UiO : Department of Informatics
University of Oslo

Content Categorization for Contextual Advertising Using Wikipedia

Ingrid Grønlie Guren

August 2, 2015

Content Categorization for Contextual Advertising Using Wikipedia

Ingrid Grønlie Guren

August 2, 2015

Abstract

Automatic categorization of content is an important functionality in online advertising and automated content recommendations, both for ensuring contextual relevancy of placements and for building up behavioral profiles for users that consume the content. Within the advertising domain, the taxonomy tree that content is classified into is defined with some commercial application in mind to somehow reflect the advertising platform's ad inventory. The nature of the ad inventory and the language of the content might vary across brokers (i.e., the operator of the advertising platform), so it was of interest to develop a system that can easily bootstrap the development of a well-working classifier.

We developed a dictionary-based classifier based on titles from Wikipedia articles where the titles represent entries in the dictionary. The idea of the dictionary-based classifier is so simple that it can be understood by users of the program, also those who lack technical experience. Further, it has the advantage that its users easily can expand the dictionary with desirable words for specific advertisement purposes. The process of creating the classifier includes a processing of all Wikipedia article titles to a form more likely to occur in documents, before each entry is graded to their most describing Wikipedia category path. The Wikipedia category paths are further mapped to categories based on the taxonomy of Interactive Advertising Bureau (IAB), which are categories relevant for advertising. The results of this process is a dictionary with entries connected to categories from the taxonomy, and forms the base of our classifier. Finally, we explored the possibilities of using Wikipedia's internal links to translate the English classifier's dictionary to a Norwegian dictionary.

The evaluation of the classifier was performed on `rappler.com` for the English classifier and `adressa.no` for the Norwegian classifier. The results of the classifiers were compared with a class tag within the url structure of published articles, and we could see that the classifiers were able to correctly categorize most articles. However, there is room for further improvement of the classifier in order to achieve higher evaluation scores. This is partly because our dictionary-based classifier is a one-to-many classifier, while we compare the results to a one-to-one classification.

Overall, we found that we are able to create a varied and thorough dictionary by just exploring the titles of Wikipedia articles, and that the classifier gives a good indication of the content of articles.

Acknowledgements

This study has been a collaboration project between the Department of Informatics at the University of Oslo and Cxense (http://www.cxense.com). It was started in the Spring 2014 and finished in August 2015.

First, I would like to express my gratitude to my supervisor Professor Aleksander Øhrn for all his incredibly important feedback and for all the advice he has given me through the process. His ideas for this project have been incredible valuable, and his comments have helped me every time I needed help with a problem. I am very grateful for his thorough scrutiny of the thesis. I would also like to thank his colleague Gisle Ytrestøl for all his help, including all the quick responses on emails, long discussions about implementations and his never-ending support and optimism about the project.

A special thanks goes to my study friends, especially Sindre, Frida and Karine, for constantly reminding me how little time I had left, but still supporting me with discussions and good company at the University. I would also like to thank my friend Elisabeth for motivating me every day and for never asking me to stop talking about my thesis. Your support has been beyond words.

Finally I wish to thank my family for all help, comments, discussions, feedback on what I've written, and most important; for supporting me every day. I could never have done this without any of you.

Ingrid Grønlie Guren
August 2, 2015

Contents

List of Figures

List of Tables

Chapter 1

Introduction

> *"The ultimate search engine would understand everything in the world. It would understand everything that you asked it and give you back the exact right thing instantly. You could ask 'what should I ask Larry?' and it would tell you."*
>
> – Larry Page, *The Guardian, May 2006*[29]

1.1 Motivation

Imagine the possibilities if your computer was able to understand what you wanted to do at all times. This could be a computer that knows your address so it can remind you to take the last bus home from friends, or it could read emails from your boss and remind you of deadlines. The computer would need to be intelligent to perform such tasks. The study of creating intelligent computer software is called *Artificial Intelligence* (AI) and is one of the most discussed fields in modern time.

There are some challenges before computers today are considered intelligent. One of the main challenges is the task of making computers understand natural language. This task is commonly called *Natural Language Processing* (NLP) and defined as the task of getting computers to perform useful tasks involving human language [14, p. 35].

Our idea is that computers may perform better in many settings if they are able to determine the meaning of a text. Thus, the goal of this study has been to develop an automatic content categorization which could take any article as input, and determine the most likely category based on its content. Our approach for determining the most likely category is by creating a dictionary-based classifier from Wikipedia, where the titles of Wikipedia articles are used as entries, and each entry is connected to one or more suitable categories describing the content of the Wikipedia article.

1.2 The Project

Automatic content categorization is a process where the text is categorized to the most describing category or categories from a set of desirable categories.

There are various ways of performing automatic content categorization. This project focuses on categorizing text based on which keywords occur in the text and these keywords' categories.

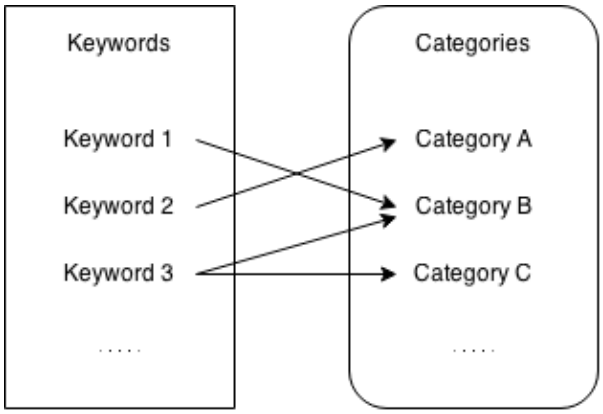


Figure 1.1: Illustration of the mapping between keywords and categories.

Creating this automatic content categorization consists of three main steps.

1. Create a list of keywords and a set of desirable categories for the categorization process. For this project, titles of Wikipedia articles are chosen as keywords, and the set of desirable categories is based on the taxonomy from *Interactive Advertising Bureau* (IAB). Both Wikipedia article titles and IAB's taxonomy need to be processed before they are suitable as keywords and category set.

2. Create a mapping between the keywords and the categories (see figure 1.1). This step takes advantage of the underlying structure of Wikipedia to determine the meaning of the Wikipedia articles, so that the keywords map to the category or categories that best describe their content.

3. Determine the category of any given text. Figure 1.2 shows this process, where all keywords are extracted from the given text and the text's category is determined from the keywords' categories. There are different ways of determining the category of a text. The extraction process could be exact matches of the keywords as they appear in our dictionary or by matching lemmas[1] where all inflections of words are considered equal. It is also possible to let the classifier determine the text's category based on different features; we could for instance count occurrences of all keywords leading to a category, or only count occurrences of unique keywords. The software for finding keywords in a text is provided by Cxense and is described in detail in section 2.5.

Why choose a dictionary-based classifier?

We chose a dictionary-based classifier because it is easy to understand for non-technical users. The users of our project are people without any specific knowl-

[1] Lemma is defined as the canonical form of a word [18, p. 30].

edge of categorization or computer science. The classifier uses categories written in natural language and gives output the users can understand without help from developers. It is also based on a dictionary, which is a familiar concept that might make it easier to understand the classification process. Another advantage with a dictionary is that it is easy to edit for users, which means that they can personalize the dictionary to fit their preferences by adding/removing entries.

Why use Wikipedia?

We chose to use Wikipedia titles for our classifier, for 3 main reasons.

1. Wikipedia is the largest online encyclopedia and is maintained by volunteers from all over the world.
2. Wikipedia contains a useful category structure where all articles are placed within categories descriptive of their content, and the categories form a structure which represents relations between the categories.
3. Wikipedia titles are words or phrases which are good keywords since they are found within other articles.

Access to the results

All results are based on Wikipedia, downloaded the 22nd of January 2015 from `http://dumps.wikimedia.org/enwiki`. Several programs were made for this project, and they can be found at `https://github.com/ingridguren/Master-Thesis-2015`.

Figure 1.2: Simplified illustration of the categorization process.

1.3 An Overview of Challenges

We encountered various challenges within different fields while working on this project. Some of the challenges were solved better than others. This section gives a short introduction to some of the most advanced challenges encountered that were more time consuming than the rest.

Representing the structure

The structure of Wikipedia is found in multiple files containing lots of information needed for the process of setting up the encyclopedia. The underlying

structure is quite complex and is poorly documented from a developer's point of view. The first challenge encountered was deciding which information we needed for our task and where it was found, i.e., which files. Another challenge was to determine how to represent information and to find suitable structures.

Encoding and character normalization

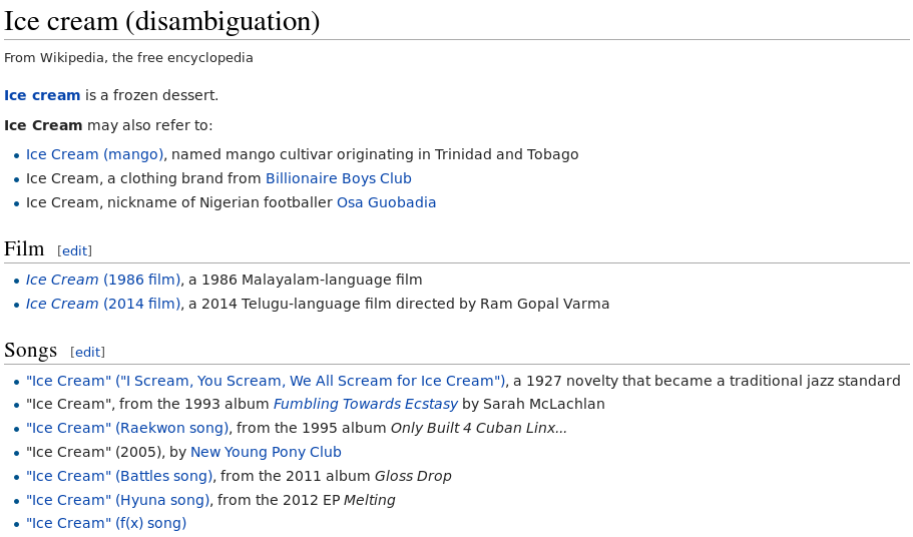
Wikipedia is available in multiple languages and is written by volunteers from all over the world. This makes Wikipedia a multilingual encyclopedia with knowledge available from everywhere since it is possible for experts from various fields and from different parts of the world to contribute with knowledge. There are both advantages and disadvantages with a multilingual encyclopedia. One of the disadvantages is that users might write with different encoding (e.g., *utf8*, *ascii* or *unicode*) because they use different keyboards and different languages. Problems occur when going through all the names of Wikipedia categories and Wikipedia article titles because titles written in different encoding might not be viewed as identical by the computer.

An example of a category name which lead to encoding trouble is *Communes in Caraș-Severin County*, which is either written with the letter *ș* (unicode character u\0218) [33] or ş (unicode character u\015e) [32]. These letters are examples of characters that makes matching of category names difficult, because *Communes in Caraș-Severin County* and *Communes in Caraş-Severin County* will not be equal to the computer even though it is clear to most users that they should be the same.

This problem was partly solved by changing all category names and article titles to the same encoding by transforming all text to *utf-8*, including escape of *unicode* characters with a python module *Unidecode 0.014.17* which transforms unicode characters to ascii [34]. The results from Unidecode was further converted to utf-8. This solved most of the problem, but some category names did not become equal even though most humans would consider them equal. A total of 10 800 categories was not able to be matched out of 519 822. These categories represent a very small part of all categories (equivalent to 2.1%), and were therefore disregarded.

Disambiguation

Antoher problem encountered is disambiguation. Wikipedia contains many titles that could have various meanings (see figure 1.3). This means that the titles are ambiguous and leads to the common problem in natural language processing: disambiguation [57]. A complete section (3.5) is dedicated to different solutions to this specific problem. However, our solution was to disregard all ambiguous dictionary entries if they were categorized to different categories.



Ice cream (disambiguation)

From Wikipedia, the free encyclopedia

Ice cream is a frozen dessert.

Ice Cream may also refer to:

- Ice Cream (mango), named mango cultivar originating in Trinidad and Tobago
- Ice Cream, a clothing brand from Billionaire Boys Club
- Ice Cream, nickname of Nigerian footballer Osa Guobadia

Film [edit]

- *Ice Cream* (1986 film), a 1986 Malayalam-language film
- *Ice Cream* (2014 film), a 2014 Telugu-language film directed by Ram Gopal Varma

Songs [edit]

- "Ice Cream" ("I Scream, You Scream, We All Scream for Ice Cream"), a 1927 novelty that became a traditional jazz standard
- "Ice Cream", from the 1993 album *Fumbling Towards Ecstasy* by Sarah McLachlan
- "Ice Cream" (Raekwon song), from the 1995 album *Only Built 4 Cuban Linx...*
- "Ice Cream" (2005), by New Young Pony Club
- "Ice Cream" (Battles song), from the 2011 album *Gloss Drop*
- "Ice Cream" (Hyuna song), from the 2012 EP *Melting*
- "Ice Cream" (f(x) song)

Figure 1.3: Example of disambiguation in Wikipedia.

1.4 Thesis Outline

We consider chapter 2 to be an *Introduction* to the project by describing the definition and purpose of content analysis. The chapter is called *Background Materials* because it also covers the basic material needed for understanding the purpose of the project as well as the methods used in the implementation. The background material includes a basic introduction to the *categorization problems* we want to solve, *Wikipedia* and its underlying structure, a brief introduction to the taxonomy of *Interactive Advertising Bureau* (IAB), and finally how our results are found with help from *Cxense*.

Chapter 3 is dedicated to *Related Works*, mostly concerned with Wikipedia categorization or extracting semantic knowledge from Wikipedia categories. The chapter also contains a discussion on whether knowledge from the previous works can be used in this project.

We consider chapter 4-5 as *Methods*. Chapter 4 focuses on the methods for representing the structure, grading different paths, and evaluating the results. Chapter 5 focuses on details of the implementation of the project, and gives a deeper discussion of the problems encountered and possible solutions. Chapter 5 describes the process of finding the full path of all Wikipedia articles in detail, how to determine the meaning of articles by grading the category paths, and the processes of mapping Wikipedia article titles to categories.

Results and Discussion are covered in chapter 6, including improvements of the implementation and discussion of the results. It also evaluates which categories are easily detected and compares our results with other text categorizations based on Wikipedia.

Finally, chapter 7 contains our *Conclusion* for the project; whether a text can be determined based on occurrences of Wikipedia article titles or not. The chapter also covers possible *Further Works* for obtaining even better results, and desirable features for the project.

Chapter 2

Background Materials

2.1 Automatic Content Analysis

2.1.1 What is Content Analysis?

Content analysis is the task of analysing and understanding collections of texts, in other words finding out what a text "is about". The task can be performed by both humans (manual content analysis) and computers (automatic content analysis), and both of the approaches have their advantages and disadvantages.

The concept of manual content analysis is easy. The task is split into first reading and understanding the text, then summarizing the content of the text and/or categorizing it into suitable categories describing the content. As an example, an article about *Ole-Johan Dahl* (the famous Norwegian computer scientist [48]) would probably be summarized as an article about a famous Norwegian computer scientist and might be categorized under the category *Norwegian computer scientists* if this category is present or the category *computer scientists* if this is present.

There are two main disadvantages of manual content analysis which makes it impossible to perform on large collections of texts. The first disadvantage is that the task is time consuming, i.e. it takes time for a human to read and understand an article. The second disadvantage is that manual content analysis requires resources that might be expensive, for instance experts needed for understanding the content of an article if the article is about something beyond common knowledge.

Automatic content analysis is based on a different approach; instead of reading and understanding the text, the machine looks for predefined properties of the text (in our case known words or phrases) and uses these properties to determine the meaning of the text. This requires some predefined connection between the properties and their associated categories. This approach has disadvantages as well; computers lack commonsense knowledge usually known to ordinary humans, for instance physical description or function of objects. Color is an example of a physical description computers have problems with determining. Most humans would understand that the phrase *same color as the sun* means yellow, while computers would need specific information about the sun being yellow to conclude the same.

Another disadvantage with automatic content analysis is dealing with disambiguation. Some words have more than one meaning, and the meaning is usually found from the context or the other words in the sentence. The task of determining the true meaning of a word or sentence is a difficult process which becomes harder if the sentences are complex.

2.1.2 Content Analysis in Advertising

Automatic content analysis can be found useful in many different settings, but two of the most dominating areas are advertising and improvement of user experience. The context of this project is to improve advertising, which makes advertising our domain.

Advertising is the main income source of most online companies that provide free services. The alternative to advertising is to charge users, which means that they have to pay a fee in order to use the services. The World Wide Web is very competitive and most users expect everything on the Internet to be free. For this reason the most common approach is to provide the services for free, and earn money on advertising instead.

There are mainly four different roles within online advertising [30]. These roles may overlap so that the same person or company can possess more than one role.

1. The advertisers, also called marketers, are people or a companies that have advertisements they want to display on webpages. The advertisers are willing to pay more for advertisements if the webpages are frequently visited or if the advertisements are displayed to users with a higher potential of buying the products.

2. The brokers, usually a third-party advertising company, manages the selection of advertisements and the placements of these. These companies collect information about the Internet users so that the advertisements are directed towards potential customers.

3. The publishers are people or companies in charge of a webpage with advertisement spaces. They sell the advertisement spaces, but the brokers are the ones responsible for choosing which advertisements to show.

4. Ad-Tech players are companies between the advertisers and the publishers. These companies get paid to provide information to optimize the advertising, which is profitable for the other roles within online advertising.

All roles within online advertising have a higher probability of earning money if the advertisements are chosen based on the interests of the users. This is called *Interest-Based Advertising (IBA)* or *Online Behavioral Advertising (OBA)*, where the advertisements are chosen depending on the user's interests or browsing history. Information about browsing performed by users are collected at all times so that advertisements displayed are more likely to be relevant for each user (see figure 2.1 [31]).

There are two different approaches of performing online advertising.

1. *Display Advertising* is a method where the advertiser pays for each display of the advertisement on a webpage. There are different ways of computing

Figure 2.1: Illustration of retargeting which is a advertising technique within Interest-based Advertising (IBA).

the cost of display advertising, but the most common is *Cost-Per-Mille* (CPM) or *cost-per-thousand impressions*. CPM is a metric where the advertiser pays for showing the advertisement to thousand viewers [13], and popular pages have a higher CPM than unpopular pages.

2. *Affiliate marketing* is based on the success of the advertisement. One of the most common approaches within this affiliate marketing is *Performance-based advertising* where the price of the advertisement is based on the interaction with the user, i.e., how successful the advertisement is [19]. There are different ways of measuring the advertisement's success. The most common ones are:

    - *Cost-Per-Click* (CPC) where the advertiser pays per click on the advertisement [8].
    - *Cost-Per-Action* (CPA) where the price of the advertisement is also based on the probability of a completed transaction [7].

All these approaches are more valuable for all roles in the advertising process if the advertisements are shown to people that are interested in the products and more likely to buy the product. Thus, our motivation is to improve advertising by creating a content classifier that categorizes text into suitable categories which is a great help when building up user profiles.

2.2 Categorization

Categorization is the process of grouping collections of text into categories, and can be done by both humans or computers. Computer categorization is the technique of teaching a classifier how to decide the category of any input [39]. The idea of this process is to find patterns which makes the machine able to predict the category or class of the input. Such patterns could be similarities between input or decision rules [44]. It is desirable to optimize the results of the classifier so that the classifier is as accurate as possible. This can be done by learning the classifier how to behave, either by machine learning where the

classifier optimizes itself based on feedback, or by improving the classifier's decision rules.

Our problem consists of two categorization problems:

1. Categorization of keywords.
2. Categorization of any text.

Categorization of keywords

The categorization of keywords is done by creating a keyword list based on titles of Wikipedia articles. These keywords have to be categorized to suitable output categories. This categorization could be split into two parts:

1. Categorize the keywords to Wikipedia categories represented as category paths (see figure 2.2). This categorization should be based on the content of the Wikipedia articles of the keywords. Our assumption is that the meaning of a Wikipedia article can be found by looking at the underlying structure of Wikipedia, i.e., the article's categories and the category structure.

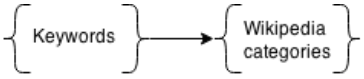


Figure 2.2: Illustration of the categorization of keywords to Wikipedia categories.

2. The complete categorization of the keywords are based on creating a connection between the keywords and categories from IAB's taxonomy (see figure 2.3). This categorization is based on rules between excerpts of Wikipedia category paths and the output categories.

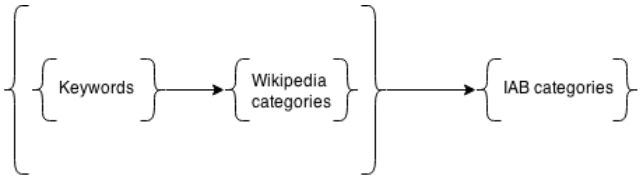


Figure 2.3: Illustration of the complete categorization process of the keywords.

Categorization of any text

The goal for this project is to be able to categorize any text based on the results from the categorization of the keywords. The classifier for this categorization process needs some rules on how it should classify. Our theory is that occurrences of keywords can determine the content of the text, and multiple keywords categorized to the same category indicate that the text should be categorized to this category. Thus, the classifier needs a way of detecting keywords in the text and a way of determining which category the text belongs to if it contains keywords from different categories.

2.3 Wikipedia

Wikipedia is a free, online encyclopedia and community that was created by Jimmy Wales in 2001. The encyclopedia is edited by the Wiki-principle, which means that everyone can create and edit articles. To understand the importance of Wikipedia it is worth mentioning that the web page has been ranked as the fifth globally most important web page (New York Times, February 2014), with more than 30 million articles and almost 500 million unique users a month [51].

Wikipedia contains a multitude of articles within many subjects and is maintained by thousands of people. Hence, the idea is to base the list on all the titles in Wikipedia, but the list has to be modified to contain only relevant titles. It is for instance not relevant to have common words in the keyword list which will occur in most articles and not provide any useful information. It is also important to remove or weight down ambiguous words, i.e., words that could confuse the categorization process or apply wrong information.

One of the main advantages of using Wikipedia is the underlying structure that is already provided. All articles are already categorized which gives information about the content of the article connected to the title.

2.3.1 Structure of Wikipedia

The structure of Wikipedia is web based, where articles with topical similarities are linked together. Since Wikipedia is language-based, articles only link to other articles within the same language, except for links to the same article in other languages. Wikipedia does also have a category structure where all articles are classified under at least one category. A category could have articles, but could also have subcategories, where the subcategories have their own articles and subcategories. Together they form a large category graph, which is an abstract structure that shows the relationships between the categories. All Wikipedia articles are placed under the most describing categories, as an example Ole-Johan Dahl is placed under the category *Norwegian computer scientists* which is under the parent category *Computer scientists by nationality* which is under *Computer Scientists*.

The category graph is created so there is a link between a category and each of its subcategories. There is no beginning of the category graph, but there are some categories which have most other categories as their subcategories. These can be thought of as *beginning categories*, also called *root categories*, and are important when we want to look through all categories in the graph and observe the relationships between them. Two categories that can be viewed as potential root categories are *Fundamental Categories* or *Main Topic Classifications*. If one of these are chosen as the root category, we can continue through the graph by looking at its subcategories and proceed by looking at each of the subcategory's subcategories an so on. One important thing is to avoid loops within the category graph, since it is possible for a category to reach itself by following its subcategories.

Figure 2.4 is an example of a structure for the category *Astrid Lindgren*, the Swedish author of children books. The figure shows a tree structure for the category from the category graph. The figure shows that the category *Astrid Lindgren* has 10 pages directly under the category, and 4 subcategories: *Astrid Lindgrens characters* (9 pages), *Films based on works by Astrid Lindgren* (1

subcategory and 23 pages), *Works by Astrid Lindgren* (2 subcategories and 7 pages) and *Pippi Longstocking* (1 subcategory and 10 pages). This means that under the category *Astrid Lindgren* there are directly or indirectly 59 or less pages (some pages might be placed under more than one category). This is without counting potential pages under the next level of subcategories.

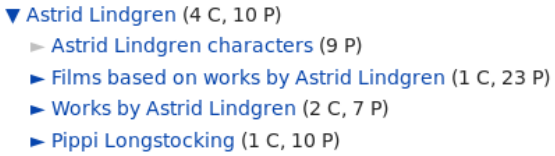


Figure 2.4: Subcategories of the category Astrid Lindgren.

Wikipedia articles are already classified under categories, but the set containing all Wikipedia categories cannot be used as a final categorization. The category set in Wikipedia is too large for such usage, where some categories do not provide information (e.g. *List of lists of lists*), and some are too descriptive for its content (e.g. *2009 Davis Cup Americas Zone Group I*). There are also cases where articles are categorized under categories where the combination of the categories does not provide any new information. An example is the article of *Ole-Johan Dahl*. Some of the article's categories are shown in figure 2.5. In this example the article is both placed in the category *People from Mandal, Norway* and in the category *Norwegian Computer Scientists*. These categories both provide information about him being Norwegian, so it would be sufficient to put him in the category *Computer Scientists*. The categories shown are also quite specific, and it might be desirable with more general categories.

Another reason for creating a new independent category set is that the Wikipedia categories are not guaranteed to be in the desirable final category set. Hence it is essential that the classifier creates a connection from the article and to a category that is know to exist in the set. The classifier should instead be based on the category information provided by Wikipedia.

Figure 2.5: Some of the categories for the article of Ole-Johan Dahl.

Instead of creating a categorization from the Wikipedia titles and to the most describing categories from Wikipedia's category set, we want to create a connection to a category in a predefined category set. This set of categories has to fit the advertising domain, i.e., contain categories relevant for advertising. It is also desirable to choose a category set that is so simple that any users of the program understand the categories.

2.3.2 Accessing Information from Wikipedia

There are two ways of accessing Wikipedia's encyclopedic information; the most common way is to enter the webpage and search for the information needed,

but it is also possible to download database dumps and access them directly to find information. All Wikipedia articles, images and categories are stored in a database which is accessed whenever a user searches for information online, and the information retrieved from the database is returned to the webpage, for instance in the form of an article. To ensure that all data are safe at all times, files containing the information needed to recover the database is stored and regularly updated [52]. This type of backup is called a database dump and is available for anyone interested at `http://dumps.wikimedia.org/enwiki` [36]. All Wikipedia files used for this project were downloaded 22nd of January 2015.

The files associated with the database dumps contain different information needed, i.e. some files contains all the articles' titles, some contain information about which images belong to which articles and so on. Together they provide all information needed to restore Wikipedia if data is lost.

Table 2.1 shows the files determined to be relevant for our task and a short description on what they contain.

File name	**Information contained**
`enwiki-latest-categorylinks.sql.gz`	Links between categories, and between categories and articles.
`enwiki-latest-page.sql.gz`	All pages in Wikipedia, including the type of page (category, article, user) and whether the page is a redirecting page or not
`enwiki-latest-page_props.sql.gz`	The properties of each page, including if the category is a hidden category or if the page a disambiguation page.
`enwiki-latest-redirect.sql.gz`	Redirects from Wikipedia pages and to other Wikpediapages.
`enwiki-latest-category.sql.gz`	Properties of all categories.
`enwiki-latest-langlinks.sql.gz`	Links from English Wikipedia pages to the same page in other languages.

Table 2.1: The relevant files from the English Wikipedia database dump and a short description of what they contain

2.4 Interactive Advertising Bureau (IAB)

The predefined category set should be well-defined and fit for the purposes of the task. Since the focus of this project is improving advertising, the predefined category set should be a category set useful for advertising.

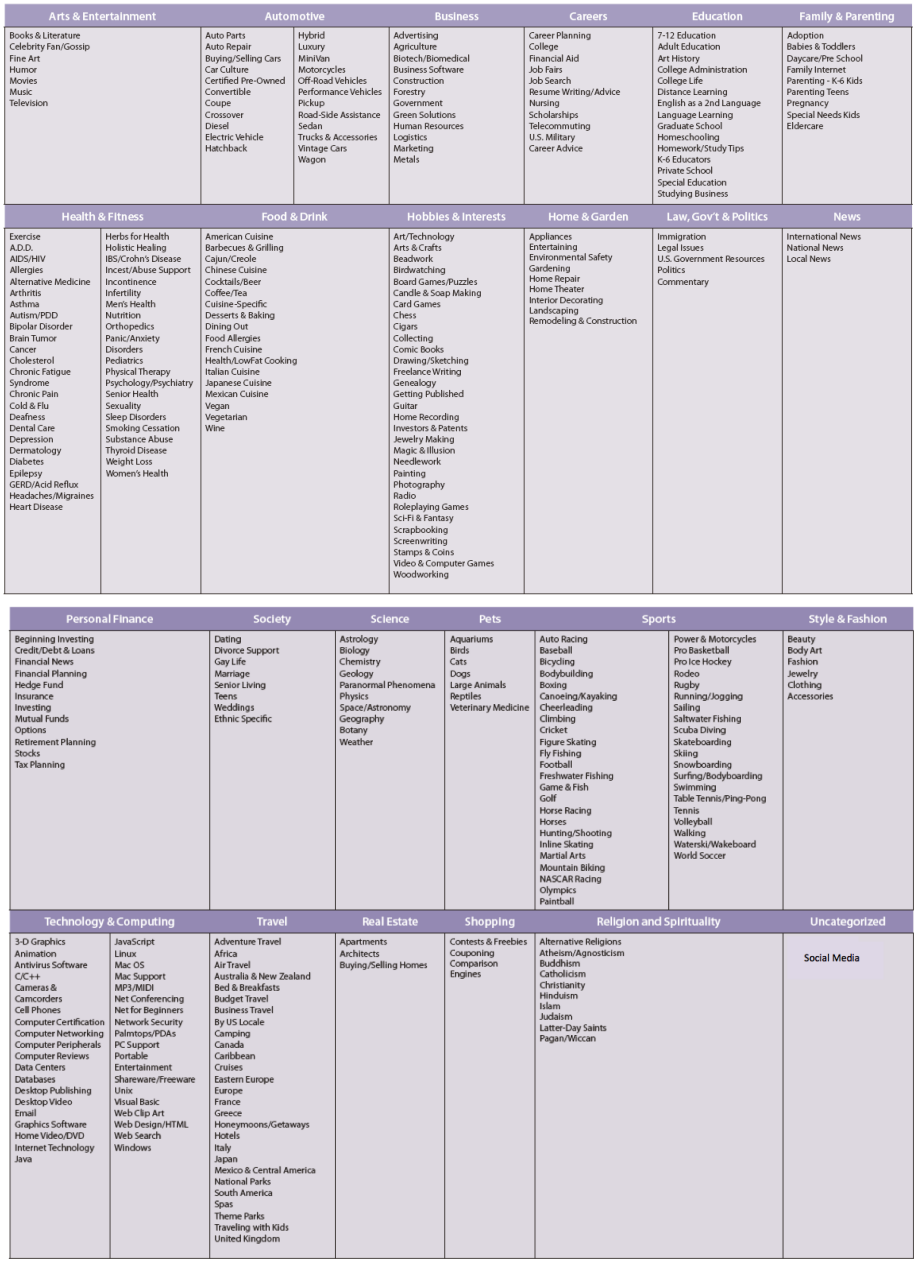
IAB is a business organization that develops, researches and maintains industry standards for the online advertising industry. The organization works for creating, coalescing and maintaining standards and practices in online advertising. In addition, IAB researches and shares knowledge on the advertisement,

and IAB members are responsible for distributing 86 % of all the online advertisement in the US [1].

IAB provides different guidelines for advertising, including *Quality Assurance Guidelines Taxonomy* (QAGT). This taxonomy is well-defined for advertising, and can be viewed as a category set. The set is split into two *layers* also called *tiers*. The layers are created for varying the grade of speciality between the first tier (a general or broad level) and the second tier (a deepening level). The first tier contains a total of 23 categories, examples are *Business* and *Food & Drinks*. The second tier contains 371 subcategories, where each subcategory is a more specific category of a category in the first tier.

Figure 2.6 shows the taxonomy of IAB as defined on their web page where the first tier is all the category names written in white (e.g., *Food & Drinks*) and the second tier is followed under the first tier (e.g., *American Cuisine*).

We wanted to use IAB's taxonomy as our output category set, i.e., the keywords are mapped to one or more categories in the category set.



Arts & Entertainment	Automotive		Business	Careers	Education	Family & Parenting
Books & Literature Celebrity Fan/Gossip Fine Art Humor Movies Music Television	Auto Parts Auto Repair Buying/Selling Cars Car Culture Certified Pre-Owned Convertible Coupe Crossover Diesel Electric Vehicle Hatchback	Hybrid Luxury MiniVan Motorcycles Off-Road Vehicles Performance Vehicles Pickup Road-Side Assistance Sedan Trucks & Accessories Vintage Cars Wagon	Advertising Agriculture Biotech/Biomedical Business Software Construction Forestry Government Green Solutions Human Resources Logistics Marketing Metals	Career Planning College Financial Aid Job Fairs Job Search Resume Writing/Advice Nursing Scholarships Telecommuting U.S. Military Career Advice	7-12 Education Adult Education Art History College Administration College Life Distance Learning English as a 2nd Language Language Learning Graduate School Homeschooling Homework/Study Tips K-6 Educators Private School Special Education Studying Business	Adoption Babies & Toddlers Daycare/Pre School Family Internet Parenting - K-6 Kids Parenting Teens Pregnancy Special Needs Kids Eldercare

Health & Fitness		Food & Drink	Hobbies & Interests	Home & Garden	Law, Gov't & Politics	News
Exercise A.D.D. AIDS/HIV Allergies Alternative Medicine Arthritis Asthma Autism/PDD Bipolar Disorder Brain Tumor Cancer Cholesterol Chronic Fatigue Syndrome Chronic Pain Cold & Flu Deafness Dental Care Depression Dermatology Diabetes Epilepsy GERD/Acid Reflux Headaches/Migraines Heart Disease	Herbs for Health Holistic Healing IBS/Crohn's Disease Incest/Abuse Support Incontinence Infertility Men's Health Nutrition Orthopedics Panic/Anxiety Disorders Pediatrics Physical Therapy Psychology/Psychiatry Senior Health Sexuality Sleep Disorders Smoking Cessation Substance Abuse Thyroid Disease Weight Loss Women's Health	American Cuisine Barbecues & Grilling Cajun/Creole Chinese Cuisine Cocktails/Beer Coffee/Tea Cuisine-Specific Desserts & Baking Dining Out Food Allergies French Cuisine Health/LowFat Cooking Italian Cuisine Japanese Cuisine Mexican Cuisine Vegan Vegetarian Wine	Art/Technology Arts & Crafts Beadwork Birdwatching Board Games/Puzzles Candle & Soap Making Card Games Chess Cigars Collecting Comic Books Drawing/Sketching Freelance Writing Genealogy Getting Published Guitar Home Recording Investors & Patents Jewelry Making Magic & Illusion Needlework Painting Photography Radio Roleplaying Games Sci-Fi & Fantasy Scrapbooking Screenwriting Stamps & Coins Video & Computer Games Woodworking	Appliances Entertaining Environmental Safety Gardening Home Repair Home Theater Interior Decorating Landscaping Remodeling & Construction	Immigration Legal Issues U.S. Government Resources Politics Commentary	International News National News Local News

Personal Finance	Society	Science	Pets	Sports		Style & Fashion
Beginning Investing Credit/Debt & Loans Financial News Financial Planning Hedge Fund Insurance Investing Mutual Funds Options Retirement Planning Stocks Tax Planning	Dating Divorce Support Gay Life Marriage Senior Living Teens Weddings Ethnic Specific	Astrology Biology Chemistry Geology Paranormal Phenomena Physics Space/Astronomy Geography Botany Weather	Aquariums Birds Cats Dogs Large Animals Reptiles Veterinary Medicine	Auto Racing Baseball Bicycling Bodybuilding Boxing Canoeing/Kayaking Cheerleading Climbing Cricket Figure Skating Fly Fishing Football Freshwater Fishing Game & Fish Golf Horse Racing Horses Hunting/Shooting Inline Skating Martial Arts Mountain Biking NASCAR Racing Olympics Paintball	Power & Motorcycles Pro Basketball Pro Ice Hockey Rodeo Rugby Running/Jogging Sailing Saltwater Fishing Scuba Diving Skateboarding Skiing Snowboarding Surfing/Bodyboarding Swimming Table Tennis/Ping-Pong Tennis Volleyball Walking Waterski/Wakeboard World Soccer	Beauty Body Art Fashion Jewelry Clothing Accessories

Technology & Computing		Travel	Real Estate	Shopping	Religion and Spirituality	Uncategorized
3-D Graphics Animation Antivirus Software C/C++ Cameras & Camcorders Cell Phones Computer Certification Computer Networking Computer Peripherals Computer Reviews Data Centers Databases Desktop Publishing Desktop Video Email Graphics Software Home Video/DVD Internet Technology Java	JavaScript Linux Mac OS Mac Support MP3/MIDI Net Conferencing Net for Beginners Network Security Palmtops/PDAs PC Support Portable Entertainment Shareware/Freeware Unix Visual Basic Web Clip Art Web Design/HTML Web Search Windows	Adventure Travel Africa Air Travel Australia & New Zealand Bed & Breakfasts Budget Travel Business Travel By US Locale Camping Canada Caribbean Cruises Eastern Europe Europe France Greece Honeymoons/Getaways Hotels Italy Japan Mexico & Central America National Parks South America Spas Theme Parks Traveling with Kids United Kingdom	Apartments Architects Buying/Selling Homes	Contests & Freebies Couponing Comparison Engines	Alternative Religions Atheism/Agnosticism Buddhism Catholicism Christianity Hinduism Islam Judaism Latter-Day Saints Pagan/Wiccan	Social Media

Figure 2.6: Categories of the IAB Taxonomy

2.5 Cxense

Cxense is a software company that collects and analyzes online information about Internet users. This information is used to create content profiles and user profiles, which can be used to understand the Internet activity. Their main goal is to understand the user's interest. Cxense provides software for companies, including tools to provide advertising, user recommendations and targeting emails [5].

This project is a collaboration project between University of Oslo and Cxense. Thus, Cxense's software is used in our process of categorizing texts. Figure 2.7 illustrates the complete categorization process of the keywords. When this process is complete, we have a list of categorized keywords which is needed for categorizing any input text. Cxense's software is part of our classifier in the process of finding keywords within the text.

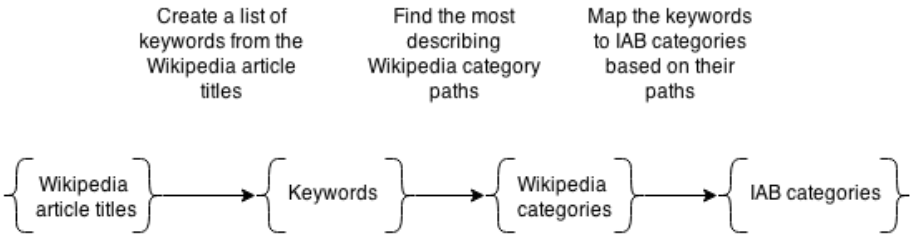


Figure 2.7: Illustration of the categorization process of the keywords.

There are different ways of finding the keywords in a text. Figure 2.8 illustrates the general approach where the whole dictionary is intersected with the document, and the result is a list of all entries found in both the dictionary and the document.

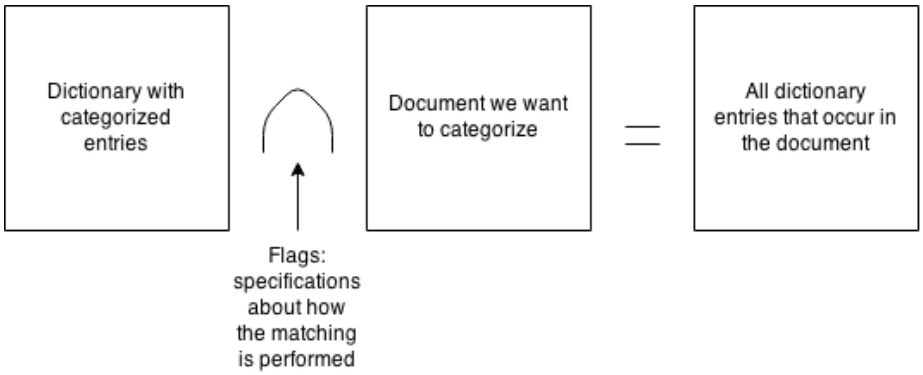


Figure 2.8: Illustration of the categorization of a text, where the classifier finds all dictionary entries that occur in the text.

Cxense's software allows the user to specify how the matching should occur. This is done by adding flags (specifications) in the intersection process. We

have chosen to use exact matching in our project which means that the words have to be identical to be considered a match. We have also chosen to use case insensitivity (words in lower case can be identical match to words in upper case) and normalization of the words. The normalization of the words were to make sure all words were in the same character encoding.

It is also possible to decide the lower limit of keyword occurrences from a category before the class is assigned to an article. This is done by creating weights of the keywords' classes and can be used to optimize the classifier.

Chapter 3

Related Works

Categorization is not a new topic, nor is taking advantage of Wikipedia in a categorization process. There are many papers about the topic, and we did some research to avoid problems already solved by others and to get inspiration for our project. This chapter is dedicated to projects with related topics. It starts by giving a short introduction to the projects we have studied.The projects concerned with Wikipedia's category structure is covered in 3.2, while section 3.3 is dedicated to the process of extracting keywords from Wikipedia. Section 3.4 covers classifiers based on Wikipedia, including the evaluation of the classifiers studied. Finally, section 3.5 gives an introduction to different types of disambiguation in NLP and reviews projects for solving disambiguation.

3.1 Similar Projects

Several projects have been studied in the process of creating a dictionary-based classifier. We have focused on 9 of the projects studied and grouped them within 4 different project topics (some projects are in more than one group):

1. Projects dedicated to understand Wikipedia's underlying category structure.
2. Projects that use encyclopedic information from Wikipedia to determine content.
3. Projects that use information from Wikipedia to create classifiers.
4. Projects for solving disambiguation.

WordNet

The WordNet project has become one of the most used knowledge resources in NLP. The project provides a semantic lexicon for English, which is useful for the computer in order to understand and tag sentences so that it can find the meaning of the sentences.

We have not studied or focused too much on the WordNet project since it mainly covers synset of words, and our main focus is not related to meanings of words. However, it is essential to mention the WordNet project since some of the related projects are based on or are extensions of WordNet [35].

3.2 Wikipedia's Category Structure

Wikipedia articles are placed within categories, and these categories form an underlying category structure by linking the categories together. The structure is created and maintained by many users all over the world. This means that the thoroughness of a specific part (e.g., links between categories or how specified the categories are) depends on the users responsible for the creation or maintenance. We use the Wikipedia category structure to determine the content of Wikipedia articles within our project by following the category links leading to Wikipedia articles. We have studied two projects that focus on understanding Wikipedia's category structure and the category relationships in order to create an improved or more accurate taxonomy:

1. *Decoding Wikipedia Categories for Knowledge Acquisition* [23] which focuses on understanding the conceptual relationships between category links in the structure.

2. *Extracting Semantic Relationships between Wikipedia Categories* [2] which focuses on the semantic relationships within the category graph.

The human made category structure might vary depending on the user that created it. [23] is a project for automatically understanding this structure, by sorting both the categories and category links into types which describes the purpose of the categories and the category links. Project [2] analyzes the links within the category structure for automatically understanding the categories that mean the same.

Relationships between categories

Relationships between Wikipedia categories are represented as category links. One may say that there exists two types of relationships within the category structure:

1. conceptual relationship

2. semantic relationship

Conceptual relationships are covered in the first project ([23]). This project focuses on relationship types represented in links between categories and articles, and between categories. Two links within the category structure can represent similar relationship types without having similar category names. Thus, an automatic approach for representing the category links in a standardized way was created in [23].

Semantic relationship is not necessarily represented within the structure of Wikipedia. These relationships occur between categories that have the same meaning. Project [2] covers an implementation for finding articles with the same meaning by looking at the category links in Wikipedia's category structure. The semantic similarity for an article is found by creating a *Semantic Connection Strength* (SCS) which represents the semantic connection to other articles. Their result is a semantic schema that retrieves the most relevant articles for a given word, without considering the word's syntax.

Our project does not consider semantic or conceptual relationships, but both of these projects provide useful information about the category structure and contain relevant ideas for further implementation. Applying semantic information could be very useful for categorizing, where keywords with high SCS could be categorized to the same categories. Conceptual relationships between categories could help the representation of the category structure and make the ranking of article paths easier.

3.3 Wikipedia as Encyclopedic Knowledge

Our main goal is to categorize any text based on keywords from our dictionary-based classifier. This requires a way of extracting keywords from Wikipedia. There exists various projects for marking Wikipedia entries in text and taking advantage of the Wikipedia's encyclopedic knowledge already since Wikipedia is a massive resource of encyclopedic knowledge. Some of these projects are:

- *Entity Extraction, Linking, Classification, and Tagging for Social Media: A Wikipedia-based Approach* [10] which extracts Wikipedia article titles in tweets for understanding their content.

- *Large-Scale Taxonomy Mapping for Restructuring and Integrating Wikipedia* [24].

- *Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge* [9].

Project [24] provides an extension to WordNet. It takes advantage of the semantic information from WordNet's synset[2] to automatically generate a taxonomy. The project's approach is to use an already created taxonomy based on Wikipedia; WikiTaxonomy [25]. The taxonomy is improved by linking the entries in the taxonomy to the synset from WordNet. These results are used to generate a new and improved Wikipedia taxonomy.

Encyclopedic knowledge from Wikipedia is also found in [9]. This project creates a classifier that is extended with knowledge from Wikipedia. Their assumption is that each Wikipedia article represents a concept and that documents are placed within a feature space of Wikipedia concepts and words.

The last project covered here is [10], which is a project that creates a dictionary-based classifier based on knowledge from Wikipedia. This project has a goal very similar to ours; to categorize tweets[3] based on their content. The solution implemented for this problem was to use Wikipedia as a knowledge base, where Wikipedia articles are connected to concepts used in the classification process. [10] describes an approach with lots of preprocessing of both tweets and the Wikipedia concepts.

[2]Definition of synset from WordNet: "Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations."[35]

[3]Messages on Twitter (social media).

3.4 Classifiers Based on Wikipedia

We have created a dictionary-based classifier, which classifies text based on occurrences of entries in our dictionary. This is just one way of classifying text. Classifiers can be created in various ways, and the classifiers can focus on different features. We have studied some projects which creates classifiers from Wikipedia:

1. Dictionary-based classifier: *Identifying document topics using the Wikipedia category network* [28] and *Entity Extraction, Linking, Classification, and Tagging for Social Media: A Wikipedia-based Approach* [10].

2. Classifier based on Bag of Words: *Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge* [9].

3. Statistical classifier: *Automatic ontology extraction for document classification* [16].

Dictionary-based classifiers

One of the most relevant project regarding our project is [28]. This project is closely related to our research, with a similar goal; to determine whether documents can be categorized by only exploring titles and categories of Wikipedia articles.

The main difference between this project and ours, is their choice of output categories. [28] categorizes documents to Wikipedia categories, while we categorize documents to a category set based on IAB's taxonomy. Their categorization approach are similar to ours, and consists of two main steps:

1. Look for word compounds within the text that match processed titles of Wikipedia articles.

2. Retrieve the Wikipedia articles' categories.

The classifier in [28] is a dictionary-based classifier like ours, but the keywords are categorized to the corresponding Wikipedia articles' categories instead of an independent category set. Another difference is that we look at the whole category structure, while [28] looks at categories retrieved from the matched Wikipedia article titles.

Another dictionary-based classifier is found in [10], which is a project for classifying and tagging tweets. The project uses Wikipedia to create a knowledge base, where they process titles of Wikipedia articles and link them towards suitable categories representing the content of their article.

The project concluded that Wikipedia did not have coverage for classifying all tweets, and added more concepts and instances to the knowledge base or better classification results. This is interesting for our project since we create a dictionary-based classifier solely from Wikipedia.

Bag of Words (BOW)

One of the most common ways of classifying text is by representing the text as a *Bag of Words* (BOW). The idea is that the classifier looks at which words occur within the document and classifies the document based on the frequencies of these words. The BOW does not consider the order of the words, but only counts the occurrences. BOW can be advanced by weighting words so that common words have a smaller impact on the classification, and topic specific words have a larger impact.

One of the disadvantages with a classifier based on BOW, is that the classifier has problems with classification of short documents where there are few occurrences of all words, and small categories which have few connected keywords. Project [9] focuses on optimizing the BOW classifier on small classes and short documents.

The project created a program that finds the Wikipedia article most similar to the document, and extends this document with the words occurring in the Wikipedia article. This approach gives more topic specific words to the documents, which makes it easier to classify them with a simple classifier.

Statistical classification

Another way of classifying documents is by statistical classification. This approach is part machine learning where the classifier learns how to optimize its classification by using a training set. There exists various techniques within statistical classification, including *Support Vector Machine* (SVM). SVM is a method within supervised learning[4] where the classifier uses a training set to create a separation line (for 2 classes) or a hyperplane (for more than 2 classes). This line or hyperplane is used to separate classes.

Classification based on SVM is found in project [16], a project that focuses on ontology[5] extraction to improve classification. The project uses ontology to understand the semantic and syntactic relationships within Wikipedia, and creates a hyperplane to separate the classes. Many texts should be categorized to more than one class if the content is about more than one topic. The project's solution is to let the classifier create a hyperplane that correctly classifies most of the training data, but still lets some of the data be categorized to wrong classes.

The results of the ontology extraction in [16] is a interesting feature for future works with our implementation. Automatically understanding the concepts within Wikipedia could create a better taxonomy and improve the classification results.

3.4.1 Evaluation of the Classifiers

It is essential to evaluate the classifier to determine if it behaves as desired. Evaluation is therefore one of the most important parts of the categorization process. There are different ways of evaluating classifiers, but the best results are usually found when comparing with the *correct* results. We have collected

[4]Supervised learning is based on training sets which contain the correct classification results. Thus, the classifier receives feedback on its classification and can optimize the classification process.

[5]Ontology can be defined as an explicit specification of a conceptualization [11].

the evaluation techniques of the different classifiers and looked at what they have evaluated.

Evaluation measures

The evaluation measures for the classifiers have been *precision*, *recall* and F_1-*score* for [28], [10], [9] and [16][6]. All the projects have chosen a micro average evaluation in their evaluation, which means that they find the evaluation measures individually for each class.

What have been evaluated

Another way of evaluating our classifier's results is by comparing its results with the results of other classifiers. Thus, it is relevant to see what the other classifiers evaluated.

The categorization evaluation of [10] is based on topics within the tweets. Some of these topics are also categories within IAB's taxonomy, and it is possible for us to compare our evaluation results with results of the project. It is, however, important to remember that [10] added information to their knowledge base from other places than Wikipedia, which means that their classifier might have entities not available to our classifier.

The evaluation of [28] is split into evaluation of two classification experiments;

1. Classification of Wikipedia articles based on their text bodies.

   The articles chosen for this evaluation were not related to advertisement and not a priority for our classifier.

2. Classification of two independent corpora; 20 Newsgroups and RCV1.

   This classification was based on a training set, and again not well-suited for comparison.

The evaluation results in [16] is based on a training set with different sizes, and not suited for comparison with our results.

3.5 Disambiguation

Most people would prefer to interact with their computer in natural language, e.g., search for *"What is computer science?"* rather than *"computer science definition"*. We have already mentioned that the task of understanding natural language is called *Natural Language Processing* (NLP) and is a difficult task because it requires the computer to actually understand the meaning of text. This task is especially difficult because of ambiguity.

Ambiguity means that there are more than one meaning to a word, phrase or sentence, and disambiguation is the task of finding the correct meaning. There exists many different types of ambiguity [14, p.100 and p.466-468]

[6] The formulas for these evaluation measures are presented in section 4.3.

- Part-of-speech ambiguity where a part of the sentence is ambiguous. Example: *book* could be either a noun (*hand me that* book) or a verb (book *that flight*).
- Structural ambiguity where the structure of the sentence is ambiguous. This can be split into further types
  - Attachment ambiguity where it is not clear how the words are connected together. Example: *We saw the Eiffel tower flying to Paris.*
  - Coordination ambiguity where sets of phrases are joined by conjunction. Example: *Old men and women.*
- Local ambiguity where some part of the sentence is ambiguous even if the whole sentence is not ambiguous.

Many sentences in natural language are complex and combine the different types of ambiguity. This makes it hard for the computer to determine the meaning of the sentence.

Ambiguous keywords are a problem for our classifier, which means that part-of-speech ambiguity the most relevant for our project. Our solution was to remove all all ambiguous keywords from our dictionary-based classifier, but a good extension for our implementation is to handle disambiguation in a better way. Hence, we have examined some projects for resolving ambiguity:

1. *Named entity disambiguation by leveraging wikipedia semantic knowledge* [12].
2. *Large-scaled named entity disambiguation based on Wikipedia data* [4].
3. *Distributed Representations of Words and Phrases and their compositionality* [22].

Ambiguous entities

Project [12] and [4] encounter the same disambiguation problem as in our project; entities that have various meanings. The idea of their solution is to measure similarities between occurrences of names and use this to determine whether two occurrences of a specific name represent the same entity. Both projects look through internal hyperlinks in Wikipedia articles and collect all surface forms[7] of each article (entity).

In addition, [4] finds both semantic relations and social relatedness between Wikipedia in the task of determining the meaning of an entity. This is done by studying hyperlinks between them. The combination of these three factors form a way of avoiding ambiguity, since the most likely meaning is set for each Wikipedia article. Project [12] solves ambiguity by also looking at titles at the disambiguation pages and the redirecting pages, and represent the Wikipedia entities as vectors in a vector space model.

The last project studied for solving disambiguation is provided by Google in [22]. This project used a different approach, where they created an improved Skip-gram model. All words are represented in a vector space, with semantic

[7]Surface form is defined as full name, acronym, alternative names and spelling variations that occur for an Wikipedia article title

and syntactic relationships represented between the words. The training of the Skip-gram model made it possible to determine the meaning of the words based on the semantic relationships to other words within the vector space.

Chapter 4

Methods

This chapter can be viewed as an introduction to the methods we chose for the implementation of the project. It gives a brief introduction to how we determined the meaning of Wikipedia articles and the structure chosen for representing the information.

4.1 Finding the meaning of Wikipedia Articles

It is essential to know the meaning of the Wikipedia articles to be able to categorize them. Our assumption is that the meaning of Wikipedia articles can be found by looking at the categories leading to the article in the underlying category structure of Wikipedia. We base this assumption on the fact that all Wikipedia articles are placed under at least one category, and that the articles' categories should be representative for the article. This means that we need to find a representation for the underlying structure and a way of deciding the best way of reaching each article within this structure.

4.1.1 Representing the underlying structure

Taking advantage of the underlying structure of Wikipedia requires a way of representing it. Each category has links to its subcategories, and links to the articles which are placed under the category (see figure 4.1). Representing the structure could be split into two parts; one structure representing the underlying category structure (see figure 4.2), and one structure representing the categories of each article (see figure 4.3).

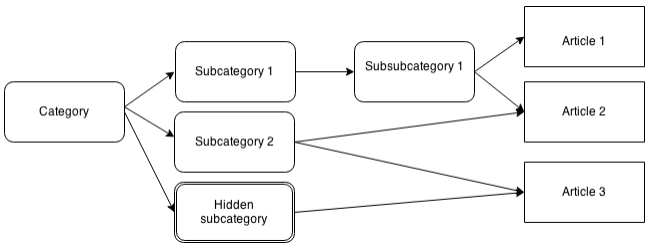


Figure 4.1: Simplified illustration of the underlying structure of Wikipedia.

Category graph

A category graph is a way of representing the links between categories. This structure contains information about which subcategories can be reached from each category. Figure 4.2 illustrates the category graph for representing the structure found in figure 4.1. The nodes in the graph (rectangles with rounded corners) represent categories, and the edges (arrows) represent the relationships between categories. The graph illustrated is a directed graph since each edge represents the relationship between the two categories (e.g., *Subcategory 1* is subcategory of *Category* since the arrow points from *Category* to *Subcategory 1*).

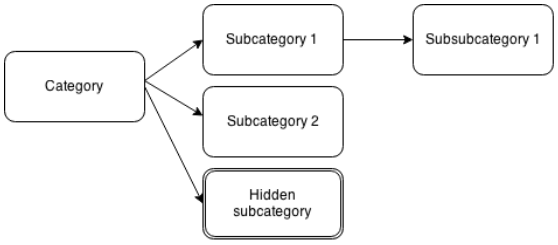


Figure 4.2: The structure where each category knows its subcategories

Article graph

A similar structure is desirable for representing articles and their most describing categories (the categories shown at the bottom of the article page in Wikipedia). Figure 4.3 illustrates how we represent each category's immediate articles by creating edges (arrows) between categories and their articles.

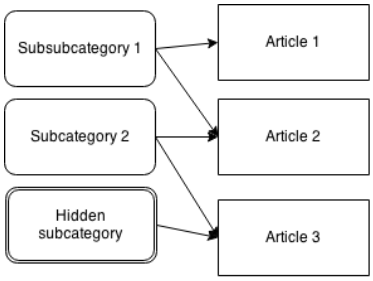


Figure 4.3: The structure where each category knows the title of its articles

Representing category and article names

Id mapping is a storage efficient way of representing category names and article titles. Category names and article titles are usually longer than their representing ids because ids can be chosen as increasing digits. The id mapper is implemented by creating a counter that assigns a unique number to each category name or article title not already observed.

4.2 Grading Categories

Many Wikipedia articles can be reached from categories that are not descriptive of the content of the article. We found multiple paths to all Wikipedia articles, but some of them were less descriptive than others. Thus, a grading was done to find the most relevant paths for each article.

4.2.1 Grading based on Inlinks and Outlinks

Each category in Wikipedia has a set of super categories (categories that lead to the current category), and a set of subcategories (categories that can be reached from the current category). The super categories and subcategories leading to the current category form a set. The size of this set can be annotated as

- *Inlink number* = number of super categories (parent categories)
- *Outlink number* = number of subcategories (child categories)

Figure 4.4 illustrates how inlink number and outlink number are connected to a category.

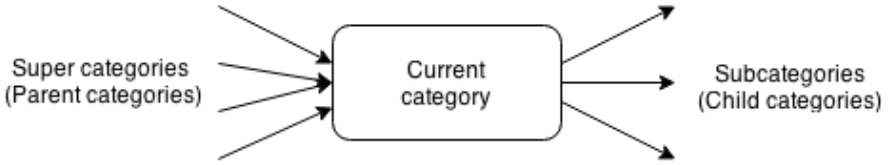


Figure 4.4: Example of how a category has links from parent categories and links to its subcategories. The inlink number *for the category is 4 and the* outlink number *for the category is 3.*

We created two assumptions based on this:

1. Categories with high inlink number can be reached from categories that are not about the same topic.
2. Categories with a high outlink number are more likely to reach articles not necessarily connected to the category name since they can reach far in all the subcategories' directions.

All categories should be given a score based on their inlink and outlink number, where low score values are given to categories within narrow topics (low inlink and outlink number), and higher score values are given to categories that cover more general topics (high inlink and outlink number).

Scoring paths

Grading based on inlink and outlink number is done by finding the inlink and outlink number for all categories in the structure, and by finding the average inlink and outlink numbers for all categories. The scoring is weighted based on the values of the inlink and outlink numbers. This gives the following formula

(equation 4.1) for finding the score of a given category, where $\overline{C_{in}}$ is the average inlink number and $\overline{C_{out}}$ is the average outlink number.

$$Score_C = \frac{inlink_c + outlink_c}{\overline{C_{in}} + \overline{C_{out}}} \tag{4.1}$$

This means that the path score of a path P is the sum of all scores for each category in the path (see equation 4.2).

$$Pathscore_P = \sum_c Score_C \tag{4.2}$$

The problem with equation 4.2 is that short paths will be favored since there are fewer scores to be added together. A way of avoiding favoritism of short paths is by normalizing the path scores.

4.2.2 Normalized Grading based on Inlink and Outlink Numbers

Grading based on inlink number and outlink number favors short paths even if the paths contain categories considered as bad. One way of handling this problem is by normalizing the score of each path. Equation 4.3 is a way of normalizing the path score of path P so the length of the path does not determine the relevance of the path, i.e.,

$$Pathscore_P = \frac{1}{N} \sum_c Score_C \tag{4.3}$$

where N is the number of categories in the path.

4.2.3 Deciding Relevant Paths

There are different ways of deciding the relevant paths among all graded paths. One way is by choosing a threshold for the path score. If the path score is lower than a given threshold, it is marked as relevant, while a path score higher than the threshold means that it is not relevant. A threshold can be found by deciding how many paths should be considered relevant.

One way of doing this is by finding the scores of all paths and sorting the scores from lowest to highest (see 4.4). Then a k has to be decided to how many paths are believed to be relevant of all paths, for instance one could assume that only 10% of the paths are relevant, which leads to $k = .10 \cdot n$.

$$Sorted\_scores = [S_1, S_2, ..., S_k, ..., S_n] \tag{4.4}$$

$$T = Sorted\_scores[k] \tag{4.5}$$

The problem with this method is that not all articles are guaranteed to have any relevant paths. The other problem is that the score of the path will vary a lot within different fields, since some of the Wikipedia articles are categorized under very specified categories.

Another approach is to choose the best k paths for each Wikipedia article. This approach is independent of the values on other articles' path score which

means all Wikipedia articles are guaranteed at least one path. The disadvantage is that some paths might be marked as relevant even though their path score is lower than path scores marked as irrelevant by other articles. Another disadvantage is that articles with many good paths will still have to choose the best k paths and good paths might be lost.

4.3 Evaluation

An evaluation of the categorization process is essential to know whether the classifier classifies correctly or not. This can also be used to find which categories are easy to classify, and which categories are difficult to classify. The evaluation is based on comparing the results with the correct results (called *Gold Standard* [18, p. 140]). The gold standard in our project is found in the url of articles, and is decided by the journalists when they publish articles. An article about sport contains *sports* in the url, for example *http://www.rappler.com/sports/by-sport/boxing-mma/pacquiao/90563-mayweather-sr-blasts-ariza*.

4.3.1 Evaluation of the Classifier

Evaluation of the classifier depends on more than just the number of correctly classified categories. A classifier that classifies all elements to all classes are not always considered good. There exists numerous ways of evaluating correctness of the classifier. We have chosen *Rand Index accuracy*, *precision*, *recall* and F_1-*score*. These evaluation measures depend on terms for calculating the results (see table 4.1). All the measures range from 0 to 1, where 0 is worst and 1 is best.

Term	Description
True Postive (TP)	Correct: Text is classified to the class by both classifier and *Gold Standard*.
True Negative (TN)	Correct: Text is neither classified to the class by the classifier, nor by *Gold Standard*.
False Negative (FN)	Incorrect: Text is not classified to the class by the classifier, but by *Gold Standard*.
False Positive (FP)	Incorrect: Text is classified to the class by the classifier, but not by *Gold Standard*.

Table 4.1: Explanation of the True Positive, True Negative, False Negative *and* False Positive *[18, p. 330-331].*

Rand Index Accuracy

Rand Index (RI) *accuracy* (also called *accuracy*) measures the percentage of decisions that are correctly classified by the classifier [18, p: 330]. Equation 4.6 shows how this is computed for evaluating the classifier [15].

$$\begin{aligned} \text{accuracy} &= \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{all documents}\}|} \\ &+ \frac{|\{\text{irrelevant documents}\} \cap \{\text{not retrieved documents}\}|}{|\{\text{all documents}\}|} \\ &= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \end{aligned} \tag{4.6}$$

Precision and Recall

Another way of evaluating the classifier is by using *precision* and *recall* which measures how many elements are correctly categorized and how many of the correct elements were found.

Precision is defined as in equation 4.7 which measures the fraction of returned results that are relevant [18, p. 5]. This means that precision can tell how many of the returned articles were categorized to the correct class.

$$\begin{aligned} \text{precision} &= \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|} \\ &= \frac{\text{TP}}{\text{TP} + \text{FP}} \end{aligned} \tag{4.7}$$

Recall is a measure of finding how many of the relevant documents were found by the classifier [18, p. 5], which is found in equation 4.8.

$$\begin{aligned} \text{recall} &= \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|} \\ &= \frac{\text{TP}}{\text{TP} + \text{FN}} \end{aligned} \tag{4.8}$$

A good classifier should have both high precision and recall. F_1-score is a combination of precision and recall which gives a measure of the overall evaluation of the classifier based on a the results of precision and recall. The F_1-score is defined as in equation 4.9.

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{4.9}$$

4.3.2 Optimizing the Classifier

The measures for evaluation are used to determine how well a classifier performs and to determine how the classifier could be optimized to perform better. A perfect classifier categorizes all documents to their most describing classes without classifying documents to classes they don't belong to. Figure 4.5 illustrates a perfect classifier which classifies all documents to their correct classes. The classification results can be seen in table 4.2.

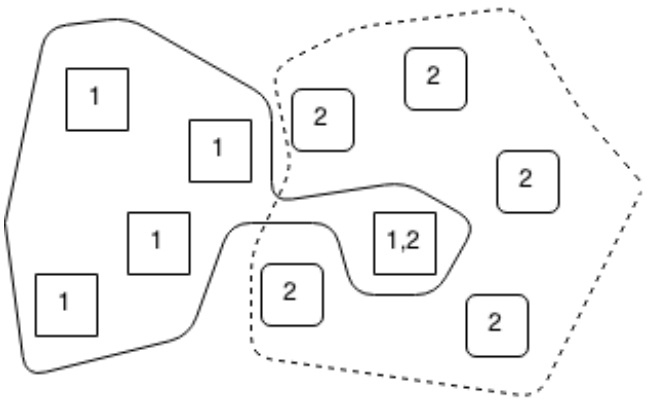


Figure 4.5: Illustration of a perfect classifier.

Perfect classifier: results			
TP	5	**Precision**	1
TN	5	**Recall**	1
FP	0	**Accuracy**	1
FN	0	F_1**-score**	1

Table 4.2: Classification results for class 1 for a perfect classifier.

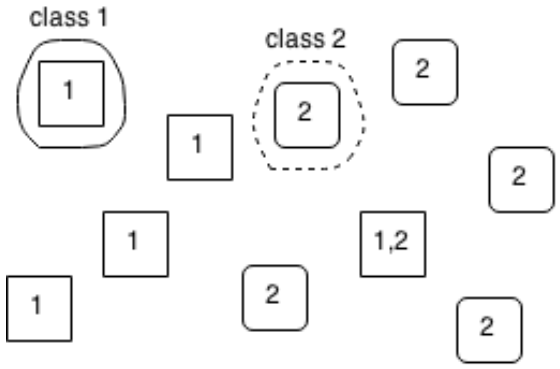
Why we need more than one measure for evaluation

Creating a perfect classifier is difficult, and it is difficult to determine if the classifier performs well. The different measurements for evaluation are best when they are combined (as F_1-score), because accuracy, precision and recall can have good results separately even if the classifier is far from perfect.

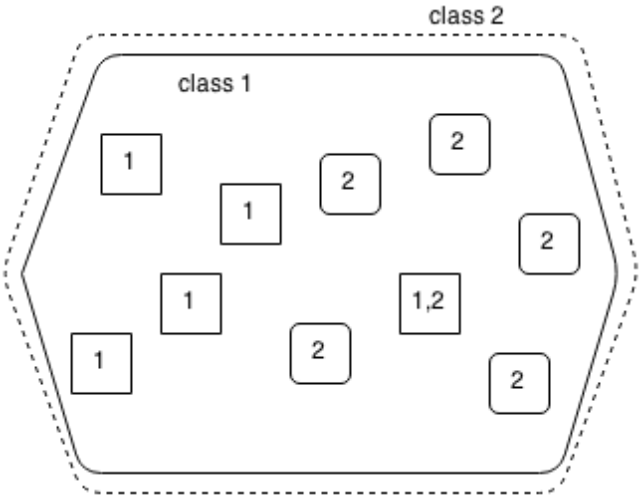
Figure 4.6a) and 4.6b) illustrates classifiers that have respectively high precision and high recall. Their results can be found in table 4.3 where we can see that high precision can be found by a classifier that only retrieves a few results and high recall is found for a classifier that retrieves many results. Thus, a good classifier should be neither of these, but instead balance the results.

Classifier 1				**Classifier2**			
TP	1	**Precision**	1	**TP**	5	**Precision**	0.5
TN	5	**Recall**	0.2	**TN**	0	**Recall**	1
FN	4	**Accuracy**	0.6	**FN**	5	**Accuracy**	0.667
FP	0	F_1**-score**	0.333	**FP**	0	F_1**-score**	0.667

Table 4.3: Evaluation of classifier A and B for class 1.



(a) Classifier A: Illustration of bad classifier with high precision.



(b) Classifier B: Illustration of classifier with high recall.

Chapter 5

Implementation

This chapter describes details of our implementation for creating a dictionary-based classifier. Several programs were made to achieve this goal, and many of the programs depend on the results from other programs. This chapter covers all the phases of the implementation in the order they were implemented. It starts with section 5.1 which describes how we found the full path of all Wikipedia articles from the Wikipedia database dump, including how to remove hidden categories from the paths and how to handle redirects. Further, the chapter describes the id mapping process (section 5.2) and compares this implementation with an implementation where full category names and article titles were used instead of ids. Section 5.3 covers two ways of deciding relevant categories for each article; grading the category paths based on inlink and outlink number (subsection 5.3.1) and grading based on normalized inlink and outlink number (subsection 5.3.2). The mapping between Wikipedia article titles and IAB categories are described with two approaches; mapping based on Wikipedia categories (subsection 5.4.1) and mapping based on Wikipedia path excerpts (subsection 5.4.2). Finally, section 5.5 describes the process of creating a dictionary-based classifier for other languages based on the English classifier and describes how we created a dictionary-based classifier for Norwegian.

5.1 Finding Full Path of Articles

The goal of our implementation was to create a dictionary where the entries are created from the titles of Wikipedia articles, and each entry leads to one or more describing categories. Wikipedia already contains an underlying category structure which is useful to decide the content for each article. Thus, the first step was to find the full paths of each article in Wikipedia, where the paths are given from the categories that lead to the articles.

5.1.1 Creating the Underlying Category Structure

Finding full paths of all articles require information about Wikipedia's structure between categories, and between categories and articles. Thus, we needed a way of representing the available information about the Wikipedia structure. The file `enwiki-latest-categorylinks.sql.gz` contains the information needed

to create a database table *categorylinks* filled with all links between categories, all links between articles and files, and all links between categories and articles. All information about the links are inserted in the database table through `INSERT` statements where all entries are on the form

(*cl_from*,*cl_to*,*cl_sortkey*,*cl_timestamp*,*cl_sortkey_prefix*,*cl_collation*,*cl_type*).

Table 5.1 describes the meaning of all the `INSERT` statement fields [20] and figure 5.1 shows an example of an entry in a `INSERT` statement, where the link between the category with id *12* and the page *Anarchism* is inserted into the table *categorylinks*.

Entry field	**Description**
cl_from	Stores the page.page_id of the article where the link was placed.
cl_to	Stores the name (excluding namespace prefix) of the desired category
cl_sortkey	Stores the title by which the page should be sorted in a category list.
cl_timestamp	Stores the time at which that link was last updated in the table.
cl_sortkey_prefix	Empty string if a page is using the default sortkey or readable version of cl_sortkey.
cl_collation	What collation is in used.
cl_type	What type of article is this (file, subcat (subcategory) or page (normal page)).

Table 5.1: Description of entry fields in `INSERT` *statements in* Categorylinks.

```
(12,'Anarchism',' \nANARCHISM','2014-11-20 17:57:05',
' ','uppercase','page')
```

Figure 5.1: Example of an `INSERT` *statement entry in* `enwiki-latest-categorylinks.sql.gz`. *The link is between the category with id* 12 *and the page* Anarchism.

Each `INSERT` statement consists of multiple links for insertion in the database table as we can see in figure 5.2. The `INSERT` statements has to be split so that all links are separated into new statements, and only relevant links are kept, i.e., links with *cl_type* as *subcat* (link between categories) or *page* (link between category and article). The file `enwiki-latest-categorylinks.sql.gz` contains 10 938 `INSERT` statements, which in total consists of 88 172 914 links. Table 5.2 shows the number of different links found in the file.

```
INSERT INTO `categorylinks` VALUES
(0,'','','2014-01-16 15:23:19','','','page'),
(10,'Redirects_from_moves','ACCESSIBLECOMPUTING
   ','2014-10-26 04:50:23','','uppercase','page'),
(10,'Redirects_with_old_history','ACCESSIBLECOMPUTING
   ','2010-08-26 22:38:36','','uppercase','page'),
(10,'Unprintworthy_redirects','ACCESSIBLECOMPUTING
   ','2010-08-26 22:38:36','','uppercase','page'),
(12,'Anarchism',' \nANARCHISM','2014-11-20 17:57:05','
    ','uppercase','page')
```

Figure 5.2: Excerpt from the file `enwiki-latest-categorylinks.sql.gz` *where each* `INSERT` *statement contains many links.*

Links between	**Number of links**
categories	1 648 873
categories and articles	21 846 996
articles and files	5 262 146

Table 5.2: Number of links found within the different link types.

Some of the links between categories are links for maintaining the encyclopedia. The categories with this purpose are called *hidden categories* and should be removed in our project in order to reduce the complexity.

Removing hidden categories

Wikipedia's category structure contains lots of hidden categories which are not displayed as categories at the bottom of an article page for the general users, even if the article is placed under the category. Examples of some hidden categories are *All articles with dead external links*, *Wikipedia Articles needing rewrite* and *Wikipedia articles in need of updating from September 2014*. Hidden categories are useful for editing articles or maintaining a trustworthy and well-structured encyclopedia. These categories provide an easy way to mark all categories with something in common, for instance mark all categories with references that needs to be checked.

Hidden categories are concerned with maintenance and administration, hence not relevant for normal users, or for our project for 2 reasons:

1. Hidden categories do not provide relevant information in full article paths.
2. Hidden categories add complexity to our structure.

Hence, it is desirable to remove all links to hidden categories, which means that we need the titles of all hidden categories. On Wikipedia's information page about the category *Hidden Categories* [41] 15 385 subcategories are listed as immediate subcategories, which means that these are also hidden categories. Many of these categories have links to their own hidden subcategories. We

did two attempts at finding all hidden categories. The first attempt was to look through all the links from the category *Hidden Categories*, where 15 006 subcategories where found and marked as not relevant. This did not give the expected number, so the second attempt was made by looking at the file `enwiki-latest-page_props.sql.gz`. All ids marked with *hiddencat* (see figure 5.3) were collected and their corresponding category title found in `enwiki-latest-page.sql.gz`. This approach led to 15 513 categories. To make sure that all hidden categories where found, a test was made to see if all categories from the first attempt was found in the list created from the second attempt. The results showed that all categories found in the first attempt was also found in the second attempt. We created a list of 15 513 category names which should be disregarded from all paths in our results.

```
(747593,'hiddencat','',NULL)
```

Figure 5.3: Excerpt from the file `enwiki-latest-page_props.sql.gz` *where we can see that hidden categories are marked with* hiddencat

The hidden categories should be removed from both links between categories and articles, and between categories. The first case is easy since the links can just be removed from the results. Removing hidden categories between categories has to be done carefully to make sure that information is not lost. Hidden categories might be subcategories of visible categories or have visible categorise as their own subcategories. Figure 5.4 is an example of a path to the article about *Stevie Wonder* which includes a hidden category.

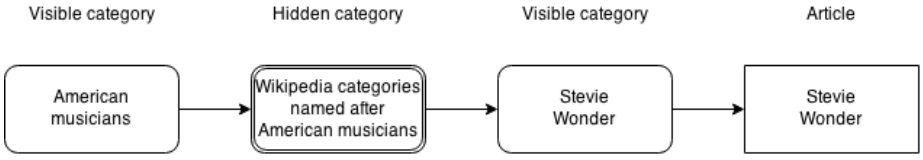


Figure 5.4: An excerpt of one path leading to the article about to Stevie Wonder, where the path contains a hidden category.

The desirable visible paths for all articles are paths without hidden categories. Thus, the hidden categories should be removed from the structure without loosing any of the subcategories which might contain relevant information or important links. Example of a how a path can be transformed is figure 5.5 which is the excerpt from the path in figure 5.4 without the hidden categories.

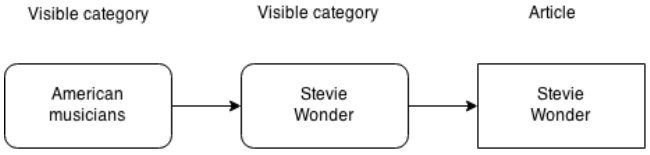


Figure 5.5: The desirable output of the excerpt of the path leading to the article about Stevie Wonder where the hidden category is removed from the path

Table 5.3 shows how number of links between categories and articles are reduced when hidden categories are disregarded. Number of links between categories has increased even though total number of categories are reduced from 519 822 to 504 309.

Links between...	**W/ Hidden Categories**	**W/o Hidden Categories**
subcategories	3 358 007	3 467 360
articles and categories	71 487 647	52 611 629

Table 5.3: Number of links removed when all hidden categories are excluded.

5.1.2 Representing the Underlying Structure

It is important that the category names are identical at all places they occur. Wikipedia is written by volunteers from all over the worlds, and users might use different encoding depending on where they are from. Thus, both a cleaning process and a normalization process should be performed on all category names. The cleaning process is to make the category names look readable, while the normalization is a process where all words are made equal regardless of case sensitivity and character encoding [18, p. 26].

Figure 5.6 is an example of an `INSERT` statement which represents a link between the category *fictional_ birds* and the subcategory *ducks\n fictional ducks*. This statement is an example of two category names that need to be processed so that they appear as *fictional birds* and *fictional ducks*. This processing is usually called a *data cleaning process* [6]. The data cleaning for our purpose is converting all words to lowercase, replacing underscores with spaces and splitting up titles containing the code for newline (*\n*). Wikipedia uses the code for newline to represent how the articles should be sorted. Figure 5.7 shows that *fictional ducks* are sorted as if it started with the word *ducks*.

```
(1517681,'fictional_birds','ducks\nfictional ducks
   ','2014-10-26 03:30:11',
'ducks','uppercase','subcat')
```

Figure 5.6: Excerpt from `enwiki-latest-categorylinks.sql.gz` showing an `INSERT` statement including a newline character.

After processing all titles, they are sorted into two files:

1. file containing links between categories
2. file containing links between categories and articles

These files are needed for creating the structures for finding full paths of all Wikipedia articles.

It is desirable to reduce the complexity in the files by removing articles whose titles are not relevant for our project. Numbers without context is an example of Wikipedia article titles that are irrelevant. The meaning of a number

Figure 5.7: The subcategories of the category Fictional birds *and how its subcategories are sorted based on the defined sortkey instead of the category title*

could have various meanings, including temperatures, grades or years. Hence, all article titles which only contains numbers could be disregarded. Wikipedia contains many such articles, and a total of 23 227 articles were found. This reduces the number of links between articles and categories (see table 5.4).

W/ Number Articles	W/o Number Articles
52 611 629	52 588 894

Table 5.4: Number of links between categories and articles removed when articles only containing numbers are disregarded

5.1.3 Following Links Between Categories

Finding the full paths for each Wikipedia article can be done when the representation of the structure is ready. Each path can be found by following the links between categories until an article is reached, and the category links visited form an article path.

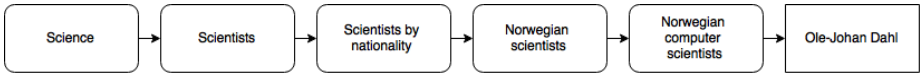


Figure 5.8: Example of one of the article paths of the article Ole-Johan Dahl. *The rectangles are categories and the rectangle with rounded corners is the article.*

Issues with finding the full path

The structure of Wikipedia is not represented as a tree, but as a graph. This means that there might be loops within the graph. A loop within the graph means that a category already visited in the search of a path can be reached again. Figure 5.9 shows an example of a path which contains a loop.

```
people/fictional characters/fictional characters by
    species/fictional life forms/legendary creatures in
     popular culture/fictional characters by species
```

Figure 5.9: Example of a loop found in a path.

This could lead to a problem if the program keeps going in loop, and does not reach an article. A solution to this problem is to keep track on categories already visited and only follow links to categories not yet visited in the path.

Another issue is to decide the start point for the paths, in other words decide the start category. Wikipedia contains some natural categories that are better to use as start categories. These categories are very general and have links to some of the major categories within different fields, thus, are able to reach most other categories in the Wikipedia category structure. The category *Main Topic Classifiers* was chosen for this task, because it has 22 subcategories within various fields and where all of them have their own subcategories (see figure 5.10)[56].

5.1.4 Redirects

Redirecting is a common technique for making a web page available to multiple URLs [50]. Wikipedia contains lots of redirects to articles, where it for instance redirects from an alternative article name, and redirects to the actual article. We are only interested in the article name it redirects to. Wikipedia has two main reasons for redirects. The first reason is to help users find the articles they are looking for regardless of misspellings or inflections of words. The second reason is to keep the encyclopedia well-structured. Wikipedia has divided the different types of redirects into 3 types:

1. Maintenance
2. Visual
3. Discussion

The reasons for redirecting is listed in figure 5.11 [55].

The redirects are divided into different types depending on the reason for redirecting. Wikipedia lists all the different reasons of redirects.

Handling redirects

It is desirable to use the article names that are redirected to in this project since they are spelled correctly. Thus, it is necessary to find all pages that are redirect pages and the pages they redirect to. If a page is supposed to be redirected to

Figure 5.10: Illustrates the first subcategories of the chosen start category Main Topic Classifiers. C *corresponds to the number of a category's subcategories and* P *corresponds to its number of pages. The figure is provided by Wikipedia's Category Tree.*

another page, this is found in `enwiki-latest-page.sql.gz` where the page's `INSERT` statement is marked with '1' in the 6th position (see figure 5.12).

The first attempt of handling redirects is to make sure the paths are found to articles with correct names and not to the redirecting pages. Finding paths to pages that redirect to other pages is unnecessary and creates more data than needed since these should be disregarded. A better way is to find all pages that other pages redirect *to*. This can be found in a separate file `enwiki-latest-redirect.sql.gz` where both page id and page title for all pages are found. After all of these ids and titles are collected, the next step is to connect them with the correct output. As an example the article from figure 5.12 would be connected to the page title in figure 5.13 after the title is converted to lowercase and underscores are removed.

- Alternative names
- Plurals
- Closely related words
- Adjectives/Adverbs point to noun forms
- Less specific forms of names, for which the article subject is still the primary topic.
- More specific forms of names
- Abbreviations and initialisms
- Alternative spellings or punctuation
- Punctuation issues—titles containing dashes should have redirects using hyphens.
- Representations using ASCII characters, that is, common transliterations
- Likely misspellings
- Likely alternative capitalizations
- To comply with the maintenance of nontrivial edit history
- Sub-topics or other topics which are described or listed within a wider article
- Redirects to disambiguation pages which do not contain "(disambiguation)" in the title
- Shortcuts
- Old-style CamelCase links
- Links auto-generated from Exif information
- Finding what links to a section, when links are made to the redirect rather than the section.

Figure 5.11: Wikipedia's reasons for redirecting a Wikipedia article.

```
(10,0,'AccessibleComputing
   ','',0,1,0,0.33167112649574004,
'20150111235554','20150112004211',631144794,69,NULL)
```

Figure 5.12: Example of a redirecting `INSERT` *statement in* `enwiki-latest-page.sql.gz`.

```
(10,0,'Computer_accessibility','','')
```

Figure 5.13: The page title AccessibleComputing *(figure 5.12) redirect to* Computer Accessibility.

5.2 Id Mapping

The files containing the results become extremely large due to the extreme amount of data. All paths to all Wikipedia articles are more than 20 GB of compressed data. It is desirable to reduce the space needed for storing all results on the computer. The solution was to create an id mapping for each category name and article name. Id mapping gives all names a unique id, and instead of writing the full path of category names to the file, we write the full paths with category ids.

The id mapping is implemented by creating a counter that assigns numbers to each category name or article name that is not found yet, i.e., a unique number represents each name. Figure 5.14 shows an excerpt of the id mapping created for our purpose, where the id *4600570* corresponds to the article about *Ole-Johan Dahl*, which means that this id is used everywhere *Ole-Johan Dahl* is used in paths.

```
...
4600566 roger matthews
4600567 pesticide drift
4600568 roxy theatre (clarksville, tennessee)
4600569 papadindar
4600570 ole-johan dahl
4600571 red square (university of washington)
...
```

Figure 5.14: Excerpt of the id mapping between id and the name of all categories and articles.

Id mapping is storage efficient because category names and article titles are usually a lot longer than their representing ids. This means that the disk space needed for storing the id is smaller than the disk space needed for storing the category names or the article titles.

Working with ids is also faster in many implementations concerning lookups in the program. This depends on the chosen data structure, but dictionaries which are frequently used in our implementation is an example of a data structure that performs faster with id mapping. An example of this is illustrated in figure 5.15 where the time to find all categories from the category with id *177678* (corresponding to the category *people*) is 0.955 minutes. Figure 5.16 shows the time (1.559 minutes) needed to find the same paths for the category names and for the article titles. Comparing the times shows that using ids are much faster, which is important when many paths have to found.

```
[INFO] Finding all article paths from 177678

[INFO] Time to find all article paths: 0.955 min
```

Figure 5.15: Time needed for finding all paths from the category 177678 *(corresponding to the category* people*) when ids are used by our program.*

```
[INFO] Finding all article paths from people

[INFO] Time to find all article paths: 1.559 min
```

Figure 5.16: Time needed for finding all paths from the category people *when our program uses full names of categories and articles.*

The last reason to use ids instead of full names is that the full names may include characters useful for describing paths, for instance the characters "/" which is a common way of describing full paths. This could lead to trouble if the category name contains the symbol.

5.3 Grading of Categories

Most of the articles can be reached from categories that are not descriptive of the content at all. The article about *Ole-Johan Dahl* (the Norwegian programmer) can be reached from the category *people*, but also found from the categories *politics* and *arts* (see Figure 5.17). This means that not all paths are good for describing the content of the Wikipedia articles. Thus, the next step is to grade the paths, to find the paths most likely to describe the content.

```
ole-johan dahl:
*people/people categories by parameter/categories by
   nationality/academics by nationality/norwegian
   academics/faculty by university or college in
   norway/university of oslo faculty

[...]

*politics/political activism/leadership/management/
   quality/software quality/formal methods/formal
   methods people

[...]

*arts/aesthetics/design/software design/data modeling/
   formal methods/formal methods people
```

Figure 5.17: Some of the full paths found for the article about Ole-Johan Dahl.

5.3.1 Grading based on Inlink and Outlink Numbers

Our first assumption is that categories with a high inlink number (number of categories linking to a category) can be reached from categories with unrelated topics. An example of a category with a high inlink number is *World War II*. This category can be reached from 87 different categories (see figure 5.18).

Figure 5.18: All categories linking to the category World War II. *This is an example of a category with high inlink number.*

The next assumption is that categories with a high outlink number (number of categories reached from a category) are more likely to reach categories not relevant since they can reach far in all the subcategories' directions. Figure 5.19 illustrates number of subcategories found for the category with highest outlink number, which is the category *Albums by artist* with a outlink number of 17 393.

These assumption combined are the foundation of grading based on inlink and outlink numbers. Categories frequently reached should obtain a lower score than categories rarely reached. We need some way of deciding whether an inlink number is high for each category. This can be done by comparing the inlink number with the average inlink number, and similarly comparing the outlink number and the average outlink number.

The average inlink and outlink numbers are found by summarizing all inlink numbers and outlink number respectively, and dividing the result on number of categories. Table 5.5 shows the average inlink and outlink numbers found for all categories in the underlying category structure.

Average *inlink number*	5
Average *outlink number*	2

Table 5.5: Average inlink number and outlink number for all categories.

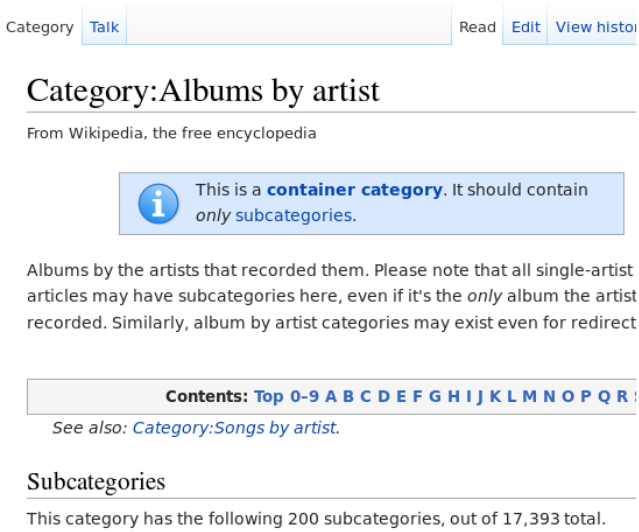


Figure 5.19: The category Albums by artist *is an example of category with high outlink number.*

The score for each category was found with equation 4.1, where each category's inlink number and outlink number is compared by the average inlink and outlink number.

Evaluation of the scores

None of the categories can have a score of 0 since all Wikipedia categories are connected to at least one other category [8]. The lowest score found was 0.376010, which was given to all categories with only one parent category and with no subcategories. This was a total of 104 471 categories. The category with the highest score is the category *Albums by Artist*, which is the category with most subcategories (17 393), hence a score of 6512.120784. Thus, the range of the scores is *<0.376010, 6 512.120784>*.

Figure 5.20 and figure 5.21 show the number of categories found for each of the possible score values. Figure 5.20) illustrates the results for all possible score values, while figure 5.21) illustrates the 25 smallest score values and their corresponding number of categories. These figures shows that there are many categories with low score values, while there are only a few categories with higher score values. The categories with high score value will have a high impact on the article path, which means that paths containing these categories will have a lower probability of being considered relevant.

[8] We mentioned in challenges (Introduction, encoding) that some of our connections were broken. This does not affect the scoring of the categories, since the inlink and outlink numbers are preserved for all categories.

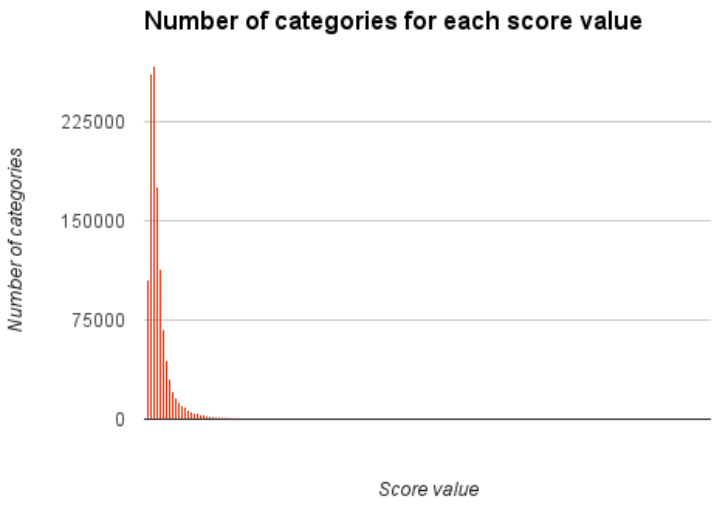


Figure 5.20: Number of categories for each possible score value. The X axis varies from 0.376010 where most categories are placed (left) to 6515.120784 for Albus by Artist *(right).*

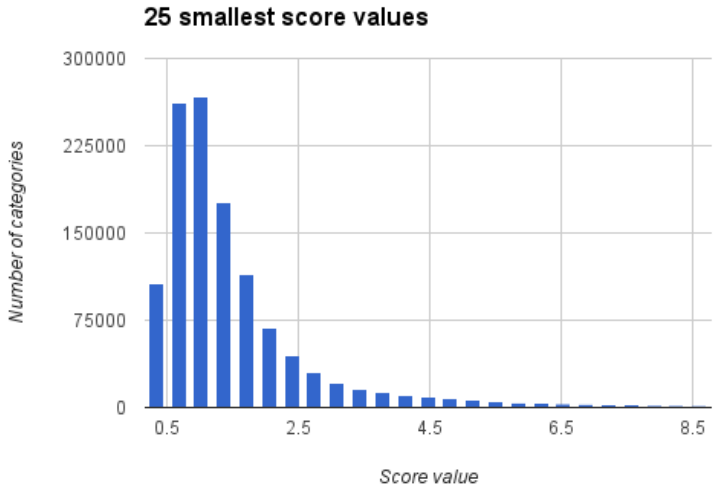


Figure 5.21: The 25 smallest score values and their corresponding number of categories. These are the most common score values since they are connected to categories with low inlink and outlink numbers.

Problems with the simplified grader

Since it is desirable to have the lowest score as possible, the first problem encountered was that the program favoured short paths. Figure 5.22 gives an

example of how the shortest path for the article *Alexander Hughes* (English football player [38]) is favoured. These paths are not very descriptive of the article, where only the third best path gives information that he is connected to football. Instead, we would like to see if longer paths are better.

```
alexander hughes:
*people/people categories by parameter/people by time/
   births by year/year of birth missing (28.200766)
*nature/life/births by year/year of birth missing
   (28.576777)
*health/health by city/health in edinburgh/sport in
   edinburgh/sports teams in edinburgh/football clubs
   in edinburgh/heart of midlothian f.c./heart of
   midlothian f.c. players (37.22501)
```

Figure 5.22: Example of how the grading based in inlink and outlink numbers favours short paths.

5.3.2 Normalized Grading Based on Inlinks and Outlinks

A way of avoiding favoritism of short paths is by normalizing the path scores. Equation 4.3 was used to normalize the path scores for each path so that the length of the path does not determine the relevance of the path.

Figure 5.23 shows the three best results for the same article (Alexander Hughes) when the paths are normalized. The results here are more descriptive of the content of the article, where all paths contains information that he is associated with football.

```
alexander hughes:
* health/health by city/health in edinburgh/sport in
   edinburgh/sports teams in edinburgh/football clubs
   in edinburgh/heart of midlothian f.c./heart of
   midlothian f.c. players/ (4.431941375)
* concepts/principles/rules/sports rules and
   regulations/sports terminology/association football
    terminology/association football positions/
   association football players by position/
   association football defenders/ (5.01043655556)
* sports/sports terminology/association football
   terminology/association football positions/
   association football players by position/
   association football defenders/ (6.08136966667)
```

Figure 5.23: The three best paths for Alexander Hughes *when the path scores are normalized.*

5.4 Mapping to Desirable Output Categories

Our goal for the mapping process is to create a link between Wikipedia article titles and one or more categories from the desirable output categories. It is essential to know the meaning of the Wikipedia articles in order to create such a mapping. Our theory is that this information can be found in the full paths of the articles, where a full path of a Wikipedia article contains the categories visited to reach the article. This means that the machine needs some predefined knowledge to identify the meaning of the paths. Two approaches were tried for this task; creating a mapping between Wikipedia categories and output categories, and creating mapping between path excerpts and output categories.

5.4.1 Mapping based on Wikipedia Categories

The first approach was to create a mapping between each Wikipedia category and one or more categories in the desirable output category set. The idea is that a matching could be performed by matching words within the Wikipedia category names and an output category name (see figure 5.24).

```
Wikipedia category: tennis
IAB category: sports/tennis
```

Figure 5.24: Desirable mapping between a Wikipedia category and an IAB category. sports/tennis *in the IAB category means that* tennis *is placed under the tier* sports.

The task of mapping each Wikiepedia category to desirable output categories is too big to be done manually since the Wikipedia category set contains 1 201 373 categories. This means that the process should be automated. One way of doing this is by looking at similarities in the words contained in the Wikipedia category and in the output category.

Expanding the IAB category

The categories in the IAB taxonomy were chosen as the desirable output category set for our task. This taxonomy only consists of two category *layers* (also called *tiers*), which are not specified enough for creating a matching based on the category names. Hence, the IAB taxonomy was extended with a third and more specified layer to improve the category mapping process. This layer is added by creating a link from the second layer and to its representative third layer.

This third layer can be viewed as giving common knowledge to the machine. *Europe* is an example of a second layer where the machine lacks common knowledge since it does not know what countries are part of Europe. Expansion of this tier could be creating a third tier containing all European countries, which means that all Wikipedia categories containing a name of a European country should map to the category *Europe*.

Lemmatization

Figure 5.25 shows how a matching between Wikipedia categories and output categories, where the output category name *sports* are found as a word in the Wikipedia category name *ministry of youth affairs and sports*.

```
Wikipedia category: ministry of youth affairs and
   sports
IAB category: sports
```

Figure 5.25: Exact match on mapping between Wikipedia category and output category, where the output category is found in the Wikipedia category.

The problem with this approach is that words like *sport* will not be an exact match of the word *sports*. This means that this Wikipedia category will not be included under the desirable output category. Thus, the next step was to find matches between the categories regardless of the declension of the word. This part is called *lemmatization* and is defined as the process where different inflected forms of a word are grouped together [18, p. 30-33]. There are various lists for lemmatization available online. We chose a list from *MBM's Lemmatization Lists* [17] which provided a list of common lemmatization. Both the words in the Wikipedia categories and the desirable output categories were processed so they where replaced by their lemma, i.e. the canonical form of the word [18, p. 30]. Figure 5.26 shows example of a match found after lemmatization is performed.

```
Wikipedia category: sailors at the 1956 summer
   olympics
IAB category: olympics
IAB category: sailing
```

Figure 5.26: Example of a match between Wikipedia category and output category after lemmatization, where sailors *match with* sailing

5.4.2 Mapping based on Wikipedia Path Excerpts

The other attempt was built on the idea that a the mapping from Wikipedia category and output categories needs more information about the Wikipedia categories. The idea is that this information could be found in excerpts of articles' full paths. Thus, the mapping process is based on excerpts of the paths, which should be mapped to one or more output categories. This approach solves the problem with ambiguous category names, because we specify the meaning of the category name in the path excerpt (see figure 5.27)

5.4.3 Automatic Mapping

We started out by manually creating mappings between path excerpts and IAB categories, but this is a large task since there exist so many categories and

```
ancient philosophers/cicero
towns in illinois/cicero,illinois
```

Figure 5.27: How disambiguation can be solved if parts of the full path is used to determine the meaning of the category name.

category links in Wikipedia's structure. Thus, it is desirable to automate the mapping process.

We tried to find a good way to predict matches between the excerpts and the output categories. We assumed that the IAB subcategory name (e.g., *Auto parts*) is a category, and wanted to find the most likely categories leading to this category. This was done by finding all categories leading to this category among the top 3 category paths for each Wikipedia title, and counting the occurrences. All patch excerpts among the 10 most common were chosen if they seemed logical.

5.4.4 Processing Titles

A match in a random article is found if a phrase or word is an exact match with a Wikipedia article title, hence the Wikipedia article titles can be viewed as entries in a dictionary. The titles should therefore be processed to make sure that matches will be found.

Disambiguation or specification of titles

Several Wikipedia titles contains parenthesis that specify what the Wikipedia article is about. Figure 5.28 shows two Wikipedia titles for the *David Sharpe*, where one article is about David Sharpe (1967-) the British athlete [46] and the other is about David Sharpe (1910-1980) the American actor [45].

Ambiguous titles are a problem if they are categorized to different categories. Many ambiguous articles are marked with *(Disambiguation)* (see also 1.3). However, we experienced that not all ambiguous articles are marked like this. Our next approach was to find all category links from the category *All disambiguation pages* and collect all these page titles.

```
david sharpe (athlete)
david sharpe (actor)
```

Figure 5.28: Wikipedia article titles with parenthesis.

Many Wikipedia articles are about events happening a specific year. Exact matching with these titles will most likely occur, hence the year should be removed from the entry. Figure 5.29 shows an example of two entries which corresponds to the Davis tennis tournaments in 1996 and in 2000. Removing the year from these entries will increase the probability of finding a match, but also make both entries look the same.

Another specification found in Wikipedia articles is specification on gender, like *2015 Dubai Tennis Championships – Women's Singles* a figure 5.30 shows.

```
Wikipedia article title: 1996 Davis Cup
Wikipedia article title: 2000 Davis Cup
```

Figure 5.29: Wikipedia article titles which will look the same when removing the year from the title.

This specification reduces the probability of an exact match, hence *women's* and *men's* are removed from the title and reduces it to a more general form which is more likely to occur.

```
2015 Dubai Tennis Championships
2015 Dubai Tennis Championships - Women's Singles
2015 Dubai Tennis Championships - Men's Singles
```

Figure 5.30: Wikipedia articles specified for gender (women and men) and gender neutral.

The next step is to decide whether the modified entries mean the same or have different meaning after the parenthesis and years are removed. This was done by looking at the mapping of the entries. Two processed entries are considered identical if they are a match of each other and are mapped to the same category. One of the entries is kept if the entries are identical, both are disregarded otherwise. The entry *David Sharpe* (figure 5.28) is an example of an entry that is removed from the dictionary since the two original entries are mapped to different categories , while *Davis Cup* (figure 5.29) is kept since both of the entries are mapped to the same categories. The gender specific entries in figure 5.30 are reduced to one entry *Dubai Tennis Championships - Singles* when gender and year is removed from the entry, and is kept in the dictionary.

There are both advantages and disadvantages with this approach. The main disadvantage is that entries are removed, hence, information is lost. We could still argue that the removed entries are the entries most likely to wrongly classify text, and that the probability to correctly classify text is increased when these entries are removed.

Removing common words

Some of the entries are reduced to very common English words. Figure 5.31 shows that Wikipedia article title *(85476) 1997 MY* (a main-belt minor planet [37]) are reduced to the entry *my* (determiner: belonging of me). This means that the dictionary entry *my* is categorized to the same as *(85476) 1997 MY*, which is *Astronomy&Space*.

```
Wikipedia article title: (85476) 1997 MY
Entry: my
```

Figure 5.31: Example of an entry that has been reduced to a common English word.

Words that occur extremely often in most documents are more likely to disturb the categorization instead of providing useful information. These words should henceforth be disregarded as entries. This was done by creating a large list containing the most common English words, called a *stop list* [18, p. 25]. An entry is removed if it is reduced to one of these words. The stop list chosen for this implementation was chosen as the 1000 most basic English words according to Wictionary, combined with the 100 most common spoken words according to TV and movie scripts [58].

5.5 Dictionary-based Classifier for Other Languages

The multilingual side of Wikipedia is one of its main advantages. Most Wikipedia articles are available in multiple languages. The available languages for an article are shown at the left side of the article. We wanted to take advantage of this property to create dictionary-based classifiers for other languages than just English.

Our assumption is that we can create dictionary-based classifiers based on the English classifier. This is based on the fact that there are twice as many articles in English than any other language [51]. The disadvantage with this approach is that some articles might not be available in English, but might be essential entries in a classifier for the language.

The file `enwiki-latest-langlinks.sql.gz` creates a database table which contains information about the available languages for each English Wikipedia article. All language links are represented as entries in `INSERT` statements on the form

$$(il\_from,\ il\_lang,\ il\_title).$$

Table 5.6 contains the description of the entry fields in each entry [21], and figure 5.32 is an example of entries translating English articles to French. The language code *'fr'* determines that the links are between the English Wikipedia articles and the corresponding French Wikipedia articles.

Entry field	**Description**
il_from	page_id of the referring page.
il_lang	Language code of the target, in the ISO 639-1 standard.
il_title	Title of the target, including namespace (FULLPAGENAMEE style).

Table 5.6: Description of the entries in the table Langlink.

5.5.1 Creating a Norwegian Dictionary-based Classifier

We chose to create Norwegian dictionary-based classifier to test the idea in real life. The main reason to choose Norwegian is that the results can be manually evaluated since our native tongue is Norwegian.

The first step was to find the Wikipedia article id for each entry in the English dictionary. Furthermore, we needed all links between English Wikipedia

```
INSERT INTO `langlinks` VALUES
(10642344,'fr','Muro de Aguas'),
(1666460,'fr','Muro de Alcoy'),
(32877065,'fr','Muro en Cameros')
```

Figure 5.32: Example of entries for linking the English ids to the corresponding French articles.

articles and the Wikipedia articles for the desired language, Norwegian in our case. Thus, we created a dictionary for the new language by using the translated titles of the Wikipedia articles and the categories found by the mapping process for the English dictionary. Finally, a cleaning process was performed on the entries in the new dictionary to ensure ambiguous entries were removed.

Article id for all dictionary entries

The dictionary for the English classifier consists of entries and their corresponding categories. It is essential to know the corresponding page ids for each dictionary entry for translating it into other languages. The page ids for a dictionary entry are found by following the same steps for transforming the Wikipedia article titles into dictionary entries while storing its ids at all times. The results of this process is independent of the language of the new classifier, thus reusable.

The file `enwiki-latest-categorylinks.sql.gz` is used in the process of creating the dictionary for the English classifier, and contains links between Wikipedia articles and categories. Both Wikipedia article titles and their ids can be found from this file. Thus, we created a list of all Wikipedia article titles and their id. Next, we processed the Wikipedia article titles similar to the process performed when creating dictionary entries (see section 5.1.2 and section 5.4.4). This makes the processed Wikipedia article titles identical to the dictionary entries.

The entry processing performed on the Wikipedia article titles could lead to multiple page ids so all page ids are stored for each dictionary entry.

Finding all Norwegian links

There is a language code for representing all available languages for Wikipedia. The English language code is *en* and the Norwegian language code is *no*. All English articles available in Norwegian is found by finding all entries with *'no'* in the entry field *il_lang*. The page id of the English Wikipedia article and its corresponding Norwegian title are stored together (see figure 5.33).

We were only interested in titles of Wikipedia articles for creating the Norwegian dictionary. Links between categories, talks or users were therefore not considered relevant. These were found by removing all pages with a name space not relevant, and were found from Wikitalk pages [47] [53] [54].

A total number of 281 617 Norwegian article titles were extracted from `enwiki-latest-langlinks.sql.gz`. The Norwegian Wikipedia lists a total

```
378466      Fidel V. Ramos
287145      Jacques Cazotte
24984364    Gnathothlibus
287149      Nyala
2653965     Papuacedrusslekten
370256      Timurid-dynastiet
33931676    Druide
778284      Flora (gudinne)
4369469     Aloandia
1246681     Edge (magasin)
5980        Karbonsluk
1902206     Nansenprisen
```

Figure 5.33: The page id of the English Wikipedia article and its corresponding Norwegian title. Notice that some of the Norwegian titles contains parenthesis for handling ambiguous words.

number of 410 286 articles[9] on the web page [49], which means that approximately 69% of all the Norwegian articles were found from the language table.

Translating the English classifier

The next step was to translate the English dictionary to Norwegian. This was done by looking at the each entry in the English dictionary and finding the corresponding article title in Norwegian. Figure 5.34 shows how the English dictionary entries correspond to a Norwegian title, and 5.35 shows the results when the English categories are added to the Norwegian titles.

```
bicycle kick        brassespark
davis phinney       davis phinney
phasi charoen       district  phasi charoen
chanthaburi         chanthaburi
hindnubben          hindnubben
kamrup district     kamrup (distrikt)
```

Figure 5.34: Example of English dictionary entries and their corresponding titles in the Norwegian Wikipedia.

Processing the Norwegian entries in the new dictionary

Finally, we needed to process the entries in the Norwegian dictionary. The reason for this is that some of the Norwegian entries might be ambiguous even though the English entries are not. Thus, we collected all entries which contained parenthesis and regarded this. It was also essential to remove all entries

[9] Total number of Norwegian articles per 27th of April 2015. This number might have been slightly different 25th of January 2015, which is the date we downloaded the database dump used to create the English dictionary-based classifier.

```
"speilreflekskamera":   ["technology & computing/
   cameras & camcorders"],
"henry hermansen":      ["sports/skiing"],
"joost wichman":        ["sports/mountain biking"],
"punt e mes":           ["food & drinks/wine"],
"mikroorganisme":       ["science/biology"],
```

Figure 5.35: The Norwegian dictionary when the English categories are given to their corresponding Norwegian entry

that were reduced to common Norwegian words. This was done by creating a stop word list, and we chose to base this list on a frequency list from *Google Code*[3]. This reduced number of entries in the Norwegian dictionary from 281 617 to 256 219.

5.5.2 Deploying the Results

The final part of our implementation was to deploy the results to Cxense, where the dictionary was added to the classifier for categorizing articles. Each of the dictionaries were added with settings for article matching (see figure 5.36).

```
{
  "global-properties": {
    "count": ["2"],
    "leftmost-longest-match": ["true"],
    "unique-count": ["2"],
    "annotate-paths": ["false"],
    "swap-fields": ["true"],
    "expand-paths": ["true"],
    "count-field": ["value"],
    "value-normalization-flags": "4",
    "mode": ["overlap"],
    "key-normalization-flags": ["4"],
    "tokenizer-context": ["en"]
  },
  "igg-iabtaxonomy6": {
    "personal effects": [
      "arts & entertainment/movies"
    ],
    ...
```

Figure 5.36: The settings for the classifier at Cxense.

Chapter 6

Results and Discussion

This chapter is dedicated to the results of our project and the evaluation of these results. The chapter starts by evaluating the categorization of the keywords where we tried two approaches; mapping between Wikipedia categories and output categories, and mapping between path excerpts and output categories. Furthermore, we describe the different versions of the classifier in section 6.2. Section 6.3 is dedicated to the evaluation of the performance of the dictionary-based classifier, including how to retrieve the results and a comparison with results from another dictionary-based classifier. We evaluate the Norwegian dictionary-based classifier in section 6.4 to determine how well the classifier performs, and end the chapter by discussing our results.

6.1 Evaluation of Category Mapping

We implemented two different approaches for mapping the Wikipedia article titles to the IAB taxonomy. The first approach was to create a mapping between Wikipedia categories and a describing category in the IAB taxonomy based on matching of words. The other approach was to create a mapping between excerpts of category paths and their most describing IAB category.

Mapping from Wikipedia Categories to Output Categories

Our first attempt was to create a mapping between Wikipedia categories and the IAB taxonomy. This was found to be very difficult. The idea of the mapping was to match words in the categories with words in the IAB taxonomy, first by only matching identical words (e.g., sport and sport) and later by matching words with similar meaning (e.g., sport and sports). A perfect result for such an approach is only achievable if the computer is able to understand natural language, i.e., know all synonyms, inflections and the true meaning of all words. There exists many projects for natural language processing, including WordNet, which shows that this task is a lot larger than the scope of our project.

Another problem encountered with the mapping was ambiguous words in the category names. Figure 6.1 shows two categories that both contain the word *Cicero*, but where the first category is for the suburb of Illinois [43] and the other is for the Roman philosopher [42]. Creating mapping rules for these names would be a difficult or even impossible task.

```
Category:Cicero, Illinois
Category:Cicero
```

Figure 6.1: Example of two category names which contains the same word with different meaning, and should be classified to different categories.

```
Towns in Illinois/cicero,illinois
Ancient philosophers/cicero
```

Figure 6.2: Example of solving disambiguation by using path excerpts instead of category names.

The conclusion for this approach is that a mapping from Wikipedia categories to desirable output categories would consist of many specified mapping rules. The task of creating such a mapping would therefore resemble a manual classification and is not a desirable approach.

6.1.1 Mapping from Path Excerpts to Output Categories

The second approach was based on mapping from excerpts of Wikipedia category paths. This approach has the advantage that we can avoid ambiguity even though the category name or parts of the category name are ambiguous. Figure 6.2 illustrates the disambiguation, where the two path excerpts are not ambiguous even though they contain an ambiguous word. The idea of this approach is to create mapping rules just like in the first attempt, but the main difference here is that we don't need to know word inflections, synonyms or the true meaning of words. Instead, we assume that the meaning of the words are found from the path excerpts. However, we need to determine which path excerpts to use in the mapping process, and it is desirable to make this process as automatic as possible.

Automatic mapping from path excerpts to output categories

We started out by manually finding path excerpts for the mapping process, but this task is large since there are so many categories, category links and possible category paths. This was our motivation for creating an automated way of finding path excerpts useful for the mapping process. We assumed that the category names in the IAB taxonomy's second tier (e.g., *Auto parts*) are existing categories, and wanted to find the most likely categories leading to these. All Wikipedia article titles were stored with their 3 best category paths (these were found by grading the path) and these were used to determine the most likely categories leading to the IAB categories. This was done by counting occurrences and keeping the top 10 categories leading to the IAB category.

We needed to evaluate the results of the program in order to see if they were useful in the mapping process. It was essential to have some correct results in order to evaluate the automatic mapping. Thus, we decided to manually create a mapping from all subcategories for *Automotive* and then compare these results with the path excerpts recommended from the automatic mapping. Figure 6.4

shows the results of the manual choice of path excerpts for the mapping process of the subcategories of *Automotive*. Figure 6.5, figure 6.6 and figure 6.7 represent the results for the top 10 category path excerpts which were automatically found for each of the subcategories.

These results show that the automatic approach finds more path excerpts than the manual approach. However, the automatic approach lacks common knowledge which means that it also finds path excerpts that are incorrect. Figure 6.3 illustrates an incorrect path excerpt which is found automatically by the program. This path excerpt is found since *coupe* is ambiguous, but most humans would immediately understand that the word has different meanings.

Hence, we decided that the best results would be found by combining the two approaches:

1. Automatically find the top 10 path excerpts leading to the subcategories.
2. Review the results and choose the path excerpts that seem correct.

```
Coupe:
2001-02 in french football/2001-02 coupe de france
```

Figure 6.3: Example of automatic categorization that does not work.

The final results of *Automotive*'s subcategories were found when we looked through all the path excerpts found from the automatic mapping and removed those that did not make sense. The results are shown in figure 6.8 and figure 6.9. We can see that the final results contain more path excerpts than the manual approach and less path excerpts than the automatic approach. This is also illustrated in figure 6.10, where we can see statistics for each of *Automotive*'s subcategories, i.e., the number of path excerpts found manually, the number of path excerpts found automatically and the number of path excerpts kept in the final results.

The function of the path excerpts

The main function of the path excerpts is to connect the keywords to IAB categories. The path excerpts can be viewed as a third tier in the IAB taxonomy, where the path excerpts are placed under their descriptive category name in the second tier. Another point of view is to think of the path excerpts as common knowledge where we tell the classifier what patterns it should look for within the category paths when categorizing the keywords to IAB categories.

Auto Parts	components/vehicle parts/auto parts
Auto Repair	automobile maintenance/automotive repair shops
Buying & Selling Cars	
Car Culture	transport culture/car culture
Certified Pre-Owned	
Convertible	car body styles/convertibles
Coupe	car body styles/coupes
Crossover	car classifications/crossover sport utility vehicles
Diesel	
Electric Vehicle	electric vehicles/hybrid electric vehicles/hybrid electric cars
	green automobiles/hybrid cars/hybrid electric cars
	green automobiles/electric cars
	electric vehicles/electric cars
	car classifications/electric sports cars
Hatchback	car body styles/hatchbacks
Hybrid	electric vehicles/hybrid electric vehicles/hybrid electric cars
	green automobiles/hybrid cars/hybrid electric cars
	green vehicles/partial zero-emissions vehicles/hybrid vehicles
Luxury	luxury vehicles
MiniVan	car classifications/minivans
Motorcycle	transport culture/motorcycling
	vehicle stubs/motorcycle stubs
	motorcycling/motorcycle sport
Off-Road Vehicles	off-roading/off-road vehicles
	dirt biking/off-road racing
Pickup	car classifications/pickup trucks
Road-Side Assistance	emergency services/emergency road services/
Sedan	car classifications/sports sedans
	automobiles/car body styles/sedans
Trucks & Accessories	wheeled vehicles/trucks
	car classifications/pickup trucks
	transport culture/trucking subculture
	electric vehicles/electric trucks
	sport utility vehicles/sport utility trucks
	hybrid vehicles/hybrid trucks
Vintage Cars	automobiles by period/vintage vehicles
Wagon	automobiles/car body styles/station wagons

Figure 6.4: Manually finding path excerpts for the mapping process to subcategories of Automotive.

Auto Parts	vehicle parts/auto parts
	automotive companies/auto parts suppliers
	automotive technologies/auto parts
Auto Repair	
Buying & Selling Cars	
Car Culture	transport culture/car culture
	automobiles/car culture
Certified Pre-Owned	
Convertible	car body styles/convertibles
	automotive accessories/convertible top suppliers
	classes of mobile computers/convertible laptops
	tablet computers/convertible laptops
	laptops/convertible laptops
Coupe	car body styles/coupes
	association football matches/coupe de france finals
	seasons in french football competitions/coupe de france seasons
	association football matches/coupe de la ligue finals
	figure skating competitions/coupe internationale de nice
	international ice hockey competitions hosted by france/coupe de chamonix
	football cup competitions in france/coupe de la ligue
	2009–10 in french football/2009–10 coupe de france
	dutch books by writer/books by louis couperus
	national association football cups/coupe de france
	international association football competitions hosted by france/coupe de l\'outre-mer
	2001–02 in french football/2001–02 coupe de france
	2000–01 in french football/2000–01 coupe de france
	2007–08 in french football/2007–08 coupe de france
	musical families/couperin family
Crossover	postmodern works/crossover video games
	continuity (fiction)/fictional crossovers
	metafictional works/fictional crossovers
	hardcore punk groups/crossover thrash groups
	continuity (fiction)/intercompany crossovers
	car classifications/crossover sport utility vehicles
	tokusatsu/crossover tokusatsu
	popular music/crossover (music)
	hardcore punk albums/crossover thrash albums
	jazz albums by genre/crossover jazz albums
	classical albums by genre/classical crossover albums
	fusion music genres/crossover (music)
	pop albums by genre/classical crossover albums

Figure 6.5: Automatically finding path excerpts for the mapping process to subcategories of Automotive *(part 1).*

Diesel	locomotive stubs/diesel locomotive stubs
	petroleum products/diesel
	caterpillar inc. subsidiaries/electro-motive diesel locomotives
	british rail rolling stock/british rail diesel multiple units
	rail transport preservation/preserved diesel locomotives
	engines by model/diesel engines by model
	sustainable transport/biodiesel
	british rail locomotives/british rail diesel locomotives
	metre gauge locomotives/metre gauge diesel locomotives
	oil companies/biodiesel producers
	locomotives of sri lanka/diesel locomotives of sri lanka
	multiple units of portugal/diesel multiple units of portugal
	locomotives of tasmania/diesel locomotives of tasmania
	energy crops/biodiesel feedstock sources
	internal combustion piston engines/diesel engine technology
Electric Vehicle	low-carbon economy/electric vehicles
	electric power/electric vehicles
	sustainability organisations/electric vehicle organizations
	battery manufacturers/electric vehicle battery manufacturers
	battery chargers/electric vehicle infrastructure developers
	vehicles by fuel/diesel-electric vehicles
	sustainability advocates/hybrid electric vehicle advocates
Hatchback	car body styles/hatchbacks
Hybrid	fictional life forms/fictional hybrid life forms
	mythic humanoids/mythological human hybrids
	bus operating companies/transit authorities with hybrid buses
	organisms/hybrid organisms
	buddleja/buddleja hybrids and cultivars
	green buses/hybrid electric buses
	human-derived fictional species/fictional human hybrids
	electric vehicles/hybrid electric vehicles
	partial zero-emissions vehicles/hybrid vehicles
	martial arts by type/hybrid martial arts
	controversial bird taxa/hybrid birds of paradise
	green automobiles/hybrid cars
	grape varieties/hybrid grape varieties
	legendary creatures/mythological hybrids
	hominini/hominid hybrids
Luxury	brands/luxury brands
	car classifications/luxury vehicles
	culture/luxury
	real estate/luxury real estate
	wealth/luxury
	sport utility vehicles/luxury sport utility vehicles
	hotel types/luxury hotel

Figure 6.6: Automatically finding path excerpts for the mapping process to subcategories of Automotive *(part 2).*

Motorcycle	personal transporters/motorcycles
	wheeled vehicles/motorcycles
	motorcycle classifications/standard motorcycles
	motorcycling/motorcycles
	lists of vehicles/lists of motorcycles
	motorcycle classifications/cruiser motorcycles
	motorcycle classifications/dual-sport motorcycles
	20th century in transport/20th-century motorcycles
	grand prix motorcycle racing/grand prix motorcycles
	motorcycle classifications/sport touring motorcycles
	wheeled military vehicles/military motorcycles
	racing vehicles/racing motorcycles
	1940s in transport/motorcycles introduced in the 1940s
	electric two-wheel vehicles/electric motorcycles
	motorcycle classifications/touring motorcycles
Off-Road Vehicles	off-road vehicles:
	off-roading/off-road vehicles
	land vehicles/off-road vehicles
Pickup	car classifications/pickup trucks
	audio transducers/guitar pickups
	auto racing series/pickup truck racing series
	seduction community/pickup artists
	electrical generators/guitar pickups
Road-Side Assistance	
Sedan	car body styles/sedans
	auto racing series in australia/australian sports sedan championship
	car classifications/sports sedans
Trucks & Accessories	
Vintage Cars	
Wagon	car body styles/station wagons
	history of road transport/wagons
	rail freight transport in the united kingdom/british railway wagons
	rolling stock of victoria (australia)/victorian railways goods wagons
	steam road vehicle manufacturers/steam wagon manufacturers
	manufacturing plants in england/birmingham railway carriage and wagon company
	freight rolling stock/british railway wagons
	skate punk albums/lagwagon albums
	protected areas of oklahoma/protected areas of wagoner county, oklahoma

Figure 6.7: Automatically finding path excerpts for the mapping process to sub-categories of Automotive *(part 3).*

Auto Parts	components/vehicle parts/auto parts
	automotive companies/auto parts suppliers
	automotive technologies/auto parts
Auto Repair	automobile maintenance/automotive repair shops
Buying & Selling Cars	
Car Culture	transport culture/car culture
	automobiles/car culture
Certified Pre-Owned	
Convertible	car body styles/convertibles
	automotive accessories/convertible top suppliers
Coupe	car body styles/coupes
Crossover	car classifications/crossover sport utility vehicles
Diesel	locomotive stubs/diesel locomotive stubs
	petroleum products/diesel
	caterpillar inc. subsidiaries/electro-motive diesel locomotives
Electric Vehicle	low-carbon economy/electric vehicles
	electric power/electric vehicles
	electric vehicles/hybrid electric vehicles/hybrid electric cars
	green automobiles/hybrid cars/hybrid electric cars
	green automobiles/electric cars
	electric vehicles/electric cars
	car classifications/electric sports cars
	sustainability organisations/electric vehicle organizations
	battery manufacturers/electric vehicle battery manufacturers
Hatchback	car body styles/hatchbacks
Hybrid	electric vehicles/hybrid electric vehicles/hybrid electric cars
	green automobiles/hybrid cars/hybrid electric cars
	green vehicles/partial zero-emissions vehicles/hybrid vehicles
	bus operating companies/transit authorities with hybrid buses
	partial zero-emissions vehicles/hybrid vehicles
Luxury	car classifications/luxury vehicles
	sport utility vehicles/luxury sport utility vehicles
MiniVan	car classifications/minivans
Motorcycle	personal transporters/motorcycles
	wheeled vehicles/motorcycles
	motorcycle classifications/standard motorcycles
	lists of vehicles/lists of motorcycles
	motorcycle classifications/cruiser motorcycles
	motorcycle classifications/dual-sport motorcycles
	transport culture/motorcycling
	grand prix motorcycle racing/grand prix motorcycles
	vehicle stubs/motorcycle stubs
	motorcycling/motorcycle sport
	motorcycle classifications/sport touring motorcycles

Figure 6.8: Final results of the path excerpts leading to each of Automotive*'s subcategories (part 1).*

Off-Road Vehicles	off-roading/off-road vehicles
	dirt biking/off-road racing
	land vehicles/off-road vehicles
Performance Vehicles	
Pickup	car classifications/pickup trucks
	auto racing series/pickup truck racing series
Road-Side Assistance	emergency services/emergency road services/
Sedan	car classifications/sports sedans
	automobiles/car body styles/sedans
	auto racing series in australia/australian sports sedan championship
Trucks & Accessories	wheeled vehicles/trucks
	car classifications/pickup trucks
	transport culture/trucking subculture
	electric vehicles/electric trucks
	sport utility vehicles/sport utility trucks
	hybrid vehicles/hybrid trucks
Vintage Cars	automobiles by period/vintage vehicles
Wagon	automobiles/car body styles/station wagons

Figure 6.9: Final results of the path excerpts leading to each of Automotive*'s subcategories (part 2).*

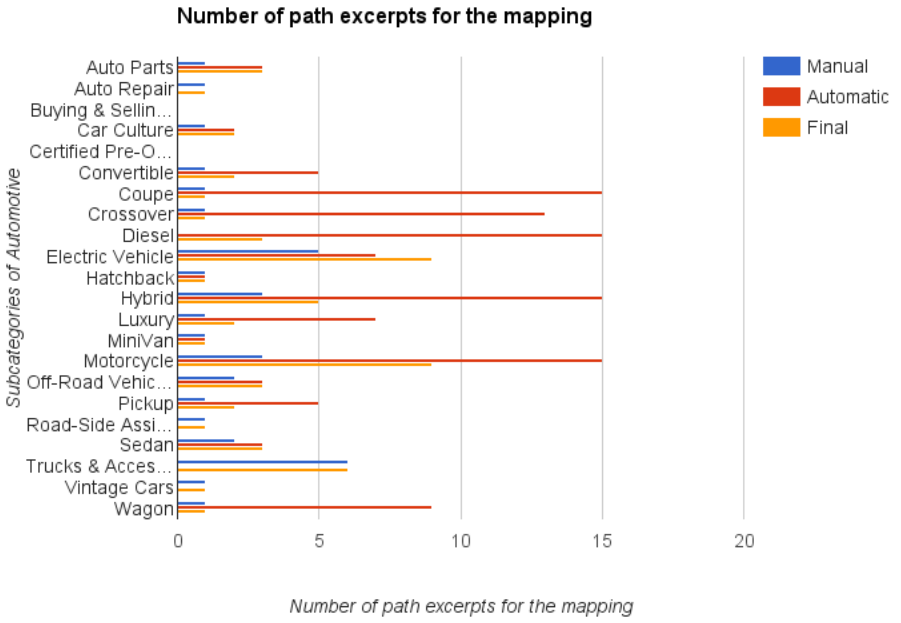


Figure 6.10: Number of paths found for each of subcategories of Automotive.

6.2 Versions of the Dictionary-Based Classifier

It is desirable to optimize the performance of the classifier. Thus, the classifier was improved by creating new versions of the classifier's dictionary. This section is dedicated to the different versions of the dictionary, where we focus on the improvements we made for each version and why these improvements were chosen. The varieties of available categories and number of dictionary entries depend on the dictionary version, because some categories were removed to focus on optimizing the results of others.

6.2.1 IAB Dictionary-1 (iab-1)

The first dictionary for the classifier was an attempt to create a mapping between keywords and categories. We started by creating mapping between the categories that we thought would be easy to map from; *Automotive*, *Science*, *Sports* and *Religion & Spirituality*.

We needed to evaluate the mapping process between keywords and categories in order to know if the mapping process worked as desired. Thus, the results of the classifier had to be compared with some correct results. The categories chosen at the first version were not good for evaluation with `www.rappler.com`. For this reason, we chose to look at the 10 first keywords for each of the categories, and manually classify these keywords. The results of the manual classification were used for a comparison of the automatic classification.

The manual classification was done by looking at the online Wikipedia article which the entry was based upon and deciding the most descriptive category based on the most descriptive categories. The results are shown in table 6.1, table 6.2, table 6.3 and table 6.4.

Dictionary entry	**Automatic mapping**	**Manual mapping**
alex fong	sports/swimming	*ambiguous*
axel rauschenbach	sports/figure skating	sports/figure skating
u.s. open - singles qualifying	sports/tennis	sports/tennis
shooting wr sk75 junior women teams	sports/ hunting& shooting	*unknown*
jorgen aukland	sports/skiing	sports/skiing
uss roebuck	sports/sailing	sports/sailing
davis phinney	sports/bicycling	sports/bicycling
ohno-group hiroshima oilers	sports/volleyball	sports/volleyball
harry jones	sports/sailing	*ambiguous*
sunshine millions distaff	sports/horse racing	sports/horse racing

Table 6.1: Results of the 10 first keywords automatically categorized to sports *when compared to a manual categorization.*

Dictionary entry	**Automatic mapping**	**Manual mapping**
aciurina	science/biology	science/biology
neurl2	science/chemistry	science/chemistry
project icarus	science/biology	ambiguous
darboux	science/space&astronomy	science/physics
sprague	science/physics	ambiguous
altenia	science/biology	ambiguous
stylochyrus	science/biology	ambiguous
distribution transformer motor	science/physics	science/physics
jarvzoo	science/biology	science/biology **or** travel/theme parks
tomopterus similis	science/biology	science/biology

Table 6.2: Results of the 10 first keywords automatically categorized to science *when compared to a manual categorization.*

Dictionary entry	**Automatic mapping**	**Manual mapping**
trojan	automotive/vintage cars	*ambiguous*
yamaha xtz 660	automotive/motorcycle	automotive/motorcycle
tatra 813	automotive/ trucks&accessories	automotive/ trucks&accessories
tatra 810	automotive/ trucks&accessories	automotive/ trucks&accessories
tatra 816	automotive/ trucks&accessories	automotive/ trucks&accessories
tatra 815	automotive/ trucks&accessories	automotive/ trucks&accessories
yamaha yz85	automotive/motorcycle	automotive/motorcycle
les schwab tire centers	automotive/auto repair	automotive/auto repair **or** automotive/trucks&accessories
man truck and bus	automotive/ trucks&accessories	automotive/ trucks&accessories
daryl ecklund	automotive/motorcycle	automotive/motorcycle **or** automotive/ crossover

Table 6.3: Results of the 10 first keywords automatically categorized to automotive *when compared to a manual categorization.*

Dictionary entry	**Automatic mapping**	**Manual Mapping**
jean baptiste perrin	religion&spirituality/ atheism&agnosticism	science/physics
carlo mazzacurati	religion&spirituality/ atheism&agnosticism	*ambiguous*
annie laurie gaylor	religion&spirituality/ atheism&agnosticism	religion&spirituality/ atheism&agnosticism
secular ethics	religion&spirituality/ atheism&agnosticism	religion&spirituality/ atheism&agnosticism
antonio carluccio	religion&spirituality/ atheism&agnosticism	food&drinks/ italian cuisine
c. delisle burns	religion&spirituality/ atheism&agnosticism	religion&spirituality/ atheism&agnosticism
irreligion in bangladesh	religion&spirituality/ atheism&agnosticism	religion&spirituality/ atheism&agnosticism
criticism of atheism	religion&spirituality/ atheism&agnosticism	religion&spirituality/ atheism&agnosticism
maryse joissains-masini	religion&spirituality/ atheism&agnosticism	Law,gov't&politics/politics
boston investigator	religion&spirituality/ atheism&agnosticism	*ambiguous*

Table 6.4: Results of the 10 first keywords automatically categorized to religion & spirituality *when compared to a manual categorization.*

Category	**Correct**	**Wrong**
Sports	7	3
Science	6	4
Automotive	9	1
Religion & Spirituality	4	6
Total	24	14

Table 6.5: Evaluation of the automatic mapping process.

Evaluation of the mapping

The results from the manual classification show that most of the selected keywords were mapped to the same categories for both the manual and the automatic classification. A total number of 24 categories were correct, which gives a correct percentage of 60% (see table 6.5). Some of the keywords were also found to be ambiguous. This means that the automatic classification is not necessarily wrong, but we would still like the classifier to skip these keywords.

The main disadvantage of this evaluation is that we only checked with a small set for each category. The task of manually classifying a large set of data is extremely time consuming, and this was the main reason for using a small set

of only 10 keywords per category. Hence, it is possible that the evaluated sets seem correct even though the overall results of the classifier are bad. However, this evaluation is created only to test if some of the classification seemed correct.

We concluded from this evaluation that we could continue with the same approach and extend the classifier with more categories.

6.2.2 IAB Dictionary-2 (iab-2)

We continued by extending the dictionary with keyword mappings to other IAB categories since we achieved positive results of the simple evaluation of *iab-1*. The next version of the dictionary was created with another of the categories considered relevant for the evaluation (e.g., *Arts & Entertainment*), and we added other categories as well.

6.2.3 IAB Dictionary-3 (iab-3)

We evaluated the results of *iab-2* with `www.rappler.com` and noticed that almost all articles were mapped to the category *Science* in *iab-2*. The reason for this was found to be that we processed the dictionary entries (e.g., removed parenthesis and years which are not likely to occur within keywords) after we removed all entries that were common words. Figure 5.31 illustrates the problem where *(85476) 1997 MY* was reduced to *my*, which is also a common English word. This means that the the common word *my* is categorized to *Science* since *(85476) 1997 MY* (a minor planet) is categorized to *Science/Space&Astronomy*. All articles containing multiple occurrences of the word *my* will accordingly be categorized to *Science*, even if the content is not science related. Thus, the main correction for this version was to remove common words after the processing was finished.

6.2.4 IAB Dictionary-4 (iab-4)

Studying some of the classification results showed that we still had many ambiguous entries in our dictionary. So far ambiguous entries were assumed to contain a parenthesis, but these entries did not contain parenthesis. Thus, we decided a new approach for finding all ambiguous entries in our dictionary. Wikipedia contains a category to keep track of all disambiguation pages which is called *All disambiguation pages*. The new approach was to find all disambiguation pages by going through `enwiki-latest-categorylinks.sql.gz` and storing all pages that had a link from this category. We also removed some categories to focus on some categories, which reduced number of entries from 617 466 to 382 544.

6.2.5 IAB Dictionary-5 (iab-5)

The results of the classifier showed that it classified few articles correctly. We looked at the results and realized that the classifier favoured short category paths. Favor of short category paths led to wrong categorization of the keywords. Thus, we decided to normalize the grading of the category paths (equation 4.3). This was done by going through category paths for all articles and

grading them again with the new formula. Finally, the top 3 category paths were picked for each article.

Version 5 (*iab-5*) was the first version that used normalized grading of the category paths.

6.2.6 IAB Dictionary-6 (iab-6)

The results for *iab-5* showed that there were still too many articles assigned to the category *Arts & Entertainment*. We realized that the main reason for this is that many keywords within this category are titles of movies, songs or books which are often quite common words or phrases consisting of common words (e.g., "dirty man" (music) or "are you blind" (music) which are phrases that might be used in articles without referring to music).

We decided to remove all entries that contained *any* of the top 1000 most common English words [58]. This reduced the number of dictionary entries from 378 307 to 234 899. The disadvantage is that we might loose information from our dictionary, but the advantage is that it might reduce the number of wrongly categorized elements.

6.2.7 Variation of Categories and Number of Entries for the Different Dictionary Versions

The results in section 6.3 show that improvements were achieved when we created new dictionary versions for the classifier. The first dictionary consisted of keywords for only a few categories, and later it was extended by new categories. This means that available categories varied for each version, and number of entries varied too. Figure 6.11 shows number of entries per category for the different dictionary versions, while table 6.6 shows the total number of entries per dictionary version. Table 6.7 shows the available categories for each dictionary version.

Dictionary version	**Number of entries**
iab-1	228 936
iab-2	668 250
iab-3	617 466
iab-4	382 544
iab-5	378 307
iab-6	234 899

Table 6.6: Number of entries in each dictionary version.

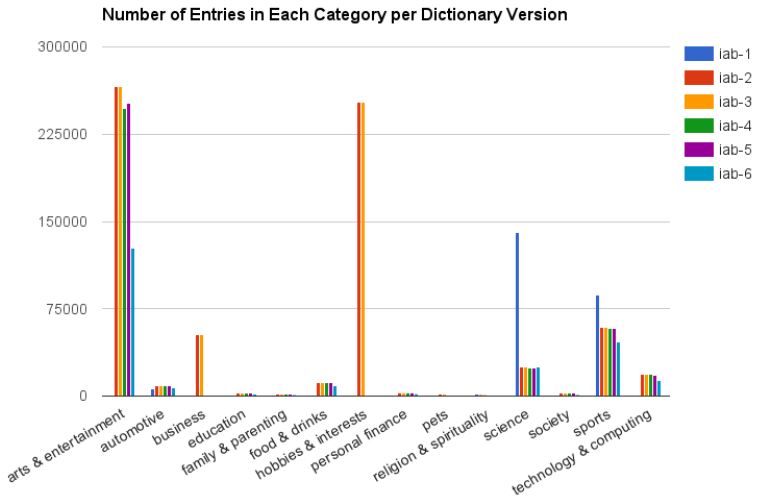


Figure 6.11: Number of entries per category for each of our dictionary versions. Notice that not all categories are present in all versions.

	iab-1	iab-2	iab-3	iab-4	iab-5	iab-6
Arts & Entertainment		X	X	X	X	X
Automotive	X	X	X	X	X	X
Business		X	X			
Education		X	X	X		
Family & Parenting		X	X	X	X	X
Food & Drinks		X	X	X	X	X
Hobbies & Interests		X	X			
Personal Finance		X	X	X	X	X
Pets		X	X			
Religion and Spirituality	X	X	X			
Science	X	X	X	X	X	X
Society		X	X	X	X	X
Sports	X	X	X	X	X	X
Technology & Computing		X	X	X	X	X

Table 6.7: Available categories for each dictionary version.

6.3 Results from the Classifier

The main purpose of the evaluation is to determine how well the classifier performs and to find possible improvements of the classifier. We applied our classifier to Rappler's web page(`www.rappler.com`) and compared the results with three categories that were available both in IAB's taxonomy and on Rappler. This section is dedicated to the results we found, including how we found the results, comparison of the results from different versions of our classifier and comparison with results of another dictionary-based classifier.

Rappler

Our project was tested at the webpage `www.rappler.com` which is an online Indonesian news site where most articles are written in English. This webpage was chosen because it is a well-structured newspaper which contains categories suitable for our evaluation. Articles on Rappler are sorted by the publishers based on the articles' contents. The available categories and their subcategories are shown in table 6.8.

Main category	**Subcategories**
News	Philippines, World, #BalikBayan, Science & Nature, Specials
Video	Newscast, Shows, Reports, Documentary, Specials
Business	Economy, Brighter Life, Industries, Money, Features, Specials
MoviePH	Issues, #ProjectAgos, #BudgetWatch, #HungerProject, Community, IMHO
Views	Thought Leaders, iSpeak, Rappler Blogs, #AnimatED
Life & Style	Food, Books, Arts & Culture, Travel, Specials, #PugadBaboy
Entertainment	Entertainment News, Movies, Music, Special Coverage
Sports	Boxing, Basketball, Football, Other Sports, University Sports
Tech	News, Features, Reviews, Hands on, Social Media
Live	#RStream, Newscast
BrandRap	Stories, Specials, #BuildWealth, #HomeMagic, #BrighterLife, #BetterWorld

Table 6.8: Rappler's available categories and subcategories.

We have focused on 3 categories which are present both on Rappler and in IAB's taxonomy:

- *Sports*
- *Entertainment/Arts & Entertainment*
- *Tech/Technology & Computing.*

These categories were evaluated for all versions of the classifier (if present). The evaluation of these categories were chosen to see how well the classifier performs, but also to find possible improvements for later versions of our classifier.

6.3.1 Retrieving Results from Cxense

The results of our classifier were retrieved from *Cxense Insight*, which is a software tool for analyzing user behavior for a specific web page. This software tool makes it possible to retrieve urls with certain tags which is ideal for our evaluation. We chose to look at results from the last 5 days[10]. Figure 6.12 shows an example of code for retrieving all articles which contain the tag *Sports* within both the url taxonomy and *igg-iabtaxonomy6*. We did similarly to retrieve articles with none of the tags or with only one of them. As an example: all articles with only the tag *Sports* in *igg-iabtaxonomy6* are retrieved by changing line 10 in figure 6.12 to the line shown in figure 6.13.

```
1  cx.py /traffic
2  '{ "siteId":"9222338298879175891",
3  "groups":["url"],
4  "start":"-5d",
5  "fields":["urls"],
6  "filters":[
7  {"type":"and", "filters":[
8  {"type":"keyword", "group":"language","item":"en"},
9  {"type":"keyword", "group":"'igg-iabtaxonomy6'", "item
       ":"'sports'", "minWeight":'0.5'},
10 {"type":"keyword", "group":"taxonomy", "item":"'sports
       '"}],
11 "count":1000}'
```

Figure 6.12: Example of code for retrieving all events with Sports *within the url taxonomy and within igg-iabtaxonomy6.*

```
{"type":"not", "filter":{"type":"keyword", "group":"
   taxonomy", "item":"'sports'"}}]}
```

Figure 6.13: Example of code excerpt for retrieving elements without Sports *in the taxonomy.*

All evaluation scores are based on the formulas in section 4.3. However, we have multiplied all score values (precision, recall, accuracy and F_1-score) with 100. This means that the range of the results are between 0 (worst) and 100 (best).

[10]The results for evaluation were retrieved 23rd of July 2015.

Bias with our evaluation

Some of Rappler's articles are not written in English. These articles cannot be classified by our classifier since it is based on an English dictionary. We have tried to select all English articles by looking at articles containing the tag "language":"en" (line 8 in 6.12), but articles in multiple languages might give wrong results.

6.3.2 Weight for Classification

Another issue is to decide the boundaries of the classification, i.e., minimum number of keyword occurrences necessary for categorizing an article to a class. This can be found by setting a boundary which determines if the article should be returned or not. It is not desirable to set the boundary to a number of occurrences, but instead let a weight determine the relevancy of the category. Figure 6.12 has a minimum weight of 0.5 for the tag *Sports* in *igg-iabtaxonomy6* (see line 9), which means that articles need a minimum of 0.5 for this tag to be returned. However, the question is to determine the minimum weight so that the results of the classifier is as best as possible.

We tested different values of the minimum weight to determine when the classifier returned the best results. The range of the minimum weight was tested from 0.1 to 0.9 with intervals of 0.1 for dictionary version 6 (*iab-6*) and for all three evaluation categories; *Sports*, *Arts & Entertainment* and *Technology & Computing*. The results of this test can be viewed in table 6.9, table 6.10 and table 6.11. The results in these tables were used to find the evaluation scores for each of the categories; these are shown in table 6.12, table 6.13 and table 6.14 (grey columns mark the best results for each category).

Sports

	0.1	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
TP	1335	817	817	817	817	581	581	287	25
TN	22347	25290	25290	25290	25290	25889	25889	26369	26493
FN	1049	1560	1559	1559	1559	1796	1796	2077	2341
FP	4002	1121	1121	1121	1121	596	596	173	5

Table 6.9: Classification results for iab-6 *and* Sports *with different values of minimum weight.*

Arts & Entertainment

	0.1	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
TP	2574	2576	2576	2576	2578	2578	2578	2578	2578
TN	23	23	23	23	23	26	26	55	196
FN	1	1	1	1	1	1	1	1	2
FP	26630	26630	26630	26630	26630	26630	26630	26569	26488

Table 6.10: Classification results for iab-6 *and* Arts & Entertainment *with different values of minimum weight.*

Technology & Computing

	0.1	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
TP	175	93	93	93	93	64	64	31	7
TN	25589	28022	28022	28022	28018	28267	28267	28474	28512
FN	346	429	429	429	429	456	456	490	513
FP	2618	543	543	543	543	237	237	71	6

Table 6.11: Classification results for iab-6 *and* Technology & Computing *with different values of minimum weight.*

Sports

	0.1	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
Precision	25.01	42.16	42.16	42.16	42.16	49.36	49.36	62.39	83.33
Recall	56.00	34.37	34.39	34.39	34.39	24.44	24.44	12.14	1.06
Accuracy	82.42	90.69	90.69	90.69	90.69	91.71	91.71	92.22	91.87
F1-score	34.58	37.87	37.88	37.88	37.88	32.70	32.70	20.33	2.09

Table 6.12: Evaluation scores for iab-6 *and* Sports *with different values for minimum weight.*

Arts & Entertainment

	0.1	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
Precision	8.81	8.82	8.82	8.82	8.83	8.83	8.83	8.84	8.87
Recall	99.96	99.96	99.96	99.96	99.96	99.96	99.96	99.96	99.92
Accuracy	8.89	8.89	8.89	8.89	8.90	8.91	8.91	9.02	9.48
F1-score	16.20	16.21	16.21	16.21	16.22	16.22	16.22	16.25	16.29

Table 6.13: Evaluation scores for iab-6 *and* Arts & Entertainment *with different values for minimum weight.*

Technology & Computing

	0.1	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
Precision	6.27	14.62	14.62	14.62	14.62	21.26	21.26	30.39	53.85
Recall	33.59	17.82	17.82	17.82	17.82	12.31	12.31	5.95	1.35
Accuracy	89.68	96.66	96.66	96.66	96.66	97.61	97.61	98.07	98.21
F1-score	10.56	16.06	16.06	16.06	16.06	15.59	15.59	9.95	2.63

Table 6.14: Evaluation scores for iab-6 *and* Technology & Computing *with different values for minimum weight.*

As discussed in section 4.3.2, there is a trade-off between precision and recall. We can see that precision is higher for high numbers of minimum weight, while recall is higher for low numbers of minimum weight in all three tables. However, F_1-score is a combination of precision and recall, and is the evaluation score we want to optimize.

The evaluation scores of all three categories show that the best results were achieved with different minimum weights. Thus, we decided to find the global results of the classifier, i.e., summarize the results of all three categories to find the global results. Table 6.15 contains the global classification results, and table 6.16 contains the global evaluation scores.

The best F_1-score is found to be best when the minimum weight is 0.1 (19.08%) and the second best is found when the minimum weight is 0.5 (18.72%) (see table 6.16). Both the global precision and recall are best for low values of the minimum weight, which is different from the independent evaluation scores. However, we chose to use 0.5 as the minimum weight value because this is also one of the best values for both *Sports* and *Technology & Computing*, while *Arts & Entertainment* had best F_1-score for 0.9.

Total Classification Scores

	0.1	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
TP	4084	3486	3486	3486	3488	3223	3223	2896	2610
TN	47959	53335	53335	53335	53331	54182	54182	54898	55201
FN	1396	1990	1989	1989	1989	2253	2253	2568	2856
FP	33250	28294	28294	28294	28294	27463	27463	26813	26499

Table 6.15: Classification results for iab-6 *for minimum weights between 0.1 and 0.9 when all three classes are summarized.*

Total Evaluation Scores

	0.1	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
Precision	10.94	10.97	10.97	10.97	10.97	10.50	10.50	9.75	8.97
Recall	74.53	63.66	63.67	63.67	63.68	58.86	58.86	53.00	47.75
Accuracy	60.03	65.23	65.23	65.23	65.23	65.89	65.89	66.30	66.32
F1-score	19.08	18.71	18.71	18.71	18.72	17.83	17.83	16.47	15.10

Table 6.16: Evaluation scores for iab-6 *for all three classes, based on the summarized classification results.*

6.3.3 Results for Sports

We started by evaluating the results for the IAB category *Sports* for the different versions of the classifier. Table 6.17 shows the results of the different versions of the classifier. We can see that number of TP (correctly categorized by our classifier) starts with 233 for *iab-1* and 243 for *iab-2*, before it is reduced to 76 in *iab-3*. However, it is important to notice that number of FP (wrongly

categorized by our classifier) is also reduced from 1756 in *iab-2* to 44 in *iab-3*. We can also see that number of correctly categorized articles are higher in the later versions of the classifier than in the first versions.

Table 6.18 contains the evaluation scores for *Sports* for the different versions of the classifier. It is noticeable that *iab-2* has a higher F_1-score than *iab-3* and *iab-4*, but we can also see that the results from *iab-5* are considerably better.

The trade-off between precision and recall is clear in table 6.18 when we look at the results for *iab-5* and *iab-6*. The precision for *iab-5* is 73% which is very good, while the precision for *iab-6* is reduced to 42%. However, the F_1-score is clearly better for *iab-6*, where it is 38% compared to 29% for *iab-5*.

Classification results for *Sports*

	iab-1	**iab-2**	**iab-3**	**iab-4**	**iab-5**	**iab-6**
TP	233	243	76	46	413	809
TN	24555	24414	26474	26493	26298	25278
FN	2127	2132	2294	2318	1955	1560
FP	1843	1756	44	25	154	111

Table 6.17: Classification results for the different versions of the classifier for the category Sports.

Evaluation scores for *Sports*

	iab-1	**iab-2**	**iab-3**	**iab-4**	**iab-5**	**iab-6**
Precision	11.2235	12.1561	63.3333	64.7887	72.8396	41.9606
Recall	9.8729	10.2316	3.2068	1.94589	17.4409	34.1494
Accuracy	86.1951	86.3794	91.9067	91.8877	92.6822	90.6869
F1-score	15.7215	11.1111	6.1044	3.7782	28.1431	37.6542

Table 6.18: Evaluation scores for the different versions of the classifier when classifying Sports.

6.3.4 Results for Arts & Entertainment

Rappler contains a category called *Entertainment*, which corresponds to IAB taxonomy's category *Arts & Entertainment*. We retrieved all classified results for evaluating this category from *Cxense Insight* similarly as for *Sports*.

The results of the classification is shown in table 6.19, where the category is not present for the first version and the results for this version are therefore marked as *NaN*[11]. The next two versions (*iab-2* and *iab-3*) are almost identical for this category. The results showed that few articles where categorized to these classes: a total number (corresponds to TP + FP) of 3 061 articles were

[11] NaN stands for *Not A Number* and is used to mark that there does not exist any data for this version.

classified for *iab-2* and *iab-3*, and 3 198 for *iab-4*. Table 6.20 shows that these results gave a low score for both precision and recall, and it was desirable to assign more articles to the class.

Classification results for *Arts & Entertainment*

	iab-1	**iab-2**	**iab-3**	**iab-4**	**iab-5**	**iab-6**
TP	NaN	261	261	261	2578	2586
TN	NaN	23741	23741	23625	970	23
FN	NaN	2320	2320	2320	12	1
FP	NaN	2755	2755	2937	25624	26616

Table 6.19: Classification results for the different versions of the classifier for the category Arts & Entertainment.

Evaluation scores for *Arts & Entertainment*

	iab-1	**iab-2**	**iab-3**	**iab-4**	**iab-5**	**iab-6**
Precision	NaN	8.6539	8.6539	8.1614	9.1412	8.8556
Recall	NaN	10.1124	10.1124	10.1124	99.5367	99.9613
Accuracy	NaN	82.5463	82.5463	81.9614	12.1573	8.9270
F1-score	NaN	9.3264	9.3264	9.0328	16.7446	16.2698

Table 6.20: Evaluation scores for the different versions of the classifier when classifying Arts & Entertainment.

Version 5 (*iab-5*) was the version where when we applied the normalized grading, and this helped a lot for categorizing more articles to the category. The precision did not change much for this version (8% for *iab-4* and 9% for *iab-5*), but the recall went from 10% for *iab-4* to 99.5% with *iab-5*.

The evaluation scores in table 6.20 show that the best results were achieved by version 5 (*iab-5*) for this class, but the difference in the F_1-score values of *iab-5* and *iab-6* is very small.

6.3.5 Results for Technology & Computing

Rappler contains a category called *Tech*, which is connected to news about technology (*Technology & Computing* in IAB's taxonomy). This category was evaluated similarly as the other categories (*Sports* and *Arts & Entertainment*).

Table 6.21 contains the classification results and table 6.22 contains the evaluation scores for the category. We can see that few articles were categorized to the category by the first versions of the classifier with a recall less than 1% for the *iab-3* and *iab-4*. The last version of the classifier (*iab-6*) is the classifier with best results. This classifier has a lower precision than the other versions, but the recall is considerably better. This makes the F_1-score best for *iab-6* with a F_1-score at 16%, which is almost 5 times better than the F_1-score for *iab-3*.

Classification results for *Technology & Computing*

	iab-1	**iab-2**	**iab-3**	**iab-4**	**iab-5**	**iab-6**
TP	NaN	NaN	10	7	29	94
TN	NaN	NaN	28516	28533	28458	28048
FN	NaN	NaN	512	514	491	428
FP	NaN	NaN	37	14	72	543

Table 6.21: Classification results for the different versions of the classifier for the category Technology & Computing.

Classification results for *Technology & Computing*

	iab-1	**iab-2**	**iab-3**	**iab-4**	**iab-5**	**iab-6**
Precision	NaN	NaN	21.2766	33.3333	28.7129	14.7567
Recall	NaN	NaN	1.9157	1.3436	5.5770	18.0077
Accuracy	NaN	NaN	98.1118	98.1836	98.0620	96.6647
F1-score	NaN	NaN	3.5150	2.5830	9.3398	16.2209

Table 6.22: Evaluation scores for the different versions of the classifier when classifying Technology & Computing.

6.3.6 Global Evaluation of the Classifier

It is desirable to evaluate the categories together in addition to evaluating all three categories independently. This was done by summarizing all classification results for all three classes, and computing global evaluation scores based on these. We chose to evaluate the latest version of the classifier for all three classes for this task. Table 6.23 contains the summarized results and table 6.24 contains the evaluation scores based on these.

The global evaluation scores for all three classes are found to have a low precision, which means that only 11% of the classified articles contain the desired tag. The recall is 64% which shows that our classifier is able to find most of the correct articles. The final F_1-score is low for our classifier, only 19%. Discussion about this is found in section 6.5.

Global Classification Results

	Sports	**Entertainment**	**Tech**	**Total**
TP	809	2586	94	3489
TN	25278	23	28048	53349
FN	1560	1	428	1989
FP	111	26616	543	27270

Table 6.23: Classification results when summarizing all classification results for all three categories.

Global Evaluation Scores

	Sports	Entertainment	Tech	Total
Precision	41.96	8.86	14.76	11.34
Recall	34.15	99.96	18.01	63.69
Accuracy	90.69	8.93	96.66	66.01
F1-score	37.65	16.27	16.22	19.25

Table 6.24: Evaluation scores based on the summarized classification results for all three categories.

6.3.7 Comparison with Another Classifier

It is interesting to compare the results of our classifier with results of another classifier. The main reason for this is to see how well the classifier performs.

We compared our results with the results of the dictionary-based classifier in *Entity Extraction, Linking, Classification, and Tagging for Social Media: A Wikipedia-based Approach* [10]. The results of their classifier is presented in table 2 in the paper and contained precision, recall and F_1-score.

Table 6.25 contains their results compared with our results[12]. The comparison of the results shows that the classifier in [10] have higher evaluation scores than our classifier. However, the exception is recall for *Arts & Entertainment* where our classifier has 99.96% compared to 70% at [10]. In addition, it is noticeable that the classifier in [10] was best for *Sports* and *Technology & Computing* just like our classifier, and has lower F_1-score for *Arts & Entertainment*.

It is also important to note that [10] have extended the dictionary-based classifier with information from other resources than Wikipedia. Classification of *Arts & Entertainment* is probably easier when including information from *MusicBrainz*, while the other resources might help classification of the other categories.

	Sports		Entertainment		Technology	
	iab-6	**tweet**	**iab-6**	**tweet**	**iab-6**	**tweet**
precision	41.96	60	8.86	53.85	14.76	60.00
recall	34.15	100.00	99.96	70.00	18.01	100.00
F_1**-score**	37.65	75.00	16.27	60.87	16.22	75.00

Table 6.25: Comparison of the three categories Sports, Arts & entertainment *and* Technology & Computing*. The results are compared with the classifier from [10].*

[12]In table 6.25 *tweet* is the classifier of [10], while *iab-6* is the results of version 6 of our classifier.

6.4 Evaluation of the Norwegian Classifier

Finally, we wanted to evaluate the results from the Norwegian classifier. This was done by comparing the Norwegian classifier with a Norwegian online news paper (Adressa). This section is dedicated to the results found in the evaluation process.

Adressa

The Norwegian classifier was evaluated with Adressa (`www.adressa.no`). This online newspaper contains news articles written in Norwegian, and the articles are sorted into categories by the publisher just like Rappler. We decided to evaluate the classifier with the same categories as the English classifier, but only two of the categories were present at Adressa. Hence, the evaluation categories were:

- *Sport (Sports)*
- *Kultur (Arts & Entertainment)*

Retrieving results from the Norwegian classifier

We used *Cxense Insight* to retrieve the results from the Norwegian classifier in a similar way as we did for the English classifier (see section 6.3.1). The main difference was that we were now interested in Norwegian news articles and looking for articles containing tags used by Adressa. Figure 6.14 is an example of code for retrieving all articles categories to both *Arts & Entertainment* by *igg-noiabtaxonomy1* and to *kultur* by the taxonomy.

```
1 cx.py /traffic
2 '{"siteId":"9222270286501375973",
3 "groups"frown emoticon"url"],
4 "start":"-5d",
5 "fields"frown emoticon"urls"],
6 "filters"frown emoticon
7 {"type":"keyword","group":"igg-noiabtaxonomy1","item
     ":"arts & entertainment"},
8 {"type":"keyword","group":"taxonomy","item":"'kultur
     '"}],
9 "count":1000}'
```

Figure 6.14: Example of code for retrieving all Norwegian articles categorized to Arts & Entertainment *by* igg-noiabtaxonomy1 *and to* kultur *by the taxonomy.*

The results from the Norwegian classifier

The results from the Norwegian classifier was collected and stored in table 6.26, including the combined results of both *Sports* and *Arts & Entertainment*. Table 6.27 contains the evaluation scores based on these results. We can see that the category *Arts & Entertainment* achieved the highest F_1-score in contrast to the

English classifier. However, *Sports* has highest precision just like for the English classifier and *Arts & Entertainment* has highest recall.

	Sport	**Kultur**	**Total**
TP	165	774	939
TN	18030	14808	32838
FN	1677	1292	2969
FP	92	3467	3559

Table 6.26: Classification results for the Norwegian classifier for the categories Sports, Kultur *(*Arts & Entertainment*) and the combination of the two categories.*

	Sport	**Kultur**	**Total**
Precision	64.2023	18.2504	20.8759
Recall	8.9577	37.4637	24.0276
Accuracy	91.1391	76.6039	83.8035
F1-score	15.7218	24.5442	22.3411

Table 6.27: Evaluation scores for the Norwegian classifier for the categories Sports, Kultur *(*Arts & Entertainment*) and the combination of the two categories.*

The overall results of the Norwegian classifier is surprisingly good with an F_1-score of 22% compared to 19% for the English classifier. We assume that the Norwegian classifier might perform even better if its dictionary is extended (it only contained 21 320 entries). Another improvement would be to add distinctively Norwegian words and phrases, which are words or phrases that only occur in the Norwegian Wikipedia.

It would also be interesting to create a Norwegian dictionary-based classifier by the same approach as the English classifier (i.e., create category paths for all Norwegian articles and grade the category paths). Comparing these results would give a good indication of how successful our simple approach was.

6.5 Discussion of the Results

The results of our classifier shows that all versions of the classifier categorizes too many articles to all categories, i.e., too many false positive (FP) for each category. The main reason for this is that we have created a one-to-many classifier, i.e., our dictionary-based classifier can assign more than one class to each article, while we compare with a one-to-one classification, i.e., each article is placed under one class in its url structure. This means that articles might be evaluated as wrongly classified even though the results of the classifier are correct.

Wrongly categorized articles considered correct

We studied the results of the English classifier and found multiple articles where the results of our classifier were considered wrong by the url structure and correct by us. These articles were found by looking at the titles and the url structure, and seeing if we could find any articles that might be incorrect. We have chosen to present two articles where we agree with the tag *Arts & Entertainment* from the classification results, but where the tag *Entertainment* is not present in the url structure.

The first article were *Myanmar movie star, 4 others win Ramon Magsaysay Awards* which is an article about this year's winners of Asia's Magsaysay Awards [27]. The classification results of this article is shown in figure 6.15 where the article is placed under the category *Arts & Entertainment* by many versions of our classifier. However, the url structure of the article is shown in figure 6.16 and does not contain the tag *Entertainment*. Another example is a chronicle about the dating app *Tinder* (*My Tinder experience: A woman's perspective* [26]) which is also classified as entertainment by our classifier (see figure 6.17). This article's url structure is shown in figure 6.18 and is also without *Entertainment* within the url structure and therefore considered wrong.

TOP KEYWORDS

Keyword	Keyword Group	Weight	Count
asia pacific	entity	1.00	12
article	pageclass	1.00	1
arts & entertainment	igg-iabtaxonomy3	1.00	88
en	language	1.00	1
rappler.com	site	1.00	1
arts & entertainment	igg-iabtaxonomy6	0.98	81
arts & entertainment/music	igg-iabtaxonomy3	0.97	75
arts & entertainment	igg-iabtaxonomy4	0.97	74
arts & entertainment/music	igg-iabtaxonomy6	0.96	69
arts & entertainment/music	igg-iabtaxonomy4	0.95	65

Figure 6.15: Classification results of the article Myanmar movie star, 4 others win Ramon Magsaysay Awards*, where both the classifier and we agree that the article could also be placed under the category* Entertainment.

```
http://www.rappler.com/world/regions/asia-pacific
    /100927-ramon-magsaysay-award-winners-2015
```

Figure 6.16: The url structure of the article Myanmar movie star, 4 others win Ramon Magsaysay Awards

It is not hard to find articles where we disagree with the results considered correct. We could therefore argue that our choice of evaluation sites might be not optimal for evaluating our project.

TOP KEYWORDS

Keyword	Keyword Group	Weight	Count
tinder	entity	1.00	15
article	pageclass	1.00	1
arts & entertainment	igg-iabtaxonomy3	1.00	200
en	language	1.00	1
rappler.com	site	1.00	1
arts & entertainment/music	igg-iabtaxonomy3	0.97	172
arts & entertainment	igg-iabtaxonomy6	0.95	150
arts & entertainment/music	igg-iabtaxonomy6	0.93	137
arts & entertainment	igg-iabtaxonomy4	0.91	121
arts & entertainment/music	igg-iabtaxonomy4	0.90	111

Figure 6.17: Classification results of the article My Tinder experience: A woman's perspective*, where both the classifier and we agree that the article could also be placed under the category* Entertainment*.*

```
http://www.rappler.com/world/regions/asia-pacific/
    indonesia/100228-my-tinder-experience-woman-
    perspective
```

Figure 6.18: The url structure of the article My Tinder experience: A woman's perspectives

IAB's taxonomy as category set and Wikipedia as keyword list

We chose to use IAB's taxonomy as category set because IAB is one of the premier research organizations within advertising. However, we tested our classifier on online newspapers and discovered that the IAB taxonomy was not ideal as output category for this. Both Rappler and Adressa are serious news papers without many articles about topics like *Travel*, *Hobbies & Interests* etc. Thus, a better output category set would probably be a set containing more news related categories.

Our keyword list (entries in our dictionary) were based on titles in Wikipedia. Wikipedia is one of the largest encyclopedias and contain many articles within certain fields. However, Wikipedia lacks articles within some of the categories in the IAB taxonomy; there are few articles about *Style & Fashion* or *Home & Garden*. Thus, our classifier would probably be improved by adding information from more everyday knowledge resources, so that there exists keywords to most of the categories in our output category set.

Chapter 7

Conclusion and Further Works

This is the final chapter of our project and covers our conclusion for the project and desirable further works. The chapter starts with the conclusion of the project, before mentioning some of the desirable further works that might improve the classification results of our classifier.

7.1 Conclusion

Automatic content categorization is useful for building up user profiles and in the task of automatically deciding advertisements on web pages. We chose to create a dictionary-based classifier because it is easy to understand for brokers (which are often non-technical) and because it is based on a dictionary that easily can be modified to satisfy specific purposes.

Our classifier is based a dictionary where the entries are created from titles of Wikipedia articles. Each dictionary entry is connected to category from IAB's taxonomy, where we explored the underlying category structure of Wikipedia in order to create an automatic mapping between these. Our overall goal was to determine whether articles could be correctly categorized based on just the Wikipedia article titles and the underlying category structure.

We evaluated the classifier's results by comparing the results with url structures of articles. The sites used for the evaluation were `www.rappler.com` for the English classifier and `www.adressa.no` for the Norwegian classifier.

The English classifier was evaluated with 3 categories: *sports*, *arts & entertainment* and *technology*. We improved our classifier by creating new versions of its dictionary. The evaluation results showed that the later versions of the classifier were considerable better, i.e., higher evaluation scores for the later versions.

The results of our classifier showed that it is possible to determine the content of some articles just by exploring titles of Wikipedia articles and the underlying category structure. However, many articles were wrongly categorized when compared to the url structure. This might be because we developed a one-to-many classifier which means that the classifier can classify an article to more than one class, while the classification results are compared to a one-to-one

classification where an article contains only one class within the url structure. We found several examples of articles that were considered wrongly classified by the evaluation scores, but considered correctly classified by us.

We decided to compare the results of our English classifier with [10], because this classifier contained all three classes. Comparison showed that the classifier in [10] achieved higher evaluation scores than ours. However, it is important to notice that [10] added knowledge in addition to Wikipedia, including *MusicBrainz* which is most likely very helpful for optimizing the categorization of *arts & entertainment*. Even though the evaluation scores were higher, we could see that the classification results of [10] shows similar results as ours; *sports* were found to be easier to classify than *arts & entertainment* and *technology*.

The creation of the Norwegian classifier was based on a simple idea; all English entries in the classifier's dictionary were translated to Norwegian by using the internal language links within Wikipedia. Finally, we removed all words and phrases that were ambiguous in Norwegian and this resulted in a small Norwegian dictionary which could be used by a classifier. Only two of the categories were available on `www.adressa.no`, so we evaluated *sports* with *sport* and *arts & entertainment* with *kultur*.

The Norwegian classifier performed surprisingly well considering the simple approach for creation and that it contained few entries in its dictionary. However, an improvement of the classifier would be to add words or phrases that are distinctively Norwegian and not found in the English Wikipedia.

Finally, our conclusion is that it is possible to get reasonably good results from our classifiers just by exploring the titles of Wikipedia articles and the underlying category structure. The results of the classifier can be improved by modifying the dictionary it is based on, but the classifier is already able to give a good indication of the content of an article.

7.2 Further Works

There exists many desirable extensions for our dictionary-based classifier that might improve the results or expand the usage. The most important future improvements for our classifier is solving ambiguity in a better way and extending the dictionary by exploring more than just the titles of Wikipedia articles. Expanding the usage is possible if the classifier is well-defined for other languages than just Wikipedia.

Disambiguation

Our project removed all ambiguous titles. This means that we loose information that might be valuable for classification purposes. Instead of removing the titles, we could keep the titles that are most relevant for our classification, for instance the titles that have the longest articles. Thus, solving disambiguation instead of removing all ambiguous titles might improve the classification results. We studied different projects for solving disambiguation, and some of their findings could be applied to find the most likely meaning of an ambiguous entry.

Stubs

Another possible change for the program is to remove Wikipedia stubs which might improve the classification. Wikipedia stubs are pages that are too short to be considered articles. Wikipedia contains 1 913 507 stubs [40], and these articles might provide ambiguous information which are more likely to be removed since they contain so little information that they are not considered articles.

Explore more information from Wikipedia

We have only looked at the titles when creating the dictionary-based classifier. Another extension would be to explore the actual content of the articles before they are classified to the most describing categories. Better categorization of the Wikipedia articles could lead to better results for the classifier, which could improve the results.

Adding information

We have only used information from Wikipedia, but it might be desirable to extend our classifier with information in addition to Wikipedia. Keywords from other sources might improve the results like it did in [10] when adding information from MusicBrainz, City DB, Yahoo! Stocks, Chrome and Adam.

Improve Mapping

The mapping between keywords and categories could also be improved by creating better decision rules between category paths and IAB categories. We chose to grade our Wikipedia article titles by using inlink and outlink number. Another improvement could be to try other grading algorithms, which might be better for determining the content of the articles.

Extending for more languages

The results and implementation is created for the English Wikipedia, hence only useful for English articles. We created a Norwegian dictionary-based classifier by using the internal Wikipedia links to translate the English classifier's dictionary to Norwegian. We noticed that the Norwegian classifier lacked important information for being able to classify Norwegian articles, and concluded that this is probably because special Norwegian keywords are missing from the dictionary.

Thus, a desirable extension would be to create a more general approach for creating the dictionary so that it could be applied to other languages as well. Most of our programs are not dependent on language, except for the mapping rules. A good extension would be to create a language independent mapping process which could create dictionary-based classifiers from Wikipedia in multiple languages.

References

Main References

[1] *About the IAB*. [Online; accessed 16-January-2015]. URL: http://www.iab.net/about_the_iab.

[2] Sergey Chernov et al. "Extracting Semantics Relationships between Wikipedia Categories." In: *SemWiki* 206 (2006).

[3] Google Code. *Norwegian stop-words*. [Online; accessed 14-May-2015]. 2015. URL: https://code.google.com/p/stop-words/source/browse/trunk/stop-words/stop-words/stop-words-norwegian.txt?r=3.

[4] Silviu Cucerzan. "Large-Scale Named Entity Disambiguation Based on Wikipedia Data." In: *EMNLP-CoNLL*. Vol. 7. 2007, pp. 708–716.

[5] Cxense. *Cxense –About*. [Online; accessed 7-May-2015]. 2015. URL: http://www.cxense.com/about.

[6] *Data Cleaning is a critical part of the Data Science process*. [Online; accessed 22-April-2015]. 2015. URL: http://blog.revolutionanalytics.com/2014/08/data-cleaning-is-a-critical-part-of-the-data-science-process.html.

[7] WordStream Online Advertising Made Easy. *Cost Per Action (CPA): How to Lower Your CPA in AdWords*. [Online; accessed 6-May-2015]. 2015. URL: http://www.wordstream.com/cost-per-action.

[8] WordStream Online Advertising Made Easy. *Cost Per Click (CPC): Learn What Cost Per Click Means for PPC*. [Online; accessed 6-May-2015]. 2015. URL: http://www.wordstream.com/cost-per-click.

[9] Evgeniy Gabrilovich and Shaul Markovitch. "Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge". In: *Proceedings of the Twenty-First National Conference on Artificial Intelligence*. Boston, MA, 2006, pp. 1301–1306. URL: http://www.cs.technion.ac.il/~shaulm/papers/pdf/Gabrilovich-Markovitch-aaai2006.pdf.

[10] Abhishek Gattani et al. "Entity Extraction, Linking, Classification, and Tagging for Social Media: A Wikipedia-based Approach". In: *Proc. VLDB Endow.* 6.11 (2013), pp. 1126–1137. ISSN: 2150-8097. DOI: 10.14778/2536222.2536237. URL: http://dx.doi.org/10.14778/2536222.2536237.

[11] Tom Gruber. *Onotology*. [Online; accessed 13-May-2015]. URL: http://tomgruber.org/writing/ontology-definition-2007.htm.

[12] Xianpei Han and Jun Zhao. "Named entity disambiguation by leveraging wikipedia semantic knowledge". In: *Proceedings of the 18th ACM conference on Information and knowledge management*. ACM. 2009, pp. 215–224.

[13] Google AdWords Help. *Cost-per-thousand impressions (CPM)*. [Online; accessed 6-May-2015]. 2015. URL: https://support.google.com/adwords/answer/6310?hl=en.

[14] Daniel Jurafsky and H James. "Speech and language processing an introduction to natural language processing, computational linguistics, and speech". In: (2000).

[15] Milen Kouylekov and University of Oslo Stephan Oepen. *Classification*. [Online; accessed 16-April-2015]. 2015. URL: http://www.uio.no/studier/emner/matnat/ifi/INF4820/h14/slides/06-classification.pdf.

[16] Natalia Kozlova. "Automatic ontology extraction for document classification". PhD thesis. Saarland University, 2005.

[17] *Lemmatization Lists, Datasets by MBM*. [Online; accessed 10-February-2015]. 2015. URL: http://www.lexiconista.com/datasets/lemmatization/.

[18] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schutze. *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008. ISBN: 0521865719, 9780521865715.

[19] Marketing-Schools.org. *Performance Marketing*. [Online; accessed 6-May-2015]. 2015. URL: http://www.marketing-schools.org/types-of-marketing/performance-marketing.html.

[22] Tomas Mikolov et al. "Distributed representations of words and phrases and their compositionality". In: *Advances in Neural Information Processing Systems*. 2013, pp. 3111–3119.

[23] Vivi Nastase and Michael Strube. "Decoding Wikipedia Categories for Knowledge Acquisition." In: *AAAI*. 2008, pp. 1219–1224.

[24] Simone Paolo Ponzetto and Roberto Navigli. "Large-Scale Taxonomy Mapping for Restructuring and Integrating Wikipedia." In: *IJCAI*. Vol. 9. 2009, pp. 2083–2088.

[26] Rappler. *My Tinder experience: A woman's perspective*. [Online; 30-July-2015]. 2015. URL: http://www.rappler.com/world/regions/asia-pacific/indonesia/100228-my-tinder-experience-woman-perspective.

[27] Rappler. *Myanmar movie star, 4 others win Ramon Magsaysay Awards*. [Online; 30-July-2015]. 2015. URL: http://www.rappler.com/world/regions/asia-pacific/100927-ramon-magsaysay-award-winners-2015.

[28] Peter Schönhofen. "Identifying document topics using the Wikipedia category network". In: *Web Intelligence and Agent Systems* 7.2 (2009), pp. 195–207.

[29] Richard Wray The Guardian. *Google users promised artificial intelligence*. [Online; accessed 5-May-2015]. URL: http://www.theguardian.com/technology/2006/may/23/searchengines.news.

[30] *Understanding Online Advertising — How Does It Work?* [Online; accessed 5-May-2015]. 2015. URL: https://www.networkadvertising.org/understanding-online-advertising/how-does-it-work.

[31] *Understanding Online Advertising — What is it?* [Online; accessed 5-May-2015]. 2015. URL: https://www.networkadvertising.org/understanding-online-advertising/what-is-it.

[32] *Unicode Character 'LATIN CAPITAL LETTER S WITH CEDILLA' (U+015E)*. [Online; accessed 1-April-2015]. URL: http://www.fileformat.info/info/unicode/char/015E/index.htm.

[33] *Unicode Character 'LATIN CAPITAL LETTER S WITH COMMA BELOW' (U+0218)*. [Online; accessed 1-April-2015]. URL: http://www.fileformat.info/info/unicode/char/0218/index.htm.

[34] *Unidecode 0.04.17 - ASCII transliterations of Unicode text*. [Online; accessed 21-April-2015]. 2015. URL: https://pypi.python.org/pypi/Unidecode.

[35] Princeton University. *About WordNet*. [Online; 28-April-2015]. 2010. URL: http://wordnet.princeton.edu.

[36] *WikiMedia: Database dumps – Index of /enwiki/*. [Online; accessed 22-January-2015]. 2015. URL: http://dumps.wikimedia.org/enwiki/.

[52] Wikipedia. *Wikipedia:Database download — Wikipedia, The Free Encyclopedia*. [Online; accessed 6-February-2015]. 2015. URL: http://en.wikipedia.org/wiki/Wikipedia:Database_download.

Wikipedia References

[20] MediaWiki. *Manual:Categorylinks table — MediaWiki, The Free Wiki Engine*. [Online; accessed 9-April-2015]. 2015. URL: http://www.mediawiki.org/w/index.php?title=Manual:Categorylinks_table&oldid=1515638.

[21] MediaWiki. *Manual:Langlinks table — MediaWiki, The Free Wiki Engine*. [Online; accessed 21-April-2015]. 2015. URL: http://www.mediawiki.org/wiki/Manual:Langlinks_table.

[37] Wikipedia. *(85476) 1997 MY — Wikipedia, The Free Encyclopedia*. [Online; accessed 1-April-2015]. 2013. URL: http://en.wikipedia.org/w/index.php?title=(85476)_1997_MY&oldid=546254895.

[38] Wikipedia. *Alexander Hughes — Wikipedia, The Free Encyclopedia*. [Online; accessed 10-February-2015]. 2015. URL: http://en.wikipedia.org/wiki/Alexander_Hughes.

[39] Wikipedia. *Categorization — Wikipedia, The Free Encyclopedia*. [Online; accessed 7-April-2015]. 2015. URL: http://en.wikipedia.org/wiki/Categorization.

[40] Wikipedia. *Category:All stub articles — Wikipedia, The Free Encyclopedia*. [Online; 25-June-2015]. 2015. URL: https://en.wikipedia.org/wiki/Category:All_stub_articles?from=Sa.

[41] Wikipedia. *Category:Hidden categories — Wikipedia, The Free Encyclopedia*. [Online; accessed 6-February-2015]. 2015. URL: http://en.wikipedia.org/wiki/Category:Hidden_categories.

[42] Wikipedia. *Cicero — Wikipedia, The Free Encyclopedia*. [Online; 1-April-2015]. 2015. URL: http://en.wikipedia.org/wiki/Cicero.

[43] Wikipedia. *Cicero, Illinois — Wikipedia, The Free Encyclopedia*. [Online; 1-April-2015]. 2015. URL: http://en.wikipedia.org/wiki/Cicero,_Illinois.

[44] Wikipedia. *Classification — Wikipedia, The Free Encyclopedia.* [Online; accessed 7-April-2015]. 2015. URL: http://en.wikipedia.org/wiki/Statistical_classification.

[45] Wikipedia. *David Sharpe (actor) — Wikipedia, The Free Encyclopedia.* [Online; accessed 21-April-2015]. 2015. URL: http://en.wikipedia.org/wiki/David_Sharpe_%28actor%29.

[46] Wikipedia. *David Sharpe (athlete) — Wikipedia, The Free Encyclopedia.* [Online; accessed 21-April-2015]. 2015. URL: http://en.wikipedia.org/wiki/David_Sharpe_%28athlete%29.

[47] Wikipedia. *Help:Using talk pages — Wikipedia, The Free Encyclopedia.* [Online; 27-April-2015]. 2015. URL: http://en.wikipedia.org/wiki/Help:Using_talk_pages.

[48] Wikipedia. *Ole-Johan Dahl — Wikipedia, The Free Encyclopedia.* [Online; accessed 3-March-2015]. 2015. URL: http://en.wikipedia.org/w/index.php?title=Ole-Johan_Dahl&oldid=643887733.

[49] Wikipedia. *Statistikk — Wikipedia,* [Online; 27-April-2015]. 2015. URL: http://no.wikipedia.org/wiki/Spesial:Statistikk.

[50] Wikipedia. *URL redirection — Wikipedia, The Free Encyclopedia.* [Online; accessed 14-April-2015]. 2015. URL: http://en.wikipedia.org/w/index.php?title=URL_redirection&oldid=656225198.

[51] Wikipedia. *Wikipedia — Wikipedia, The Free Encyclopedia.* [Online; accessed 7-April-2015]. 2015. URL: http://en.wikipedia.org/w/index.php?title=Wikipedia&oldid=655221826.

[53] Wikipedia. *Wikipedia:Namespace — Wikipedia, The Free Encyclopedia.* [Online; 27-April-2015]. 2015. URL: http://en.wikipedia.org/wiki/Wikipedia:Namespace.

[54] Wikipedia. *Wikipedia:Navnerom Wikipedia — Wikipedia,* [Online; 27-April-2015]. 2015. URL: http://no.wikipedia.org/wiki/Wikipedia:Navnerom.

[55] Wikipedia. *Wikipedia:Redirect — Wikipedia, The Free Encyclopedia.* [Online; accessed 6-February-2015]. 2015. URL: http://en.wikipedia.org/wiki/Wikipedia:Redirect.

[56] Wikipedia. *Wikipedia:Special:Category tree — Wikipedia, The Free Encyclopedia.* [Online; accessed 6-February-2015]. 2015. URL: http://en.wikipedia.org/wiki/Special:CategoryTree.

[57] Wikipedia. *Word-sense disambiguation — Wikipedia, The Free Encyclopedia.* [Online; accessed 20-April-2015]. 2015. URL: http://en.wikipedia.org/wiki/Word-sense_disambiguation.

[58] Wiktionary. *Category:1000 English Basic Words — Wiktionary, The Free Dictionary.* [Online; accessed 1-April-2015]. 2015. URL: http://en.wiktionary.org/wiki/Category:1000_English_basic_words.