed specifically for that domain.

In topic indexing with Wikipedia, Maui proved robust against an expansion of the vocabulary by several orders of magnitude. Wikipedia's vocabulary is extremely large compared to the domain-specific thesauri used in term assignment, with over two million concepts and twice as many ways to refer to them. However, Maui was able to determine many of the topics agreed on by human indexers for the same documents.

Maui's keyphrase extraction ability has been successfully applied to tagging bioinformatics research papers voluntarily tagged by online users. Maui identified nearly half the tags on which at least two users agreed. The results exceed most of those reported for existing automatic tagging systems.

In each task, Maui's performance has been compared to that of human indexers. The consistency of professional indexers ranges from 26% to 46%; Maui's consistency with these indexers ranges from 26% to 34%. Although this is lower, Maui performs better than some individual indexers in terms of their consistency with their colleagues. On other tasks, Maui outperforms human indexers. It is more consistent with Computer Science students than they are with each other when assigning topics from Wikipedia to Computer Science technical reports, and it is more consistent with taggers who collect bioinformatics papers in their online

bookmarks repository than these taggers are with each other. It is possible to identify groups of best performing students and taggers whose performance is comparable to that of professionals. Even when this is done, Maui exhibits competitive performance.


Tag (metadata)
Semantics
Index (information technology)
Training set
Knowledge
Algorithm
Vocabulary
Information retrieval
Machine learning
Index (search engine)

Chapter 8
Conclusions

Topic indexing, or identifying the main topics in a document, is traditionally performed by people. Libraries employ specifically trained professional indexers. On the web, where thousands of documents are created daily, topics are provided by volunteer taggers, who organize parts of the web that are important to them. This thesis has investigated how to perform topic indexing automatically.

The main hypothesis is that "with access to domain and general semantic knowledge computers can index as well as humans" (Section 1.2). Section 8.1 summarizes the steps taken to test this hypothesis, leading to its validation through the development and evaluation of the Maui algorithm. Maui identifies topics automatically, and in many cases outperforms human indexers. Section 8.2 revisits several research questions related to the problem of topic indexing, while Section 8.3 gives directions for further research in this field.

8.1 Evidence for the hypothesis

Each chapter of the thesis addresses a different aspect of the hypothesis. Chapter 1 introduced the topic and discussed the motivation behind the research. Being able to identify the main topics of a document is useful in many areas: as subject headings in traditional libraries, as keywords and keyphrases in academic publications, and as tags on socially constructed websites. Manual topic indexing is expensive, so automatic approaches are in great demand.

Chapter 2 compared different types of topic indexing and found that the main distinctions lie in the source of terminology and the expected number of topics that can co-exist in a document. It showed that, in principle, different types of

topic indexing are closely related and can be addressed by a single multi-purpose algorithm.

Chapter 3 surveyed existing methods for automatic topic indexing, dividing them into three major groups: term assignment, keyphrase extraction and tagging. Historically, these have developed somewhat independently of each other. Machine learning techniques have been successfully applied to keyphrase extraction, but have received limited attention in other approaches. The advantages of controlled vocabularies in term assignment have not been explored in either keyphrase extraction or tagging.

The Maui algorithm described in Chapters 5 and 6 addresses the topic indexing task using a simple and generally applicable strategy. Candidate topics are identified in the document and the most significant ones chosen based on their properties using a machine learning approach. This capitalizes on the advantages of previously suggested techniques, while avoiding their limitations. Maui adopts machine learning techniques, but is parsimonious in the use of training data. It also provides ways of controlled indexing of any document collection, without the need for specially crafted vocabularies.

Chapter 5 began by presenting the candidate generation approaches that Maui utilizes. In term assignment, document phrases are mapped to terms in a controlled vocabulary. When Wikipedia is used as the vocabulary, statistics derived from the Wikipedia corpus are applied to identify meaningful phrases and disambiguate them to relevant Wikipedia articles. Tagging follows traditional keyphrase extraction approaches, using document phrases as candidate topics.

The second part of Chapter 5 discussed the properties of candidate topics, or features, that reflect different kinds of knowledge useful for automatic topic indexing. Features like a candidate's frequency and occurrence position in the document describe statistical, language-specific knowledge. The number of times a candidate has been previously assigned as a topic constitutes domain-specific knowledge. Controlled vocabularies and Wikipedia are mined for different kinds of semantic knowledge, such as the relatedness to other candidates and the generality of a topic.

Chapter 6 demonstrated how Maui uses machine learning techniques to derive a good combination of properties from manually indexed documents. A classifier analyzes the typical distribution of feature values for those candidates that human indexers have chosen as topics, and also for all other candidates. Chapter 6 presented two classifiers: Naïve Bayes, which works well when only a few features are employed, and bagged decision trees, which outperform Naïve Bayes when dependencies between many features need to be integrated. Maui ranks all candidates based on their probability of being a topic, as computed by the classifier, and chooses the top scoring ones.

Evaluation on an array of test collections in Chapter 7 demonstrated that Maui can be successfully applied to term assignment in agricultural, physics and medical domains, including assigning agricultural terms to French and Spanish documents. Maui also accurately assigns terms from Wikipedia, which outstrips the terminology of conventional controlled vocabularies by many orders of magnitude. On all tasks and collections, Maui performs better than or as well as competing systems.

Finally, analysis of Maui's indexing consistency with people addresses the central hypothesis of this thesis. While Maui does not achieve the same consistency as professional indexers, its performance is better than that of Computer Science students who assigned topics to documents in their field in a competitive environment. Maui also assigns better tags to documents than voluntary taggers on CiteULike. In order to achieve this performance, Maui draws on semantic and general background knowledge about concepts of human language derived from controlled vocabularies, Wikipedia, and manually indexed training data.

8.2 Answering the research questions

Seven research questions related to topic indexing were introduced in Section 1.4 and answered in this thesis. This section summarizes the answers and shows how they help support the main hypothesis.

1. How can indexing performance be measured?

> *Among many possible measures of indexing performance, inter-indexer consistency analysis is the one that provides a direct comparison between automatic topic indexing and human performance.*

Section 2.2 surveyed techniques commonly used to evaluate the performance of topic indexing. Human indexers are traditionally evaluated using an inter-indexer consistency measure that identifies the degree of their agreement with each other (Rolling, 1981). In contrast, algorithms are usually evaluated either by matching their topics against those assigned by just one person, or by human judges who manually assess the correctness of each topic. Both strategies are limited, partly because the "correctness" of a topic is subjective and partly because they do not give any basis for comparing with human performance on the same task.

This thesis proposes a new strategy for assessing the quality of automatic indexing. Instead of matching against a single set of "correct" topics, the algorithm is required to match the overlap between several sets of topics provided by different human indexers, who in turn are required to match the topics assigned by their colleagues. The human indexers and the algorithm are compared with each other using the same documents and the same measure: inter-indexing consistency. The values are computed per document and indexer, then averaged to return a single number representing the average consistency of each indexer (or the algorithm) with the co-indexers. The average agreement of human indexers with each other serves as the performance level to which the algorithm aspires in order to be considered human-competitive. By evaluating on several data sets representing different types of topic indexing, we demonstrate that our approach is generally applicable.

Admittedly, several problems arise. Multiply-indexed document collections are time-consuming to produce, and therefore rare. This was addressed by creating and making publicly available a new collection with 20 documents, each indexed each by 15 teams of two people. Furthermore, a new approach was developed for extracting high-quality multiply-tagged documents from socially constructed folksonomies, and an example dataset was created containing 180 documents indexed

by 330 users extracted from CiteULike.org. A second problem with consistency-based evaluation is that the results are not comparable to other approaches and systems. A common, but hardly ideal, practice among researchers is to report precision and recall, which were developed for other information retrieval tasks (van Rijsbergen, 1979). In this thesis, precision and recall values are given, as well as the inter-indexer consistency results.

2. What is the performance of human indexers?

> *The average consistency of human indexers ranges from 18.5% to 37.8%, depending on the task, indexer's language proficiency, and expertise in the domain.*

The consistency of professional indexers is reported as ranging from 13% to 70% when a controlled vocabulary is used and from 4% to 67% when topics are freely chosen (Markey, 1984). Chapter 4 presented three new studies of inter-indexer consistency on data sets representing different types of topic indexing. The first used a data set provided by the Food and Agriculture Organization of the UN containing 30 documents with topics independently assigned by six professional indexers from the domain-specific thesaurus Agrovoc. The consistency of these indexers varied from 26% to 44%, with an average of 38.7%.

The second experiment tested 15 teams, each consisting of two Computer Science students, who indexed 20 Computer Science technical reports using titles of Wikipedia articles as topic descriptors. The study was designed as a competition (see Appendix C) in which the team with the highest consistency with the 14 other teams received a prize. The competitive environment and the fact that participants were familiar with the domain ensured high quality assignments despite the lack of indexing training. Consistency ranged from 21% to 37%, with an average of 30.5%. Six teams containing two senior students of which at least one was a native speaker were particularly consistent with each other, achieving an average of 38.3%.

The third experiment analyzed the performance of the CiteULike web bookmarking platform using documents available through HighWire and Nature. The set was restricted to only those taggers who had tagged at least three documents

and agreed on at least three tags with any of the co-taggers of the same documents. These restrictions were necessary to ensure high quality of the CiteULike subgroup, against which Maui was compared. It is not known whether any of these taggers had indexing experience, but they were evidently knowledgeable—or at least interested—in the field of the documents because they chose to record them in their personal bibliography collection. The inter-indexer consistency achieved by 332 taggers in the resulting set of 180 documents varied from 3% to 92%, with an average of 18.5%. Limiting the group to taggers who had indexed at least five documents and exhibited above-average consistency with others yielded a set of best performing taggers whose inter-indexer-consistency is 37.6%—similar to that achieved by professionals and by the best performing Computer Science students.

Comparing the three studies, the professional indexers, as expected, performed best, followed by the students and then the volunteer taggers. Chapter 4 showed that indexing quality depends less on professional training and more on factors such as language skill, familiarity with the domain, existence of indexing guidelines and the availability of a searchable vocabulary.

3. How can a computer understand document concepts?

> *An algorithm can express document concepts in the form of terms from a controlled vocabulary. When the vocabulary is very large, document phrases can be disambiguated to vocabulary terms by computing the likelihood of possible meanings and their semantic relatedness to context.*

A deep understanding of a document's meaning is not required for performing topic indexing. Even professional indexers often just skim the document without actually reading it (Bonura, 1994). However, it is necessary to at least understand the concepts the document discusses. This implies identifying document words and phrases that mean the same thing, and, if a controlled vocabulary is used, linking them to vocabulary terms that correspond to the same concepts. Maui implements this process in its candidate generation stage, because every concept that appears in the document is a potential topic.

Section 5.1.1 explained how documents are mapped to terms in domain-specific thesauri. These thesauri are created for indexing a specific domain, and therefore

exhibit little ambiguity. Here, a simple matching of document words and phrases to vocabulary terms is sufficient, and normalization techniques such as case folding, stemming and word ordering help to identify more relevant concepts.

Identifying candidates with Wikipedia is more challenging, because it is general, covers a large number of topics, and lists millions of terms. Nearly every document phrase matches the title of at least one Wikipedia article; most phrases match several. Section 5.1.2 described how meaningful phrases can be identified using the *keyphraseness* measure. If a phrase matches just one Wikipedia article, this article is used as context for disambiguating other phrases. If a phrase matches several articles, the likelihood of each article's meaning and its semantic relatedness to the context articles are computed using statistics derived from Wikipedia. The best scoring article is chosen as the mapping.

This approach demonstrates how an algorithm can "understand" the meaning of document phrases by a) computing the likelihood of possible mappings and b) relating them to the context in which these phrases appear. This approach intuitively parallels how humans understand language. Everyone knows from experience that certain meanings spring to mind quicker than others, and that context helps to determine a meaning that is less likely *a priori*. The online encyclopedia Wikipedia not only provides the largest available vocabulary of possible topics but can also be mined for statistics that help to identify the correct meanings automatically. Maui is one of the first algorithms developed for this purpose.

4. Is controlled indexing possible in the absence of a vocabulary?

> *Controlled topic indexing can be performed using the online encyclopedia Wikipedia as a generic vocabulary of topics.*

As discussed in Section 4.2.1, Wikipedia was never intended for this use, but it has naturally evolved an organization that resembles the structure of domain-specific thesauri used for indexing. Although it omits many specialist terms that would be encoded in a domain-specific thesaurus, it covers the vast majority of those that are actually used as topics by professional indexers (Milne *et al.*, 2006). Although Wikipedia does not encode semantic relations between terms as accurately as a

thesaurus, its synonymy links (i.e., redirects) are nearly perfect and its link mark-up offers an additional source of synonyms for semantic conflation (Milne *et al.*).

Given Wikipedia's sheer size and wide coverage it can be applied for controlled indexing of most document collections. In this thesis it has successfully indexed Computer Science reports, entertainment and politics news, technical blog posts and other areas. Topic indexing with Wikipedia is a growing field, and new approaches were proposed as this thesis was being prepared for submission (Grineva *et al.*, 2009; Coursey *et al.*, 2009; Fader *et al.*, 2009).

5. How can main concepts be identified automatically?

> *Statistical, semantic, syntactic, lexical, domain-specific and encyclopedic features describe the likelihood of a candidate being a topic. Bagged decision trees efficiently combine these features into a single scheme.*

The second part of Chapter 5 discussed how candidate topics—each corresponding to a concept mentioned in the document—can be analyzed to identify significant ones. Whereas some researchers perform graph-based analysis (Mihalcea and Tarau, 2004; Grineva *et al.*, 2009), Maui uses the filtering method, which has been successfully applied to supervised keyphrase extraction (Section 3.2.1), and adopts it for all types of topic indexing.

For each candidate topic, a set of features is computed that describe statistical, semantic, syntactic, domain-specific and encyclopedic knowledge about it. When manually assigned topics are known for a document, the candidates serve as training instances. If a candidate matches a manually assigned topic it is a positive instance; otherwise it is negative. The distribution of feature values for positive and negative instances is recorded in a model defined by the classifier. Maui achieves the best results with bagged decision trees, which successfully handle dependencies between many features. The learned model is then applied to candidates computed for new documents, for which the main topics are unknown. The impact of the individual features was analyzed using the ROC statistic in Section 5.2.6 and in the evaluation in Chapter 7. The combination of all Maui's features produced the best result in the majority of scenarios; however, in some cases, eliminating some of the features was necessary to achieve best performance.

6. How much training is necessary?

> *Maui performs well with a few dozen training documents. Its performance increases significantly as the amount of the training data grows up to 300 documents, with smaller improvements after that.*

Maui achieves good results after training on very small sets of documents with manually assigned topics. The smallest collection used in the experiments, the Spanish corpus, contained less than 50 manually assigned topic sets, one per document. Section 7.2.4 showed that, even given this small volume of training data, Maui was able to identify over a third of the topics assigned by people. In the French corpus, when fewer than 70 topic sets were available, Maui identified over a quarter of the manually assigned topics. When topics assigned by several humans were available, Maui achieved human-competitive results with as few as 20 documents (Section 7.3.1).

Section 7.2.6 quantified the improvement in Maui's indexing performance as the size of the training set increases. The inter-indexer consistency with professional indexers nearly doubled when all 780 documents were used for training, compared to training on only a few. The largest increases were observed when adding the first 300 documents; from then on the consistency gradually stabilized but still showed potential for further improvement.

7. Can an indexing algorithm be domain and language independent?

> *The algorithm yields an F-measure of 38.5% to 47.6% on a wide range of domains, and topic indexing experiments in other languages are promising.*

Chapter 7 evaluated Maui's performance on different topic indexing tasks. Each one corresponds to a different domain. Term assignment was tested on agricultural, medical and physics documents; indexing using Wikipedia was tested on Computer Science technical reports, news, blog posts, forums and product reviews; automatic tagging was tested on science papers, including areas such as bioinformatics and economics.

Maui proved to be general enough to return accurate topic sets for documents in every domain. It achieved the greatest F-measure (47.6%) when assigning Medical

Subject Headings to 500 medical documents (provided by Gay *et al.*, 2005), and the smallest (38.5%) when assigning terms from the Agrovoc thesaurus to 780 documents from the FAO repository. It performed better than humans on Computer Science technical reports, at a level comparable to some of the native speaking graduate students. It performed nearly as well as the *best* taggers of scientific papers on CiteULike. No domain-specific modifications were required for any of these experiments.

A small-scale experiment demonstrated Maui's language independence. The algorithm was tested on separate collections of 67 French and 47 Spanish documents indexed with terms from the version of the Agrovoc thesaurus in the corresponding language. The stemmer, the list of stopwords and the encoding of the documents needed to be changed, but this did not affect the core of the algorithm. Maui's results were worse than those achieved on the English agricultural collection, but reasonable considering the very small training sets. The algorithm still matched approximately one-third of correct French topics and one-quarter of Spanish ones, and the non-matching topics were related to the document's content.

8.3 Future work

This thesis answered many research questions and supported the hypothesis stated in Chapter 1. However, many topic indexing areas were left unexplored. This section suggests future experiments with Maui and discusses its potential applications.

One of the oldest and largest manually constructed controlled vocabularies is the Library of Congress Subject Headings (LCSH) thesaurus. It has been actively maintained by the Library of Congress (2001) since 1898 and is used as the main indexing vocabulary by many libraries in the United States and other countries. The SKOS formatted version of LCSH has only just been released,[1] so Maui has not yet been tested on this vocabulary. Such an experiment would be extremely

[1] *http://id.loc.gov/authorities/search/*

valuable given the wide use of LCSH and the relative paucity of research published so far that uses it (e.g. Larson, 1992; Godby and Stuler, 2001; Paynter, 2005).

The investigation of topic indexing in other languages has been limited to small collections with French and Spanish documents in just one domain. This is another major area for future work. In particular, experiments with Wikipedia versions in languages other than English would be of interest. Wikipedia represents the largest available multi-lingual vocabulary, covering over 250 languages—including indigenous and constructed ones. Every Wikipedia version is formatted and structured in a similar way and is freely available for download. Given that multi-lingual access in Wikipedia Miner (Milne, 2009) is underway, it is likely that Maui can soon be applied directly to documents in other languages.

Given the impressive quality of the topics that Maui chooses, it is natural to speculate about possible applications and use cases. Where professional indexers are employed for topic indexing, Maui can provide suggestions and help librarians choose topics more efficiently. An interesting experiment would be to extend Maui to constantly update its indexing model using feedback provided by indexers. If an indexer did not find an automatically provided topic useful, Maui would record this as a negative example. If a topic that an indexer selected was not identified automatically, Maui would add it as an additional positive example. After each document the algorithm would re-train a model to capture the preferences of that particular indexer. Evaluating indexing quality before and after feedback-based training sessions would determine the efficiency and utility of this approach.

A similar experiment using collaborative tagging platforms would provide volunteer taggers with automatically generated tag suggestions. Currently, most bookmarks on these sites remain untagged, or are limited to just a few tags. It would be interesting to see whether Maui can improve the situation by making tagging less cumbersome and more accessible, and also whether tag suggestions improve the consistency of assigned tags and thus the quality and utility of the resulting folksonomy.

Finally, there are many use cases for automatically generated topics. They can be used for browsing the topics discussed in a personal blog, a website, or a digital li-

brary of any kind. Topics can be used to augment snippets in search results, or to organize search results by grouping them by most frequently discussed topics. A document's main topics form a succinct semantic representation of its content, which can be useful for implementing other applications: automatic clustering of large collections of documents, text summarization and information retrieval.

"Can't a computer do the indexing?—The short answer is no," states the web page of the American Society for Indexing,[2] explaining that computers cannot understand and organize ideas and information in text. This thesis investigated the performance of human indexers and compared it directly to that of an algorithm. Automatic topic indexing does not fall far behind the performance of trained professionals. In fact, it produces topics that in many cases are better than those assigned by humans. Given the vast ocean of information out there and the high demand for topic indexing, professional indexers would do well to embrace the lifeline that computers can offer, instead of disparaging their potential contribution.

[2] *http://www.asindexing.org/site/indfaq.shtml*

References

Agrovoc. (1992). Multilingual agricultural thesaurus. Food and Agriculture Organization of the United Nations, Rome: Apimondia.

Arampatzis, A., T. P. van der Weide, C. H. A. Koster, and P. van Bommel. (1998). Phrase-based information retrieval. *Information Processing and Management*, 34(6): 693–707.

Aronson, A. R., O. Bodenreider, H. F. Chang, *et al.* (2000). The NLM indexing initiative. In *Proc. Ann. Fall Symp. of the American Medical Informatics Assoc.*, *AMIA'00,* Los Angeles, CA, US. Philadelphia: Hanley and Belfus, pp. 17–21.

van Assem, M., V. Malaise, A. J. Miles and G. Schreiber. (2006). A method to convert thesauri to SKOS. In Y. Sure and J. Domingue (edt.), *Proc. 3rd Annual European Semantic Web Conf., Budva, Montenegro; Lecture Notes in Computer Science*, 4011. Springer, pp. 95–109.

Barker, K., and N. Cornacchia. (2000). Using noun phrase heads to extract document keyphrases. In Hamilton, H. J. (edt.) *Advances in Artificial Intelligence. Proc. 13th Canadian Conf. on AI*, Montréal, Québec, Canada, pp. 40–52.

Barrière, C., and M. Jarmasz. (2004). Keyphrase extraction: enhancing lists, In *Proc. Computational Linguistics in the North-East*, *CLINE'04*. Montréal, Québec, Canada. NRC 48079.

Bates, M. J. (1998). Indexing and access for digital libraries and the Internet: Human, database, and domain factors. *Journal of the American Society for Information Science*, 49(13): 1185–1205.

Bonura, L. S. (1994). *The Art of Indexing*. New York: John Wiley and Sons, Inc.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2): 123–140.

Brin S., and L. Page. (1998). The anatomy of a large-scale hypertextual web search engine. In *Proc. 7th Int. Conf. on World Wide Web*, *WWW'98*, Brisbane, Australia. *Computer Networks and ISDN Systems*, 30(1–7): 107-117.

Brooks, C. H., and N. Montanez. (2006). Improved annotation of the blogosphere via autotagging and hierarchical clustering. In *Proc. Int. Conf. on World Wide Web, WWW'06,* Edinburgh, UK. New York, NY: ACM Press, pp. 625–632.

Budura, A., S. Michel, P. Cudre-Mauroux, and K. Aberer. (2008). To tag or not to tag - harvesting adjacent metadata in large-scale tagging systems. In *Proc. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, Singapore. New York, NY: ACM Press, pp. 733–734.

Chirita, P. A., S. Costache, W. Nejdl, and S. Handschuh. (2007). P-tag: large scale automatic generation of personalized annotation tags for the web. In *Proc. Int. Conf. on World Wide Web, WWW'07,* Banff, Canada. New York, NY: ACM Press, pp. 845–854.

Church, K. W., and P. Hanks. (1989). Word association norms, mutual information and lexicography. In *Proc. 27th Ann. Conf. of the Association for Computational Linguistics, ACL'89*. New Brunswick, NJ: ACL, pp. 76–83.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1): 37–46.

Coursey, K., R. Mihalcea, and W. Moen. (2009). Using encyclopedic knowledge for automatic topic identification. In *Proc. Conf. on Natural Language Learning, CoNLL'09*, Boulder, CO. New Brunswick, NJ: ACL, pp. 210–218.

Csomai, A., and R. Mihalcea. (2007). Linking educational materials to encyclopedic knowledge. In *Proc. 13th Int. Conf. on Artificial Intelligence in Education, AIED'07*, Los Angeles, CA, USA. Frontiers in Artificial Intelligence and Applications 158. IOS Press, pp. 557–559.

Csomai, A., and R. Mihalcea. (2008). Linguistically motivated features for enhanced back-of-the-book indexing. In *Proc. Ann. Conf. of the Association for Computational Linguistics, ACL/HLT'08, Columbus, Ohio*. New Brunswick, NJ: ACL, pp. 932–940.

D'Avanzo, E., and B. Magnini. (2005). A keyphrase-based approach to summarization: the LAKE System at DUC-2005. In *Proc. DUC Workshop at the Human Language Technology Conf. HLT/EMNLP'05*, Vancouver, B.C., Canada.

David, C., L. Giroux, S. Bertrand-Gastaldy, and D. Lanteigne. (1995). Indexing as problem solving: A cognitive approach to consistency. In *Forging New Partnerships in Information, Medford, NJ, Information Today*, pp. 49–45.

Domingos, P., and M. Pazzani (1997). On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning 29*(2/3): 103–130.

Dumais, S. T., Platt, J., Heckerman, D., and M. Sahami. (1998). Inductive learning algorithms and representations for text categorization. In *Proc. Int. Conf. on Information and Knowledge Management, CIKM'98*, New York, NY: ACM Press, pp. 148–155.

Fader, A., S. Soderland, and O. Etzioni. (2009). Scaling Wikipedia-based named entity disambiguation to arbitrary web text. In *Proc. Wiki-AI Workshop at IJCAI'09 Conf.*, Pasadena, CA, US.

Fayyad, U. M., and K. B. Irani. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. In *Proc. Int. Joint Conf. on Artificial Intelligence, IJCAI'93*, San Mateo, CA. San Francisco, CA: Morgan Kaufmann, pp. 1022–1027.

Fellbaum, C. (edt.) (1998). *WordNet An Electronic Lexical Database*. Cambridge, MA: MIT Press.

Frank, E., G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning. (1999). Domain-specific keyphrase extraction. In *Proc. 16th Int. Joint Conf. on Artificial Intelligence*, *IJCAI'99*, Stockholm, Sweden. San Francisco, CA: Morgan Kaufmann, pp. 668–673.

Fuhr, N., and G. Knorz. (1984). Retrieval test evaluation of a rule based automatic index (AIR/PHYS). In *Proc. ACM SIGIR Conf. on Research and Development in Information Retrieval*, Cambridge, UK: Cambridge Univ. Press, pp. 391–408.

Fuller, M., and J. Zobel (1998). Conflation-based comparison of stemming algorithms. In *Proc. 3rd Australian Document Computing Symposium*, Basser Dept. of Computer Science, University of Sydney, pp. 8–13.

Funk, M., and C. Reid. (1983). Indexing consistency in MEDLINE. *Bulletin of the Medical Library Association*, 71: 176–183.

Gabrilovich, G., and S. Markovitch. (2007). Computing semantic relatedness using Wikipedia-based Explicit Semantic Analysis. In *Proc. 20th Int. Joint Conf. on Artificial Intelligence, IJCAI'07*, Hyderabad, India, p.1606–1611.

Gay, C. W., M. Kayaalp, and A. R. Aronson. (2005). Semi-automatic indexing of full text biomedical articles. In *Proc. Annual Fall Symp. of the American Medical Informatics Association*, *AMIA'05*, Washington DC, US. Philadelphia: Hanley and Belfus, pp. 271–275.

Godby, C. J., and J. Stuler. (2001). The library of congress classification as a knowledge base for automatic subject categorization. In *Subject Retrieval in a Network Environment: Proc. IFLA Satellite Meeting*, Dublin, OH. OCLC, pp. 14–16.

Golder, S. A., and B. A. Huberman. (2006). Usage patterns of collaborative tagging systems. *Journal of Information Science*, 32(2): 198–208.

Golub, K. (2006). Automated subject classification of textual web pages, based on a controlled vocabulary. *New Review of Hypermedia and Multimedia*, 12(1): 11–27.

Grineva, M., M. Grinev, and D. Lizorkin. (2009). Extracting key terms from noisy and multi-theme documents. In *Proc. 18th Int. Conf. on World Wide Web, WWW'09, Madrid, Spain*. New York, NY: ACM Press, 2009, pp. 661–670

Gutwin, C., G. Paynter, I. H. Witten, C. G. Nevill-Manning, and E. Frank (1998). *Improving browsing in digital libraries with keyphrase indexes*. Technical report, Department of Computer Science, University of Saskatchewan, Canada.

HaCohen-Kerner, Y., Z. Gross, and A. Masa. (2005). Automatic extraction and learning of keyphrases from scientific articles. In *Proc. Int. Conf. on Computational Linguistics and Intelligent Text Processing, CICLing'05*, Mexico City, Mexico. Springer, pp. 657–669.

Heymann, P., D. Ramage, and H. Garcia-Molina. (2008). Social tag prediction. In *Proc. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, Singapore. New York, NY: ACM Press, pp. 531–538.

Hilberer, T. (2003). Aufwand vs. Nutzen: Wie sollen deutsche wissenschaftliche Bibliotheken künftig katalogisieren? *Bibliotheksdienst*, 37: 754–758.

Hooper, R. S. (1965). *Indexer consistency tests—Origin, measurements, results and utilization*. IBM Washington Systems Center, Bethesda, Maryland. Presented at the 1965 Congress Int. Federation for Documentation.

Hripcsak, G., and A. S. Rothschild. (2005). Agreement, the F-Measure, and reliability in information retrieval. *Journal of American Medical Information Association*, 12(3): 296–298.

Hulth, A. (2003). Improved automatic keyword extraction given more linguistic knowledge. In *Proc. Conf. on Empirical Methods in Natural Language Processing, EMNLP'03*, Japan. New Brunswick, NJ: ACL, pp. 216–223.

Hulth, A. (2004). *Combining machine learning and natural language processing for automatic keyword extraction*. Ph. D. thesis, Stockholm University.

Iivonen, M. (1995). Consistency in the selection of search concepts and search terms. *Information Processing and Management 31*(2): 173–190.

Jones, S., and M. Mahoui. (2000). Hierarchical document clustering using automatically extracted keyphrases. In *Proc. 3rd Int. Asian Conf. on Digital Libraries*, New York, NY: ACM Press, pp. 113–120.

LCSH (2001). Library of Congress Subject Cataloging Division. *Library of Congress Subject Headings* (24 Ed.). Library of Congress.

Landauer, T. K., P. Foltz, and D. Laham. (1998). Introduction to latent semantic analysis. *Discourse Processes*, 25.

Larson, R. R. (1992). Experiments in automatic library of congress classification. *Journal of the American Society for Information Science,* 43(2): 130–148.

Leininger, K. (2000). Interindexer consistency in PsycINFO. *Journal of Librarianship and Information Science, 32*(1): 4–8.

Leonard, L. E. (1975). *Inter-indexer consistency and retrieval effectiveness: measurement of relationships*. Ph. D. thesis, Graduate School of Library Science, University of Illinois, Urbana-Champaign, IL.

Lovins, J. B. (1968). Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 11(1-2): 11–31.

Manning, C. D., and H. Schütze. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Massachusetts.

Markey, K. (1984). Inter-indexer consistency test: A literature review and report of a test of consistency in indexing visual materials. *Library and Information Science Research*, 6: 157–177.

Markó, K., U. Hahn, S. Schulz, P. Daumke, and P. Nohama. (2004). Interlingual indexing across different languages. In *Proc. Int. Conf. Recherche d'Information Assistée par Ordinateur*, *RIAO'04*, Avignon, France, pp. 100–115.

McBride, B. (2001). Jena: Implementing the RDF Model and Syntax Specification, In S. Staab et al. (eds.): *Proc. 2nd Int. Workshop on the Semantic Web, SemWeb'01*, Hong Kong, China.

Medelyan, O. (2005). *Automatic keyphrase indexing with a domain-specific thesaurus.* Master Thesis. Albert-Ludwig University Freiburg, Germany.

Medelyan, O., and I. H. Witten. (2006). Measuring inter-indexer consistency using a thesaurus. In *Proc. 6th ACM/IEEE-CS Joint Conf. on Digital Libraries, JCDL'06*, Chapel Hill, NC, USA. New York: ACM Press, pp. 274–275.

Medelyan, O., D. Milne, C. Legg, and I. H. Witten. (2009). Mining meaning from Wikipedia. *Int. Journal of Human-Computer Studies* (in press).

Medical Subject Headings. (2005). U. S. Department of Health and Human Services, National Library of Medicine (NLM). Bernan Assoc.

Mihalcea, R., and A. Csomai. (2007). Wikify!: Linking documents to encyclopedic knowledge. In *Proc. ACM Conf. on Information and Knowledge Management, CIKM'07,* Lisbon, Portugal. New York, NY: ACM Press, pp. 233–242.

Mihalcea, R., and P. Tarau. (2004). TextRank: Bringing orderinto texts. In *Proc. Int. Conf. on Empirical Methods in Natural Language Processing, EMNLP'04*, Barcelona, Spain. New Brunswick, NJ: ACL, pp. 404–411.

Milne, D., O. Medelyan, and I. H. Witten. (2006). Mining domain-specific thesauri from Wikipedia: A case study. In *Proc. IEEE/WIC/ACM Int. Conf. on Web Intelligence, WI'06,* Hong Kong. IEEE Computer Society Press, pp. 442–448.

Milne, D., I. H. Witten, and D. M. Nichols. (2007). A knowledge-based search engine powered by Wikipedia. In *Proc. Conf. on Information and Knowledge Management, CIKM'07*, Lisbon, Portugal, pp. 445–454.

Milne, D., and I. H. Witten. (2008). Learning to link with Wikipedia. In *Proc. ACM Conf. on Information and Knowledge Management, CIKM'08*, Napa Valley, CA, US. New York, NY: ACM Press, pp. 509–518.

Milne, D. (2009). An open-source toolkit for mining Wikipedia. In *Proc. New Zealand Computer Science Research Student Conf., NZCSRSC'09,* Auckland, New Zealand.

Mishne, G. (2006). Autotag: a collaborative approach to automated tag assignment for weblog posts. In *Proc. Int. Conf. on World Wide Web, WWW'06,* Edinburgh, UK. New York, NY: ACM Press, pp. 953–954.

Nguyen, T. D., and M.-Y. Kan. (2007). Keyphrase extraction in scientific publications. In *Proc. Int. Conf. on Asian Digital Libraries, ICADL'07,* Hanoi, Vietnam. New York, NY: ACM Press, pp. 317–326.

Paice, C., and W. Black. (2003). A three-pronged approach to the extraction of key terms and semantic roles. In *Proc. Int. Conf. on Recent Advances in Natural Language Processing, RANLP'03*, Borovets, Bulgaria, pp. 357–363.

Paukkeri, M.-S., I. T. Nieminen, M. Polla, and T. Honkela. (2008). A language-independent approach to keyphrase extraction and evaluation. In *Proc. 22nd Int. Conf. on Computational Linguistics, COLING'08*, Manchester, UK, pp. 83–86.

Paynter, G. W., S. J. Cunningham, and I. H. Witten. (2000). Evaluating extracted phrases and extending thesauri. In *Proc. 3rd Int. Asian Conf. on Digital Libraries*, *ICADL'00*, Seoul, Korea. New York, NY: ACM Press, pp. 131–138.

Paynter, G. W. (2005). Developing practical automatic metadata assignment and evaluation tools for internet resources. In *Proc. ACM/IEEE-CS Joint Conf. on Digital Libraries, JCDL'05*. New York, NY: ACM Press, pp. 291–300.

Pepe, A., and J. Yeomans. (2008). Protocols for scholarly communication. *Astronomical Society of Pacific Conf. Series,* 377. San Francisco: ASP.

Plaunt, C. and B. A. Norgard. (1998). An association based method for automatic indexing with a controlled vocabulary. *Journal of the American Society for Information Science*, 49 (10): 888–902.

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 14(3): 130–137.

Pouliquen, B., R. Steinberger, and I. R. Camelia. (2003). Automatic annotation of multilingual text collections with a conceptual thesaurus. In *Proc. Workshop on Ontologies and Information Extraction at the EUROLAN Conf.*, Cluj-Napoca, Romania, pp. 19–28.

Quinlan J. R. (1993). *C4.5: Programs for Machine Learning*. San Mateo: Morgan Kaufmann.

van Rijsbergen, C. J. (1979). *Information Retrieval*. Butterworths, London.

Rolling, L. (1981). Indexing consistency, quality and efficiency. *Information Processing and Management*, 17: 69–76.

Saarti, J. (2002). Consistency of subject indexing of novels by public library professionals and patrons. *Journal of Documentation*, 58: 49–65.

Salton, G., and C. Buckley. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24 (5): 513–523.

Saracevic, T., and P. Kantor. (1988). A study of information seeking and retrieving. III. Searchers, searches, and overlap. *Journal of the American Society for Information Science*, 39: 197–216.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1): 1–47.

Sigurbjörnsson, B., and R. van Zwol. (2008). Flickr tag recommendation based on collective knowledge. In *Proc. Int. Conf. on World Wide Web, WWW'08,* Beijing, China. New York, NY: ACM Press, pp. 327–336.

Soergel, D. (1994). Indexing and retrieval performance: The logical evidence. *Journal of the American Society for Information Science*, 45: 589–599.

Sood, S., K. Hammond, S. Owsley, and L. Birnbaum. (2007). TagAssist: Automatic tag suggestion for blog posts. In *Proc. Int. Conf. on Weblogs and Social Media, ICWSM'07*, Boulder, Colorado. Omnipress.

Strube, M., and S. P. Ponzetto. (2006). WikiRelate! Computing semantic relatedness using Wikipedia. In *Proc. Int. Conf. on Artificial Intelligence, AAAI'06*, pp.1419–1424.

Tiun, S., R. Abdullah, and T. E. Kong. (2001). Automatic topic identification using ontology hierarchy. In *Proc. 2nd Int. Conf. on Computational Linguistics and Intelligent Text Processing, CICLing'01*, Mexico City, Mexico. Springer, pp. 444–453.

Turney, P. D. (1999). *Learning to extract keyphrases from text.* Technical report ERB-1057, National Research Council Canada, Institute for Information technology.

Turney, P. D. (2000). Learning algorithms for keyphrase extraction. *Information Retrieval*, 2(4): 303–336.

Turney, P. D. (2003). Coherent keyphrase extraction via web mining. In *Proc. 18th Int. Joint Conf. on Artificial Intelligence,* Acapulco, Mexico. San Francisco, CA: Morgan Kaufmann, pp. 434–439.

Witten, I. H., and E. Frank. (2005). *Data mining: Practical machine learning tools and techniques with Java implementations.* 2nd edition. San Francisco, CA: Morgan Kaufmann.

Witten, I. H., G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning. (1999). Kea: Practical automatic keyphrase extraction. In *Proc. ACM Conf. on Digital Libraries,* Berkeley, CA, US. New York, NY: ACM Press, pp. 254–255.

Wright, S. E., and G. Budin. (2001). *Handbook of terminology management. Volume 2: Application-oriented terminology management.* Amsterdam: John Benjamins.

Xu, Z., Y. Fu, J. Mao, and D. Su. (2006). Towards the Semantic Web: Collaborative tag suggestions. In *Proc. Collaborative Web Tagging Workshop at the Int. Joint Conf. on Artificial Intelligence,* Stockholm, Sweden.

Zunde, P., and M. E. Dexter. (1969). Indexing consistency and quality. *American Documentation*, 20: 259–267.

Appendix A. Glossary

Categories are general terms that group documents based on their broad area, e.g. *Politics*, *Entertainment*, *Sport*. See also **Text Categorization**.

Clustering is related to **topic indexing** in that it identifies groups of documents on the same topic; however, these groups are unlabeled.

Collaborative tagging is the process of assigning **Tags** to documents by a group of users. A website that supports this activity allows its users to annotate the documents based on their personal preferences. See also **Folksonomy**.

Concept is a mental representation of an object. This representation reflects the unambiguous meaning of this object. A **Topic** usually refers to a concept.

Controlled vocabulary is a flat or hierarchically organized list of terms used for topic indexing. The terms can be of two kinds: **Descriptors** and **Non-Descriptors**.

Controlled indexing is topic indexing with terms from a **Controlled vocabulary**. See also **Free indexing**.

Corpus is a collection of documents.

Descriptor is the preferred phrase for specifying a topic in indexing, as opposed to a **Non-Descriptor**, which is an alternative phrase with the same meaning.

Disambiguation is the process of identifying the intended meaning of a word or a phrase in a given context.

Domain is an area or a field of human activity. It can be general, e.g. *Computer Science*, or specific, e.g. *Information Retrieval*.

Domain knowledge is the knowledge of language use in a particular domain, e.g. what terms are most frequent or what topics are often discussed in a domain.

Domain-specific thesaurus is a manually created hierarchy of terms used in a given domain. It can be used as a **Controlled vocabulary** for topic indexing.

Features are properties of terms that characterize their statistical and linguistic behavior. Based on their features, terms that are identified as candidate topics in a given document can be classified into topics and non-topics according to a **Model** derived from a **Training set**.

Folksonomy is a vocabulary of **Tags** assigned by all users participating in **Collaborative tagging** on a given website.

Free indexing is unrestricted topic indexing. Any word or phrase can be chosen to describe a topic. Examples of free indexing are **Tagging** and **Keyphrase extraction**. See also **Controlled indexing.**

Full text indexing is the opposite of topic indexing. All words and phrases that appear in a document, and not just the main topics, are included in the **Index**.

Full text index is the result of **Full text indexing**. It lists all words and phrases and their occurrences in documents of a given **Corpus**.

Full text search is the process of matching a user's **Query** to all words and phrases that appear in documents of a given **Corpus** using the **Full text index**.

Indexing is a process of collecting and organizing **Metadata** required for efficient search in a given document collection.

Index is the result of **Indexing**, which can take many forms depending on what information about documents is collected and how it is stored.

Indexing consistency measures how similar are individually created **Indexes**. See for example, **Inter-indexer consistency**.

Inter-indexer consistency (also called **inter-indexer agreement**) is a measure of agreement between indexers on **Index terms** describing a document. It is usually computed over a sample of the entire collection and used as an indicator of indexing quality.

Index term is a word or a phrase used in the **Index** to describe a given document. Traditionally an index term is a **Descriptor** from a **Controlled vocabulary**. **Subject headings**, **Keyphrases** and **Tags** are examples of index terms.

Keyphrase is a word or a phrase (usually a **Noun phrase**) that describes a topic covered by a document. Synonyms: **Key phrase, Keyword, Key word, Key term**.

Keyphrase set is a set of **Keyphrases** representing all main topics in a document.

Keyword, see **Keyphrase**.

Keyword search is a process of retrieving documents based on a **Query** that expresses the information need of the user. Depending on the implementation, keyword search can imply matching the query against **Keyphrase sets** assigned to the documents, or all **Metadata** elements, or it can be equivalent to **Full text search**.

Machine learning is a technique that allows computers to learn the execution of particular tasks based on example data, the **Training set**. The learning implies automatic recognition of patterns in this data.

Metadata (or **meta data**) is information about data that facilitates quick access to it. Common forms of metadata about documents are title, author and publisher. **Keyphrases**, **tags**, and **index terms** are examples of *content-based* metadata.

Model is a combination of rules for execution of a particular task derived using **machine-learning** techniques. In topic indexing, the model is derived from analyzing **Features** of candidate topics in the **Training set**.

Multiply indexed documents are documents indexed by several people individually, which results in several topic sets for the same document. Such documents are used to measure indexing quality in a collection by computing the **Inter-indexer consistency** of people who indexed them.

Named entities are **noun phrases** that refer to objects like people, companies, geographical locations, and dates.

Non-descriptor is an alternative way of referring to a **Concept** represented by a **Descriptor** in a **Controlled vocabulary**.

Noun phrase is a phrase consisting of a head noun (e.g. *biology*) and, optionally, its modifiers like other nouns (e.g. *cell biology*), adjectives (e.g. *molecular biology*) or prepositional phrases (e.g. *biology of marine mammals*). Most topics are expressed as noun phrases.

Query, or **Search query**, is a word or phrase expressing a user's information need.

Semantic knowledge is knowledge about the meaning of words and phrases, e.g. about their relatedness to other words and phrases.

Stopword (or **stop word**) can be any word that the algorithm ignores in the processing. Stopwords are usually all words from closed word classes such as prepositions (e.g. *in, for*) or pronouns (e.g. *me, someone*), but sometimes other words as well that are very frequent and bear little meaning (e.g. *usually*, *different*)

Subject heading is the same as **Descriptor**—an **Index term** used to refer to a topic in indexing.

Subject indexing is the same as **Topic indexing**, commonly used to describe the process of assigning **Subject headings**.

Supervised learning is a form of **Machine learning** where the algorithm is supplied with labeled examples for deriving a model. In **Topic indexing** the labeled examples are documents with manually assigned topics. An example of *unsupervised* learning is **Clustering**.

Tag is a word or a phrase describing a topic of a document. It is essentially the same as a **Keyphrase**. See also **Collaborative tagging**.

Tagger is a person who assigns tags, usually on websites that supports **Collaborative tagging**, e.g. *flickr.com*, *del.icio.us*, *CiteULike.com*.

Tagging is the process of assigning **Tags**. See also **Collaborative tagging**.

Thesaurus is a hierarchy of terms and their semantic relations. See also **Domain-specific thesaurus.**

Text categorization is the process of assigning **Categories** to documents.

Topic is a subject discussed in a document. It can be expressed as a **Subject Heading**, a **Keyphrase**, or a **Tag**.

Topic indexing is the process of identifying the main topics in a document.

Topic set is a set of the main **Topics** in a document, see also **Keyphrase set**.

Training is the process of deducing a **Model** from a **Training set**.

Training set is a collection of labeled examples, i.e. documents with manually assigned terms, used in **Supervised learning** to deduce a **Model** of how a task is performed.

Appendix B. Publications

List of publications that have arisen out of this PhD research (2005 – 2009).

1. **Journal articles**

   - **O. Medelyan**, D. Milne, C. Legg and I. H. Witten. 2009. Mining meaning from Wikipedia. *International Journal of Human-Computer Studies* (in press)

     An extensive survey of research involving mining meaningful information from Wikipedia for use in natural language processing, information extraction and retrieval tasks.

   - **O. Medelyan**, I. H. Witten, 2008. Domain independent automatic keyphrase indexing with small training sets. *Journal of the American Society for Information Science and Technology*. Vol. 59 (7), pp. 1026-1040

     An approach for automatic topic indexing using manually created thesauri, tested on three domains and two additional languages.

2. **Peer-reviewed conference and workshop articles**

   - S. Sarjant, C. Legg, M. Robinson, **O. Medelyan**. 2009. "All You Can Eat" Ontology-Building: Feeding Wikipedia to Cyc. In Proc. *of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence, WI'09*, Milan, Italy (Sept.).

     Adding new children to Cyc concepts using pattern matching and link analysis.

   - **O. Medelyan**, E. Frank, I. H. Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Proc. of the Int. Conf. on Empirical Methods in Natural Language Processing, EMNLP'09*, Singapore (Aug.).

     Analysis of tagging quality in the collaboratively-created tagging taxonomy CiteULike. A new technique to automatic tagging, based on an existing keyphrase extraction algorithm, which was extended and improved.

   - **O. Medelyan**, I. H. Witten, D. Milne, 2008. Topic indexing with Wikipedia. In *Proc. of Wiki-AI Workshop at the AAAI'08 Conference*. Chicago, US (June).

     An approach for automatic indexing using Wikipedia as a vocabulary. An evaluation on 30 computer science reports indexed by 15 teams of students shows that the algorithm performs as well as the humans.

   - **O. Medelyan**, C. Legg, 2008. Integrating Cyc and Wikipedia: Folksonomy meets rigorously defined common-sense. In *Proc. of Wiki-AI Workshop at the AAAI'08 Conference*. Chicago, US (July).

     An new technique for mapping concepts from the Cyc ontology onto Wikipedia.

- **O. Medelyan**, D. Milne, 2008. Augmenting domain-specific thesauri with knowledge from Wikipedia. In *Proc. of NZ CSRSC*, Christchurch, NZ.

  Given a domain-specific thesaurus, its concepts are mapped onto Wikipedia articles. Next, we analyze what new information can be added from Wikipedia.

- **O. Medelyan**, 2007. Computing lexical chains with graph clustering. In *Proc. of Student Research Workshop at the ACL'07 Conference*. Prague, Czech Republic.

  Document concepts build meaningful groups that can be automatically detected using graph clustering. The resulting lexical chains improve text summarization and topic indexing.

- **O. Medelyan**, 2007. Topical indexing with WordNet. In *Proc. of NZ CSRSC*, Hamilton, NZ (April).

  We investigate WordNet as a general vocabulary for topic indexing. It turns out to be problematic because its concepts are too general and the senses are too specific.

- D. Milne, **O. Medelyan** and I. H. Witten, 2006. Mining domain-specific thesauri from Wikipedia: A case study. In *Proc. of the IEEE/WIC/ACM International Conference on Web Intelligence, WI'06*, Hong Kong (Dec.).

  We compare an agricultural thesaurus Agrovoc with Wikipedia in order to determine parts of Wikipedia that are equivalent to manually defined semantic relations.

- I. H. Witten, **O. Medelyan**, D. Milne, 2006. Finding documents and reading them: Semantic metadata extraction, topic browsing and realistic books. In *Proc. of Russian Conference on Digital Libraries, RCDL'06*. Suzdal, Russia (Oct.).

  An overview of the state-of-the art techniques used in digital libraries.

- **O. Medelyan**, 2006. Semantically enhanced automatic keyphrase indexing. *In Proc. of Women in Machine Learning Workshop, WiML, Grace Hopper Conf.* San Diego, US, (Oct.). (Poster)

  A poster presenting application of lexical chains to automatic keyphrase indexing.

Appendix C.
Indexing competition

The first part of this appendix is a slightly rewritten description of the user study conducted during the research for this thesis, described in Section 4.2.2.[1] The second part discusses the outcomes of the study and presents diagrams reflecting the performance of the participants.

C.1 Design of the user study

Background

Keywords and keyphrases describe the main topics in a document and are useful metadata in traditional and digital libraries. When human indexers select such phrases, they often disagree with each other. An objective decision on what phrases are correct can be made when several keyphrase sets defined for the same document are available. The more people select a keyphrase, the higher its relevance to the particular document. The goal of this study is to collect such multiply indexed data for the task of keyphrase indexing with Wikipedia.

Instructions

What will be indexed? The document collection consists of 20 articles covering different Computer Science topics, such as programming theory, algorithms, graphs, image compression, software visualization, usability studies, and AI. The documents will be shown in randomized order to help eliminate fatigue effects.

How to index? Skim—or read if necessary—the shown article and select single words or phrases describing its content. The number of phrases should ideally correspond to the number of main topics described in the document. Depending on the document's length there should be between 5 and no more than 15 topics. Try to assign a *perfect*

[1] The original is at: *http://www.cs.waikato.ac.nz/~olena/indexing_competition.html*

keyphrase set by guessing what phrases all the other participants would choose. Use Wikipedia as a controlled vocabulary.

What is a controlled vocabulary? Wikipedia is an online encyclopedia and is used as a controlled vocabulary in this experiment. It contains over 1 million articles describing concepts that can serve as keyphrases. The article titles are all listed in a large index, along with redirects—phrases that refer to the same concepts but have different spelling, grammatical form or wording. Each redirect links to the equivalent concept, e.g. *visualisation* → *visualization* or *distance learning* → *distance education*.

The best way to search Wikipedia is to look through its index. On the webpage *http://en.wikipedia.org/wiki/Special:Allpages/* you can search for phrases starting with a prefix of your choice, which has the advantage of approximate searching. Redirects are shown in italic.

Valid keyphrases are terms that have a corresponding article in Wikipedia. Read the article about the concept in Wikipedia to ensure that it refers to the same topic as in the document. If the document contains a redirect, choose the article to which it links.

Documents

Twenty most interesting and easily readable documents were manually chosen from the collection of Computer Science technical reports. The students were presented with an example, an article about distance learning using the web published in 1995:

http://vlib.org/cuisung/papers/OpenDistLearning/paper

and example keyphrases:

Distance education
World Wide Web
Educational software
Computer program
Hypermedia

Each keyphrase was linked to the Wikipedia article describing its meaning.

Competition

The class was divided into fifteen teams of two students who collaborate in solving the indexing task. The students read the instructions, including the example document and its Wikipedia keyphrases.

Suggestion on the collaboration:

a) First, the students sit together on the same computer and read the text. Then, one of them suggests phrases and the other one checks them on Wikipedia (e.g. by using two computers placed next to each other).

b) Students may read and index the document independently, then compare their lists and discuss the differences. However, this would probably take more time.

Together the students decide on what keyphrases are appropriate and submit their decision via an online interface.

Each team is in competition with all the others. After all teams have solved the task, the scores are computed by using the collected data and the winner is announced.

Design of the experiment

The data was collected over a web interface created specifically for this study. The task starts with the registration of the team; after that all documents need to be indexed in one go. Once keyphrases are entered, they cannot be changed. The competition should be completed within the arranged time in the computer labs at the University of Waikato. No interaction between the teams is allowed.

Positive aspects

The collaboration within the team plus each team's desire to win should influence the quality of results in a positive way. In teams of two, the task can potentially be solved twice as fast. Wikipedia provides a detailed description of each index term, so there is no confusion on their meaning. The students will be reading only documents related to their study area. While reading and selecting the index term they learn more about subjects related to their study.

C.2 Examples and results

In the beginning, it was not clear how long it would take the students to assign topics to documents. During the first test, in 1 hour and 30 minutes—the standard length of a lesson—the teams could only index 10 documents. Therefore, a second sesson was used to collect a data set of a sufficient size: 20 documents. The students enjoyed the exercise a lot and were enthusiastic about the task.

After the study the following statistics were computed:

1. What is the average consistency among the teams?
2. What is the perfect keyphrase set for each article?

23267. P++: A language for software system generators			
Team 1	*Team 2*	*Team 3*	*Team 4*
Programming language	Programming language	High-level progr. lang.	Programming language
Software engineering	Software architecture	Software engineering	Object-oriented prog.
Software architecture	Software development	Object-oriented prog.	Inheritance (Comp. Sc.)
Software devel. process	Inheritance (Comp. Sc.)	Inheritance (Comp. Sc.)	Software componentry
Object-oriented prog.	Systems design	Information hiding	C++
Comp. programming	C++	Abstraction (Comp. Sc.)	
Software componentry			

13259. Cone trees in the UGA graphics system: Suggestions of a more robust visualization tool			
Team 1	*Team 2*	*Team 3*	*Team 4*
Scientific visualization	Visualization (Graphic)	Visualization (Graphic)	Widget (Computing)
3D Computer Graphics	Hierarchical model	Computer Graphics	Hierarchical model
Widget (Computing)	Three-dimens. space	3D	Model
UXGA	Rapid prototyping	Constr. solid geometry	Shapes
HCI	Tree (Data structure)	Visual complexity	Animation
		Visual communication	

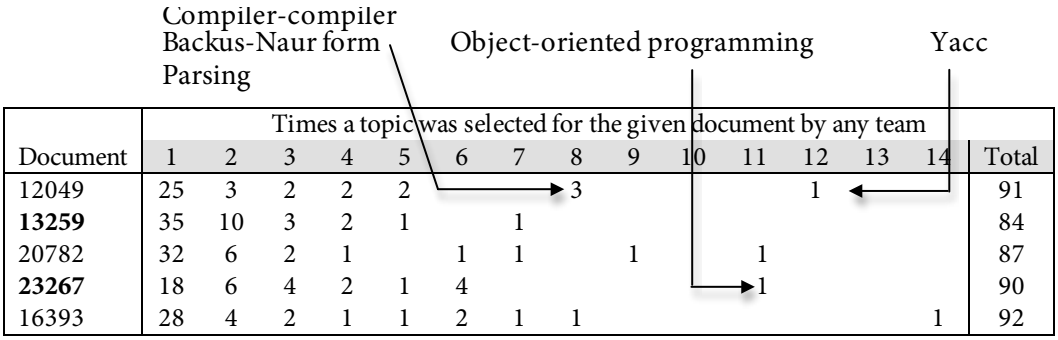
Figure C.1 Examples of topics assigned by four teams to two documents

3. Which of the teams performed the best? (i.e. has greatest average consistency with all topic sets).

The data set was used to develop an approach to automatic topic indexing with Wikipedia described in this thesis (see Sections 4.2.1 and 7.3).

Before the second exercise, several diagrams were presented to the users describing their performance on the first 10 documents. This was useful to demonstrate the nature of topic indexing: agreement and disagreement between the indexers on what topics describe the document the best.

Figure C.1 shows two example documents and topics assigned by four of the teams. The documents were chosen based on the team's consistency: the first document, 13259, had the highest consistency, whereas the second, 23267, the lowest. Very little overlap can be observed, particularly on the second document.



	Times a topic was selected for the given document by any team														
Document	1	2	3	4	5	6	7	8	9	10	11	12	13	14	Total
12049	25	3	2	2	2			3				1			91
13259	35	10	3	2	1		1								84
20782	32	6	2	1		1	1		1		1				87
23267	18	6	4	2	1	4					1				90
16393	28	4	2	1	1	2	1	1						1	92

Figure C.2 Distribution of agreement on topics across the documents

Figure C.2 lists the documents and the number of topics assigned to five sample documents. The last column is the sum of the numbers, whereas the intermediate columns describe how many teams agreed on a particular topic. For example, 8 teams agreed on 3 topics for document 12049: *Compiler-compiler*, *Backus-Naur form* and *Parsing*. The topic *Yacc* was chosen by 12 out of 15 teams. In document 13259, on which the teams performed the worst, 35 out of 84 topics were idiosyncratic, i.e. chosen by just one team. The document 23267, on which the teams achieved the highest consistency, contains the least number of idiosyncratic terms: only 18 out of 90. Section 4.2.2 contains a detailed analysis of the indexing performance of the teams, whereas Section 7.3.1 compares their inter-indexer consistency with that of Maui. Appendix D.5 lists topics on which most of the teams have agreed and compares them to those assigned automatically by Maui.

Appendix D. Additional results

This appendix expands on the evaluation results reported in Chapter 7.

D.1 Domain keyphraseness baseline

Evaluation of document topics automatically generated from candidates with the highest domain keyphraseness values, computed over all manually indexed documents in the same collection. The results can be used as one of the baselines (Section 7.1.2).

	P	R	F
Term assignment			
FAO-780: Agricultural	8.6	8.8	8.7
67 French FAO documents	9.6	7.2	8.2
47 Spanish FAO documents	8.5	10.9	9.5
CERN-290: Physics	9.0	12.8	10.5
NLM-500: Medicine	23.9	17.5	20.2
	Inter-indexer consistency		
	min	avg	max
Term assignment			
FAO-30: Agricultural	7.8	12.4	17.9
Indexing with Wikipedia			
WIKI-20: Computer Science	6.8	12.1	17.9
Tagging			
CiteULike-180: Science	25.2	26.3	25.5

D.2 Medicine and physics examples

Examples of Maui's topics assigned to medical and physics documents evaluated in Section 7.2.3.

Medicine NLM-500 corpus and Medical Subject Headings thesaurus (MeSH terms).
Determining lifestyle correlates of body mass index using multilevel analyses.

assigned by professionals	assigned by Maui
Adult	**Aged**
Aged	**Body Mass Index**
Body Mass Index	Cardiovascular Diseases
Female	**Humans**
Humans	**Longitudinal Studies**
Life Style	**Middle Aged**
Linear Models	Motor Activity
Longitudinal Studies	**Norway**
Male	**Risk Factors**
Middle Aged	
Norway	
Obesity	
Questionnaires	
Risk Factors	
Urban Population	

Physics CERN-290 corpus & High Energy Physics thesaurus.
Two-loop electroweak corrections to Higgs production at hadron colliders.

assigned by professionals	assigned by Maui
quark: top	**quark: top**
Higgs mass	**Higgs particle**
standard model	**standard model**
LHC-B	**LHC-B**
quantum chromodynamics	**quantum chromodynamics**
spontaneous symmetry breaking	cross section
Higgs boson	Higgs particle: mass
symmetry breaking	hadron: production
intermediate boson: mass	gluon: fusion
	boson

D.3 French and Spanish examples

Examples of Maui's topics assigned to French and Spanish documents evaluated in Section 7.2.4.

French FAO-67. Relations terres-eau dans les bassins versant ruraux

assigned by professionals	assigned by Maui
Impact sur l'environnement	**Impact sur l'environnement**
Ressource en eau	**Ressource en eau**
Utilisation des terres	**Utilisation des terres**
Aménagement de bassin versant	Bassin versant
Conservation de l'eau	Pollution de l'eau
Qualité de l'eau	Changement climatique
Utilisation de l'eau	Pollution atmosphérique
Réglementation	Pollution par l'agriculture
Eau de ruissellement	Étude de cas
Développement rural	Métal lourd
Organisation socioéconomique	

Spanish FAO-47. Evaluación de los recursos forestales mundiales 2000

assigned by professionals	assigned by Maui
Ordenación forestal	**Ordenación forestal**
Plantación forestal	**Plantación forestal**
Productos forestales	**Productos forestales**
Recursos forestales	**Recursos forestales**
Ciencias forestales	Inventarios forestales
Reforestación	Tierras forestales
Sostenibilidad	Bosques
Transferencia de tecnología	Biodiversidad
	Utilización de la tierra
	Muestra

D.4 FAO-30 results

Examples of Maui's topics assigned to 30 agricultural documents indexed by 6 professional indexers. The topics were evaluated in Section 7.2.5 and 7.2.7.

Most frequent topics by 6 professionals	Topics assigned by Maui
The dynamics of sanitary and technical requirement assisting the poor to cope	
Food safety (5)	Food safety (5)
Livestock (5)	Livestock (5)
Standards (5)	Standards (5)
Poverty (5)	Developing countries (4)
Developing countries (4)	Food chains (4)
Food chains (4)	Animal health (2)
Phytosanitary measures (4)	FAO (2)
Animal production (2)	Risk management (2)
National plan of action to prevent, deter and eliminate illegal fishing	
Fishery policies (5)	Fisheries (4)
Pacific islands (4)	Pacific islands (4)
Legislation (4)	Fishing operations (3)
Fisheries (4)	Fishing vessels (2)
Fishing operations (3)	FAO (2)
Fishing vessels (2)	Pacific islands (trust territory) (0)
Monitoring (2)	International law (0)
Manual for the monitoring and management of queen conch	
Strombus gigas (6)	FAO (2)
Monitoring (6)	Fisheries (2)
Fishery management (6)	Fishery data (1)
Caribbean (4)	Stock assessment (1)
Fishery policies (3)	Management (0)
Data analysis (3)	Overfishing (0)
Data collection (3)	Fishing operations (0)
The legal framework for the management of animal genetic resources	
Animal genetic resources (6)	Food safety (3)
Legislation (4)	Animal health (2)
Biodiversity (4)	Animal products (2)
Wto (3)	Animal breeding (2)
Food safety (3)	Management (1)
Animal products (2)	Genetic resources (1)
Animal health (2)	Agriculture (1)

Most frequent topics by 15 teams	Topics assigned by Maui
Guidelines for soil description	
Soil genesis (5)	Soil genesis (5)
Soil classification (5)	Soil classification (5)
Land use (4)	Carbonates (2)
Soil chemicophysical properties (3)	Soil structure (2)
Topography (3)	FAO (2)
Vegetation (3)	Soil density (2)
Soil morphological features (3)	Gypsum (2)
Carbonates (2)	Soil (1)
Changing the gender situation in forestry	
Gender (6)	Gender (6)
Forestry (5)	Forestry (5)
Europe (4)	Role of women (4)
Role of women (4)	FAO (3)
Employment (3)	Women (1)
North America (3)	Forest management (1)
Armenia (3)	Self management (0)
FAO (3)	Continuing education (0)
Climate change and the forest sector	
Climatic change (6)	Climatic change (6)
International agreements (5)	Forests (4)
Forests (4)	Greenhouse gases (3)
Greenhouse effect (4)	Forest management (3)
Legislation (4)	Property (0)
Forestry policies (4)	Climate (0)
Pollution control (4)	Land use (0)
Greenhouse gases (3)	Forest products (0)
Feeding Asian cities: food production and processing issues	
Food production (5)	Food production (5)
Food supply (5)	Food supply (5)
Food policies (5)	Urban agriculture (3)
Asia (5)	Rural urban relations (3)
Urbanization (4)	Agricultural sector (2)
Towns (4)	Supply balance (1)
Rural urban relations (3)	Urban population (0)
Urban agriculture (3)	Agriculture (0)
Population improvement: A way of exploiting the genetic resources of Latin America	
Rice (6)	Latin America (6)
Latin America (6)	Plant genetic resources (4)
Breeding methods (5)	Recurrent selection (3)
Plant genetic resources (4)	Genetic resources (2)
Plant population (3)	FAO (1)
Oryza (3)	Brazil (1)
Genetic markers (3)	Brazil (federal district) (0)

Most frequent topics by 15 teams	Topics assigned by Maui
A practical manual for producers and exporters from West Africa	
Certification (6)	Certification (6)
Regulations (6)	Regulations (6)
Standards (6)	West Africa (5)
Exports (5)	European union (3)
West Africa (5)	FAO (1)
Agricultural products (5)	Africa (0)
USA (3)	UN (0)
European union (3)	Maximum residue limits (0)
Role of local institutions in reducing vulnerability to natural disasters: Viet Nam	
Viet Nam (6)	Viet Nam (6)
Flooding (5)	Flooding (5)
Sustainable development (5)	Natural disasters (5)
Risk management (5)	Rural development (2)
Natural disasters (5)	Disasters (1)
Early warning systems (4)	Disaster preparedness (0)
Emergency relief (4)	Risk (0)
Case studies (4)	High water (0)
The growing global obesity problem: Some policy options to address it	
Overweight (6)	Overweight (6)
Nutrition policies (4)	Food consumption (3)
Price policies (4)	Diet (2)
Food consumption (3)	Developing countries (2)
Prices (2)	Developed countries (1)
Taxes (2)	Consumption (0)
Fiscal policies (2)	Agriculture (0)
Feeding habits (2)	Food supply (0)
Estimating poverty over time and space: Costa Rica	
Costa Rica (6)	Costa Rica (6)
Statistical methods (5)	Statistical methods (5)
Poverty (5)	Poverty (5)
Measurement (4)	Land use (2)
Data analysis (3)	Methods (1)
Land use (2)	Buenos Aires (0)
Economic indicators (2)	FAO (0)
Cartography (2)	World bank (0)
Selected indicators of food and agriculture development in Asia-Pacific region	
Asia and the Pacific (6)	Asia and the Pacific (6)
Trade (5)	Trade (5)
Development indicators (5)	Fisheries (3)
Agricultural development (5)	Farmland (2)
Livestock (4)	Agriculture (2)
Forestry (4)	FAO (1)
Foods (3)	Animal products (0)
Fisheries (3)	Asia (0)

Most frequent topics by 15 teams	Topics assigned by Maui
The rehabilitation of fisheries and aquaculture in tsunami affected countries in Asia	
Aquaculture (6)	Aquaculture (6)
Asia (6)	India (4)
Natural disasters (6)	Capacity building (3)
Thailand (4)	Fisheries (2)
Indonesia (4)	Fisheries development (2)
Myanmar (4)	FAO (1)
Coastal fisheries (4)	Asia and the Pacific (0)
Special R&D report on the FAO Viet Nam coffee project	
Viet Nam (6)	Coffee (5)
Coffee (5)	FAO (4)
FAO (4)	Coffea (2)
Ochratoxins (4)	Coffee beans (2)
Quality (4)	Drying (1)
Storage (4)	Relative humidity (0)
Home gardens key to improved nutritional well-being	
Domestic gardens (6)	Domestic gardens (6)
Food security (5)	Food security (5)
Households (4)	Lao people's democratic republic (4)
Lao people's democratic republic (4)	FAO (3)
Vegetables (3)	Nutrition education (2)
Nutritional status (3)	Leaf vegetables (0)
FAO (3)	Gardens (0)
Workshop on quality and safety in the horticultural marketing chains of Asia	
Asia (6)	Asia (6)
Food safety (6)	Food safety (6)
Vegetables (5)	Marketing (3)
Horticulture (4)	Fruits (3)
Quality (4)	FAO (2)
Fruits (3)	Thailand (2)
Marketing (3)	Safety (1)
Introducing the international bioenergy platform	
Bioenergy (6)	Bioenergy (6)
Knowledge management (4)	Knowledge management (4)
International cooperation (3)	FAO (3)
FAO (3)	Capacity building (2)
Capacity building (2)	Wood energy (2)
Partnerships (2)	Information needs (0)
Evaluation (2)	Fuel crops (0)
Community diversity seed fairs in tanzania. Guidelines for seed fairs	
Seed (6)	Seed (6)
Biodiversity (5)	Biodiversity (5)
United republic of Tanzania (3)	FAO (2)
Villages (3)	Crops (2)
Tanzania (3)	Food security (1)
Exhibitions (3)	Maize (0)

Most frequent topics by 15 teams	Topics assigned by Maui
Tackling the bushmeat crisis in Africa	
Africa (6)	Africa (6)
Wild animals (6)	Wild animals (6)
Food security (5)	Food security (5)
Nature conservation (5)	West Africa (2)
Meat (4)	Wildlife (1)
Trade (4)	FAO (1)
Food production (3)	Ghana (1)
Game meat (3)	Animal protein (1)
Seaweed farming: An alternative livelihood for small-scale fishers?	
Seaweed culture (6)	Seaweed culture (6)
Indonesia (5)	Indonesia (5)
Sulawesi (4)	Income (1)
Socioeconomic development (4)	Philippines (0)
Artisanal fisheries (4)	Fishing effort (0)
Fishery management (3)	Fisheries (0)
Seaweed products (2)	Farming systems (0)
Fishermen (2)	Developing countries (0)
Prospects for food, nutrition, agriculture and major commodity groups	
Agriculture (5)	Agriculture (5)
World (4)	World (4)
Food consumption (4)	Food consumption (4)
Food production (3)	FAO (1)
Nutritional status (3)	Developing countries (1)
Forecasting (3)	East Asia (0)
Agricultural products (3)	China (0)
Sugar (2)	South Asia (0)
Overview of techniques for reducing bird predation at aquaculture facilities	
Aquaculture (5)	Aquaculture (5)
Bird control (5)	Fencing (3)
Predatory birds (4)	North America (1)
Fencing (3)	Predation (1)
Damage (3)	Walls (0)
Noxious birds (3)	Fish larvae (0)
Fishery production (2)	Water (0)
Fish culture (2)	Utah (0)
Phosphorus limitation of microbial processes in moist tropical forests	
Tropical rain forests (6)	Carbon (2)
Phosphorus (6)	Costa rica (2)
Soil chemicophysical properties (3)	Tropical forests (0)
Soil fertility (3)	Respiration (0)
Soil microorganisms (3)	Rain forests (0)
Soil biology (2)	Primary productivity (0)
Microorganisms (2)	Forests (0)
Biodegradation (2)	Soil (0)

Most frequent topics by 15 teams	Topics assigned by Maui
Conserving plant genetic diversity for dependent animal communities	
Arthropoda (6)	Species (2)
Plant genetic resources (6)	Genetic variation (2)
Biodiversity (6)	Populus (1)
Resource conservation (4)	Salicaceae (1)
Plant animal relations (3)	Dominant species (1)
Population genetics (2)	Host plants (0)
Species (2)	Genetic distance (0)
Genetic variation (2)	Natural hybridization (0)
The optimal allocation of ocean space: Aquaculture and wild-harvest fisheries	
Marine areas (6)	Marine areas (6)
Marine fisheries (5)	Aquaculture (5)
Aquaculture (5)	Carrying capacity (1)
Fishery production (3)	Fishing operations (1)
Fishery resources (3)	Fishing effort (0)
Economic competition (2)	Fisheries (0)
Fishery management (2)	Growth rate (0)
Fishing rights (2)	Fish (0)
Cassava for livestock feed in sub-Saharan Africa	
Africa south of Sahara (6)	Cassava (5)
Cassava (5)	Livestock (4)
Livestock (4)	Feeds (4)
Feeds (4)	Africa (1)
Animal feeding (4)	Maize (0)
Feed production (2)	Poultry (0)
Food security (2)	Production (0)
Marketing (2)	Nigeria (0)
Case study of the Papua New Guinea vanilla industry	
Papua new guinea (6)	Papua new guinea (6)
Prices (5)	Prices (5)
Vanilla (spice) (5)	Vanilla (spice) (5)
Diversification (5)	Processing (0)
Exports (4)	Exchange rate (0)
Vanilla planifolia (3)	Agriculture (0)
Products (2)	Pacific islands (0)
Industry (2)	Vanilla (genus) (0)
Codex Alimentarius. Food import & export	
Exports (6)	Codex alimentarius (4)
Imports (6)	Food inspection (4)
Certification (6)	Food safety (3)
Codex alimentarius (4)	FAO (1)
Food inspection (4)	Risk assessment (1)
Foods (4)	Public health (0)
Food safety (3)	International trade (0)
Standards (3)	Monitoring (0)

D.5 WIKI-20 results

Examples of Maui's topics assigned to 20 Computer Science technical reports indexed by 15 teams of Computer Science students. The topics were evaluated in Section 7.3.1 and 7.3.2.

Most frequent topics by 15 teams	Topics assigned by Maui
10894. A safe, efficient regression test selection technique	
Regression testing (15)	Algorithm (7)
Software maintenance (13)	Control flow (0)
Control flow graph (10)	Software maintenance (13)
Software testing (9)	Computer software (1)
Algorithm (7)	Test suite (2)
12049. Occam's razor: the cutting edge for parser technology	
Yacc (13)	Compiler-compiler (9)
Parsing (12)	Yacc (13)
Compiler-compiler (9)	Programming language (4)
Backus Naur form (9)	Parsing (12)
Compiler (6)	Compiler (6)
13259. Cone trees in the UGA graphics system	
Hierarchical model (7)	Visual display unit (0)
3D computer graphics (7)	Graphics (0)
Visualization (graphic) (6)	Computer graphics (3)
Tree (data structure) (5)	Visualization (2)
Computer graphics (3)	PARC (company) (2)
16393. Cache coherence for shared memory multiprocessors	
Virtual memory (15)	Microprocessor (0)
Multiprocessing (9)	Cache coherence (9)
Cache coherency (9)	CPU cache (0)
Consistency model (7)	Parallel computing (2)
Sequential consistency (6)	Sequential consistency (6)
18209. Mutable object state for object-oriented logic programming	
Object-oriented programming (15)	Object-oriented programming (15)
Logic programming (14)	Logic (2)
Linear logic (6)	Logic programming (14)
Immutable object (5)	Prolog (1)
Deductive database (4)	Programming language (1)
19970. A new deterministic parallel sorting algorithm	
Sorting algorithm (13)	Sorting algorithm (13)
Parallel computing (10)	Algorithm (2)
Deterministic algorithm (6)	Sampling (statistics) (1)
Computational complexity theory (5)	Sampling (music) (1)
Load balancing (computing) (4)	Parallel computing (10)

Most frequent topics by 15 teams	Topics assigned by Maui
23267. P++: A language for software system generators?	
Programming language (12)	Programming language (12)
C++ (6)	Abstraction (2)
Component-based software engineering (6)	Abstraction (computer science) (5)
Encapsulation (6)	Software system (0)
Abstraction (computer science) (5)	Software engineering (4)
23507. Models for computer generated parody	
Parody (11)	Artificial intelligence (9)
Language model (9)	Ernest Hemingway (5)
Artificial intelligence (9)	Computer science (0)
Vocabulary (7)	Grammar (4)
Natural language processing (6)	Statistics (3)
23596. The effect of group size and communication modes in CSCW environments	
Communication (8)	Problem solving (2)
Computer supported cooperative work (8)	Human communication (1)
Collaborative software (7)	Computer supported cooperative work (8)
Collaborative workspace (7)	Computer (0)
Collaboration (5)	Communication (8)
25473. Extracting multi-dimensional signal features	
Content-based image retrieval (7)	Tree (data structure) (1)
Image processing (5)	Wavelet (1)
Computer vision (5)	Discrete cosine transform (2)
Image compression (5)	Database (2)
Feature extraction (5)	Color histogram (2)
287. Clustering full text documents	
Machine learning (13)	Machine learning (13)
Cluster analysis (10)	Natural language processing (4)
Information retrieval (8)	Natural language (5)
Index (search engine) (6)	Algorithm (2)
Natural language (5)	Information retrieval (8)
37632. The internet software visualization laboratory	
Software visualization (12)	Software visualization (12)
Electronic learning (8)	Computer programming (3)
Distance education (6)	Visualization (2)
Education (4)	Computer (0)
Internet (4)	Computer science (0)
39172. Block edit models for approximate string matching	
NP-complete (12)	Approximate string matching (3)
String searching algorithm (10)	Edit distance (1)
Computational complexity theory (5)	Algorithm (2)
Levenshtein distance (5)	NP-complete (12)
String (computer science) (5)	String searching algorithm (10)

Most frequent topics by 15 teams	Topics assigned by Maui
39955. Structured interviews on the object-oriented paradigm	
Object-oriented programming (15)	Software engineering (6)
Software engineering (6)	Software maintenance (5)
Structured interview (6)	Object-oriented programming (15)
Interview (6)	Structured interview (6)
Software maintenance (5)	Programming language (0)
40879. Instance pruning techniques	
Machine learning (14)	Algorithm (5)
Nearest neighbour algorithm (9)	Metric (mathematics) (0)
Algorithm (5)	Machine learning (14)
Training set (5)	Learning (0)
Artificial intelligence (4)	Training set (5)
43032. Observations and recommendations on software internationalization	
Internationalization and localization (15)	Character encoding (4)
User interface (6)	User interface (6)
Software engineering (6)	Internationalization and localization (15)
Translation (5)	Interface (computer science) (1)
Linguistics (5)	Programming language (0)
7183. The challenge of deep models, inference structures, and abstract tasks	
Expert system (17)	Abstraction (computer science) (3)
Artificial intelligence (12)	Expert system (17)
Model (abstract) (6)	Artificial intelligence (12)
Abstraction (5)	Scientific modeling (0)
Knowledge base (5)	Medical diagnosis (0)
7502. Using introspective reasoning to select learning strategies	
Machine learning (10)	Introspection (7)
Reasoning (7)	Reasoning (7)
Introspection (7)	Machine learning (10)
Artificial intelligence (6)	Case-based reasoning (2)
Learning (3)	Artificial intelligence (6)
9307. Specifying and adapting object behavior during system evolution	
Object-oriented programming (13)	Graph (mathematics) (0)
Software development process (7)	Object-oriented programming (13)
Software engineering (7)	Graph theory (2)
Class (computer science) (5)	Software engineering (7)
Computer-aided software engineering (4)	Graph (0)
20782. High performance geographic information systems	
Geographic information system (15)	Parallel computing (9)
Parallel computing (9)	Load balancing (0)
Distributed Interactive Simulation (9)	Geographic information system (15)
Load balancing (computing) (8)	Algorithm (0)
Parallel programming model (4)	Message passing (1)

Appendix E. Installation

This appendix expands on the installation and usage instructions discussed in Section 6.6.

E.1 Download

Go to *http://www.maui-indexer.googlecode.com*. Select the "*Downloads*" tab. When installing Maui for the first time, choose the archive file *maui_1.1_with_libs.tar.gz* for download. Move the archive file to the preferred installation directory and extract its contents. In a shell (e.g. *Terminal* on Mac) this can be done with the following command:

gzip –dc maui_1.1_with_libs.tar.gz | tar xf –

The extracted directory *Maui1.0* contains the following subdirectories:

- *data* – stopwords, vocabularies and test documents for different tasks
- *lib* – required jar libraries
- *src* – source code
- *doc* – Javadoc documentation

E.2 Set up

Maui is written in Java 5.0. Before running the main classes, they need to be compiled. There are two choices for doing this:

1. Use an IDE (integrated development environment) like *Eclipse* (*http://www.eclipse.org*).
2. Use command line scripts in a shell, e.g. *Terminal*, *Xterm*, *C shell*.

Set up Maui in Eclipse

Import the project into the workspace: *File --> Import*, then follow the instructions. Make sure that all jar files in the *lib* directory are added to the build path: *Project --> Properties --> Java Build Path --> "Libraries" tab --> "Add jars…"*

Make sure that the correct *JRE System Library* is used. If *JVM Java 1.5.0.* is **not** the last item in the list of libraries, click *"Edit…"* and choose it in *Alternate JRE*.

If everything is done correctly, Eclipse will automatically compile Maui without error messages. Now the main class files can be run by clicking on them in the *maui.main* package and opening *Run --> "Open Run Dialogue…"*. In the *Arguments* tab, *Program arguments* window, set the options as described in Section E.5. In the *VM arguments* window, if necessary, increase the size of memory: *–Xmx1200m*.

Set up Maui in a shell

Set MAUIHOME to the directory *Maui1.0*, using the full path address, e.g.:

- for *bash*, *bourne* and related shells:
  export MAUIHOME = /Users/Shared/Olena/Maui1.0
- for *csh* and related shells:
  setenv MAUIHOME /Users/Shared/Olena/Maui1.0
- for DOS and Windows shells
  set MAUIHOME = C:\Maui1.0

Set $MAUIHOME to the CLASSPATH environment variable. Example:

export CLASSPATH=$CLASSPATH:$MAUIHOME

Add *src* directory and all jar files in the *lib* directory to CLASSPATH. Example:

export CLASSPATH=$CLASSPATH:$MAUIHOME/src
export CLASSPATH=$CLASSPATH:$MAUIHOME/lib/jena.jar

If everything is set up correctly, compiling the following scripts should not return any error messages:

javac src/maui/main/Examples.java
javac src/maui/main/MauiModelBuilder.java
javac src/maui/main/MauiTopicExtractor.java

E.3 Wikipedia Miner installation (optional)

Wikipedia Miner (*wikipedia-miner.sourceforge.net*) is required for topic indexing with Wikipedia. It is also used for computing encyclopedic features in other tasks, although this is optional. Download the package and the data from:

https://sourceforge.net/projects/wikipedia-miner/files/

Then follow the installation guide in the readme file distributed with the package.

Once installed, Maui will require the location of the server (e.g. *localhost*), the name of the database containing Wikipedia data (e.g. *en_20090306*), and, optionally, the name of the directory with cvs files containing Wikipedia data, which Wikipedia Miner loads into memory for quick access. The latter is advisable if many documents are processed at a time. It requires approximately 3MB RAM.

E.4 Data preparation

Section 6.6 explained how the input data for Maui needs to be prepared. The directory *data* contains example documents for training and testing Maui for different tasks. Training means creating a topic indexing model; testing means generating topics for new documents. For example, *data/automatic_tagging* contains two directories: *train* and *test*. The first contains three documents and their manually assigned topics from which Maui creates the model, while the second contains a document for which Maui will compute topics. The topics provided for this document are used for evaluation.

Each document is stored in a separate file with extension *.txt*, in plain text form. The topics are saved one per line in corresponding *.key* files; each line may have an optional number that indicates how many people agreed on this topic. Note that the supplied *train* directories contain very little training data, and are only intended for demonstration and testing purposes. When using Maui, either provide your own training data, or download it from:

http://code.google.com/p/maui-indexer/wiki/MultiplyIndexedData

Choose a data set that is similar to the one for which topics are to be extracted.

E.5 Running Maui

The fastest way to understand how Maui works is to look at the *Examples* class. Maui can be also applied to new data using *MauiModelBuilder* and *MauiTopicExtractor*.

Examples

The example script *maui.main.Examples* demonstrates how to use Maui for three kinds of topic indexing: tagging, term assignment and indexing with Wikipedia.

It can be used as a source of code snippets for a direct access from other programs. It can also be used as a test script to check whether Maui is installed correctly.

The simplest kind of topic indexing is tagging:

java maui.main.Examples tagging

Alternatively, choose *term_assignment* or *indexing_with_wikipedia* as the argument, instead of *tagging*. The debugging output will show the created model and the generated topics, which will be evaluated against the manually assigned topics stored in the *test* directories.

In case of *term_assignment*, *Examples* will use the controlled vocabulary *data/vocabularies/agrovoc_sample.rdf*. This is a subset of the original Agrovoc thesaurus, used for demonstration. Appendix G shows where to download the complete Agrovoc vocabulary and where to find other vocabularies in SKOS format.

In case of *indexing_with_wikipedia*, *Examples* will require access to the Wikipedia database, which should be installed as a part of Wikipedia Miner (see Section E.3). Change the parameters accordingly in the main method of *Examples* and recompile it.

ModelBuilder and TopicExtractor

Maui can be applied directly to the document collections in the *data* directory or to the new collections supplied by the user. In either case, a model needs to be created first using MauiModelBuilder. Then MauiTopicExtractor can be applied to generate topics for new documents.

For automatic tagging, the directory name and model name are the only required arguments, e.g.:

java maui.main.MauiModelBuilder
–l data/automatic_tagging/train/ –m test –d
java maui.main.MauiTopicExtractor
–l data/automatic_tagging/test/ –m test –d

For term assignment, the vocabulary name and format need to be supplied, e.g.:

java maui.main.MauiModelBuilder
–l data/term_assignment/train/ –m test
–v agrovoc_sample –f skos –d

java maui.main.MauiTopicExtractor

–l data/term_assignment/test/ –m test
–v agrovoc_sample –f skos –d

For topic indexing with Wikipedia, the vocabulary should be set to *wikipedia* and database access should be supplied, e.g.:

java maui.main.MauiModelBuilder
–l data/wikipedia_indexing/train/
–m indexing_model –v wikipedia –w enwiki@localhost

java maui.main.MauiTopicExtractor
–l data/wikipedia_indexing/test/
–m indexing_model –v wikipedia –w enwiki@localhost

Installation problems and usage questios can be added to the Issue Tracker:

http://code.google.com/p/maui-indexer/issues/list,

or discussed on the SourceForge forum:

https://sourceforge.net/forum/forum.php?forum_id=950425.

Appendix F. Command line arguments and settings

This appendix lists command line arguments for setting up Maui for building topic indexing models and computing topics for new documents.

Command line argument	Description	Default value
For both MauiModelBuilder and MauiTopicExtractor		
–l *directory_name*	Specifies name of directory*	
–m *model_name*	Specifies name of model*	
–e *encoding*	Specifies encoding	"default"
–v *vocab_name*	Specifies vocabulary name, e.g. *agrovoc*	
–f *vocab_format*	Specifies vocabulary format, e.g. *skos* or *txt*	
–i *doc_language*	Specifies document language, e.g. *en*, *es*, *fr*	en
–d	Turns debugging mode on	off
–s *stopwords_class*	Sets the name of the class implementing the stopwords	*StopwordsEnglish*
–t *stemmer_class*	Sets the name of the class implementing the stemmer	*PorterStemmer*
–w *wikip_db@server*	Specifies the name of the MySQL Wikipedia database and the server where it is stored, e.g. *enwiki20090106@localhost*	
For MauiModelBuilder only		
–x *length*	Sets maximum phrase length	3
–y *length*	Sets minimum phrase length	1
–o *number*	Sets minimum number of times a phrase needs to occur	1
For MauiTopicIndexer only		
–n	Specifies the number of topics per document	10
–a	Outputs additional information about the topics	off
–p	Prints a graph vizualising the main topics	off
–g	Builds dictionary with global frequencies from the test set	off

* required arguments

These and several other settings can be also specified programmatically by modifying private class variables:

Private class variable	Setting	Default
For both MauiModelBuilder and MauiTopicExtractor		
String inputDirectoryName	setDirName(String d)	
String modelName	setModelName(String m)	
String documentEncoding	setEncoding(String e)	"default"
String vocabularyName	setVocabularyName(String v)	"none"
String vocabularyFormat	setVocabularyFormat(String f)	null
String documentLanguage	setDocumentLanguage(String l)	"en"
String debugMode	setDebugMode(boolean d)	false
Stopwords stopwords	setStopwords(Stopwords s)	StopwordsEnglish
Stemmer stemmer	setStemmer(Stemmer s)	PorterStemmer
Wikipedia wikipedia	setWikipedia(Wikipedia w)	null
String wikipediaDatabase, String wikipediaServer	setWikipediaConnection(String c) e.g. "database@localhost"	"database" and "localhost"
boolean cacheWikipediaData	setCacheWikipediaData	false
String wikipediaDataDirectory	setWikipediaDataDirectory	null
For MauiModelBuilder only		
int maxPhraseLength	setMaxPhraseLength(int x)	3
int minPhraseLength	setMinPhraseLength(int y)	1
int minNumOccur	setMinNumOccur(int o)	1
boolean basicFeatures	setBasicFeatures(boolean b)	true
boolean keyphrasenesFeature	setKeyphrasenesFeature(boolean k)	true
boolean freqencyFeatures	setFreqencyFeatures(boolean f)	true
boolean positionFeatures	setPositionFeatures(boolean p)	true
boolean lengthFeature	setLengthFeature(boolean l)	true
boolean nodeDegreeFeature	setNodeDegreeFeature(boolean n)	true
boolean basicWikipediaFeatures	setBasicWikipediaFeatures(boolean w)	false
boolean allWikipediaFeatures	setAllWikipediaFeatures(boolean f)	false
Classifier classifier	setClassifier(Classifier c)	null
double minKeyphraseness	setMinKeyphraseness(double k)	0.01
double minSenseProbability	setMinSenseProbability(double s)	0.005
int contextSize	setContextSize(int c)	5
For MauiTopicIndexer only		
int topicsPerDocument	setNumTopics(int n)	10
boolean additionalInfo	setAdditionalInfo(boolean a)	false
boolean printGraph	setPrintGraph(boolean p)	false
boolean buildGlobalDictionary	setBuildGlobal(boolean g)	false

Appendix G. Web resources

Here are some useful links related to automatic topic indexing.

G.1 Software, tools, demos

- **Bibclassify – *http://invenio-demo.cern.ch/help/admin/bibclassify-admin-guide***
  A module in CDS Invenio (CERN's document server software) for automatic assignment of terms from SKOS vocabularies, developed on the High Energy Physics vocabulary. Developed in a collaboration between CERN and DESY.
  Guide – *http://cdsware.cern.ch/tmp/bibclassify/hacking.html*
- **Extractor – *http://www.extractor.com/***
  Commercial software for keyword extraction in different languages.
  Demo – *http://www.extractorlive.com/on_line_demo.html*
- **Keyphrase extraction algorithm Kea - *http://www.nzdl.org/Kea.***
  From version 4.0, Kea also provide automatic assignment of terms from controlled vocabularies. Developed at the University of Waikato, NZ.
- **Maui – *http://maui-indexer.googlecode.com/***
  Multi-purpose topic indexing algorithm described in this thesis. Suitable for automatic term assignment, subject indexing, keyword extraction, keyphrase extraction, indexing with Wikipedia, autotagging, terminology extraction. Developed at the University of Waikato, NZ.
  Sourceforge site – *http://maui-indexer.sourceforge.net/*
  Topic indexing blog – *http://maui-indexer.blogspot.com/*
- **TerMine – *http://www.nactem.ac.uk/software/termine/***
  A term extraction tool developed at the National Centre for Text Mining, UK.
- **Topia – *http://pypi.python.org/pypi/topia.termextract/1.1.0***
  Part-of-speech and frequency based term extraction implemented in Python.
  Demo – http://fivefilters.org/term-extraction/
- **Orchestr8 – *http://www.alchemyapi.com/api/keyword/***
  A commercial API for keyword extraction using statistical and natural language processing methods. Applicable to web pages and text files in several languages.
- **Wikipedia Miner – *http://wikipedia-miner.sourceforge.net/***
  University of Waikato's API for accessing Wikipedia data. Also provides a tool for mapping documents to relevant Wikipedia articles, similar to Wikifier.
  Demo 1 – *http://wdm.cs.waikato.ac.nz:8080/service?task=wikify*
  Demo 2 – *http://wdm.cs.waikato.ac.nz:8080/service?task=compare*
- **Wikifier – *http://www.wikifyer.com/***
  A demo of detecting Wikipedia articles in text developed at the Language and Information Technologies group at the Univ. of North Texas, US.

- **SEO keyword extraction** - ***http://seokeywordanalysis.com/seotools/***
  Online keyword and keyphrase extraction tool for search engine optimization.
- **Scorpion – *http://www.oclc.org/research/software/scorpion/default.htm***
  OCLC's tool for automatic classification of documents.
- **Tagthe.net – *http://tagthe.net/***
  A demo and API for automatic tagging of web documents and texts. Tags can be single words only. The tool also recognizes named entities, e.g. locations.
- **Yahoo term extractor –**
  http://developer.yahoo.com/search/content/V1/termExtraction.html
  Web-service based content analysis via term extraction; includes a demo.

G.2 Vocabularies and test data

- **LSCH – *http://id.loc.gov/authorities/search/***
  Library of Congress Subject Headings.
- **MeSH – *http://thesauri.cs.vu.nl/eswc06/mesh/rdf/meshdata.rdf***
  Medical Subject Headings thesaurus.
- **Agrovoc – *http://aims.fao.org/en/website/Download/sub***
  FAO's agricultural thesaurus. Info: *http://www.fao.org/agrovoc/*
- **HEP – *http://invenio-demo.cern.ch/help/hacking/bibclassify-hep-taxonomy***
  DESY's High Energy Physics thesaurus.
- **W3C's list of SKOS thesauri – *http://esw.w3.org/topic/SkosDev/DataZone***
- **Maui's datasets –**
  http://code.google.com/p/maui-indexer/wiki/MultiplyIndexedData
- **Keyphrase extraction data set –**
  http://aye.comp.nus.edu.sg/downloads/keyphraseCorpus/

G.3 Other resources

- **NLM Indexing Initiative *http://ii.nlm.nih.gov/***
  Website about the National Library of Medicine's project on automatic indexing using MeSH terms. Research details, evaluation and examples.
- **Dublin Core tools *http://dublincore.org/tools/***
  A list of tools for automatic extraction of Dublin Core metadata.
- **ASI resources – *http://www.asindexing.org/site/software.shtml***
  List of back-of-the-book indexing tools by American Society of Indexing.
- **ANZSI resources – http://www.anzsi.org/site/software.asp**
  List of indexing tools by Australian and New Zealand Society of Indexing.