NTNU
Det skapende universitet

Generering av en navnedatabase for kinesiske tegn i japanske navn med utgangspunkt i Wikipedia

Lars Alexander Holdaas

Master i datateknologi
Innlevert: januar 2016
Hovedveileder: Rune Sætre, IDI
Medveileder: Björn Gambäck, IDI

Norges teknisk-naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

Creating a database of pronunciations of Chinese characters in Japanese names using Wikipedia

Lars Holdaas

Supervisors:
Rune Sætre
Björn Gambäck

Norwegian University of Science and Technology
Department of Computer and Information Science

Abstract

Finding the correct reading of a Japanese name written with Chinese characters is a recurring problem for both foreigners and Japanese natives. In Japanese names, Chinese characters can often be pronounced in various ways, and new variations are continuously developed.

Although dictionaries exist that contain readings of all Chinese characters used in Japan, this Master thesis focuses on building a database system that contains Chinese characters used in names and their reading by extracting information from the Japanese Wikipedia. This database will be built for linguists and researchers who study Japanese names in particular, and is meant as a proof-of-concept for utilizing unconventional data sources as input for automatically creating large-scale name dictionaries.

The thesis describes the steps necessary to develop the system. As such, a large part of the thesis focuses on using simple natural language processing methods to extract certain information from the Japanese Wikipedia, and will function as a proof of concept that such a database can be created using publicly available information.

The system developed managed to create two SQLite databases. The first one contained information about names and readings, the second one contained information about persons. The system was not entire accurate meaning some of the data might in fact not be about persons but rather about companies or fictional characters.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Research Questions . . . 2
- 1.3 Objective . . . 2
- 1.4 Research Approach . . . 3
- 1.5 Rationale for Choosing Wikipedia . . . 4
- 1.6 Thesis Structure . . . 5

2 Related Works **7**
- 2.1 Wikipedia as Data . . . 7
- 2.2 Wikipedia and Entity Recognition . . . 8
- 2.3 Japanese-Language Content Extraction . . . 10

3 Background **12**
- 3.1 Wikipedia Article Types . . . 12
- 3.2 Kanji . . . 13
- 3.3 F-Score Testing . . . 16
- 3.4 Tokenization . . . 18
  - 3.4.1 Tokenization of Japanese . . . 19
  - 3.4.2 Kuromoji . . . 19
- 3.5 Bag of Words . . . 21

4 Method **23**
- 4.1 Data . . . 23
- 4.2 Training Set . . . 25
- 4.3 Test Set . . . 26

5 Implementation **28**
5.1 Pipeline Architecture 28
5.2 Simple Article Analysis 29
5.2.1 Title Analysis 30
5.2.2 Body Analyzer 31
5.3 Token Analyzer 32
5.3.1 Final Token Analysis 32
5.3.2 Token Sequence Analysis 32
5.3.3 Token Parenthesis Romaji/Katakana Analysis 33
5.3.4 Token Work Title Analysis 33
5.4 Bag of Tokens 33
5.5 Database Extraction 35

6 Results **37**
6.1 Bag of Tokens training 38
6.2 Bag of Tokens Testing 39
6.3 Isolated Parts Testing 41
6.4 Article Recognition System 43
6.5 Database Generation Test Run 46
6.6 Database Generation Final Run 47

7 Discussion **48**
7.1 Bag of Tokens 48
7.2 Person Classification Issues 49
7.3 Overall Discussion 49

8 Conclusion **51**
8.1 Review of Research Questions 51
8.2 Future Work 53
8.2.1 Detection of Persons in Article Texts 53
8.2.2 Contextual Information About Name Readings 53
8.2.3 Companies/Locations Databases 54
8.3 Final Words 54

A Accessing the Generated Data **55**

B User Manual **56**

C Arguments list **59**

D File Answer Key Notation **60**

E Stopword List Samples **61**

List of Tables

3.1 Explaining F-score (algorithm assessment) 17
3.2 Explaining Kuromoji output 20

4.1 Relevant data for training set 1 25
4.2 Relevant data for training set 2 25
4.3 Relevant data for training set 3 26
4.4 Relevant data for test set 26

6.1 Most commonly occurring person tags 38
6.2 Most commonly occurring nonperson tags 39
6.3 Bag of Tokens initial testing 39
6.4 Bag of Tokens high threshold testing 40
6.5 Bag of Tokens strict exclusiveness 40
6.6 Title Analysis Isolated Testing Results 42
6.7 Body Analysis Isolated Testing Results 42
6.8 Token Analysis Isolated Testing Results 42
6.9 Baseline results for test set 1 44
6.10 Test set run 1 44
6.11 Test set run 2 45
6.12 Test set run 3 45
6.13 Test set run 4 46

C.1 Valid System Arguments 59

E.1 Sample from titlekeywords.txt 61
E.2 Sample from titlesuffixkeywords.txt 62
E.3 Sample from finaltokenkeywords.txt 62

List of Figures

2.1 Cucerzan Database Visualisation (based on figure in text) . . 9

3.1 The two Kanjis used in the Japanese word "Kanji" 14
3.2 Onyonmi and Kunyomi example from the Jisho dictionary . . 16

5.1 System Pipeline Architecture 29

Abbreviations & Vocabulary

The following is a short list of vocabulary and abbreviations frequently occurring in this text that are not necessarily commonly understood.

Kanji Pictograms of Chinese origins used in the Japanese language

Kanji Reading One pronunciation of a Kanji

Kunyomi A native Japanese reading of a Kanji

Onyomi An imported Chinese reading of a Kanji

Hiragana One of two phonetic character sets used to write Japanese

Katakana The other of two phonetic character sets used to write Japanese

Romaji The word for the roman alphabet in the context of it being used to write Japanese words

NLP Natural Language Processing; A branch of computer science that focuses on developing tools and methods for extracting computer-readable data from natural human language

POS Tag Part-of-Speech Tag; A method of tagging words in a sentence with grammatical significance

Precision In this context, the proportion of articles labelled as articles about persons that correctly were about persons

Recall In this context, the proportion of articles about persons that were correctly labelled as such

F-score The harmonic mean of Precision and Recall

Bag of Words An NLP-algorithm for analyzing texts based on the frequency of words that occurs in them

Chapter 1

Introduction

This chapter introduces the project and thesis idea. The chapter consists of the following sections: Section 1.1 introduces the motivation for doing this thesis; Section 1.2 introduces the research questions for the thesis; Section 1.3 introduces the objective of the research; Section 1.4 introduces the research approach of the thesis; Section 1.5 discusses why Wikipedia was chosen to be used for data extraction; Section 1.6 gives a short overview of the structure of the thesis.

1.1 Motivation

Readings of Japanese names that use Kanji is a challenge for both Japanese and non-Japanese. Name dictionaries are frequently used by natives to determine both regular and irregular ways of reading certain combinations of Kanjis that make up a name. This thesis focuses on building a database for names to be used by name dictionaries by analyzing and extracting information from a large user-created encyclopedia, namely the Japanese Wikipedia.

The online encyclopedia Wikipedia[1] has in the recent years become more in focus for various research due to the vast data available and the relative uniformity of how it is presented, despite issues of inconsistent styling and formatting. Although recent research often focuses on using the English

[1] https://en.wikipedia.org/

Wikipedia to improve efficiency of already established NLP techniques (see chapter 2), this thesis will focus on practical applications of using Wikipedia for creating a functional system. The thesis uses specifically the Japanese Wikipedia[2] as input data.

Finally, the finished result will hopefully be useful for the following:

- Linguists that want to trace pronunciations of common Kanjis through various time periods and locations.
- Name research that focus on the context of certain names
- Computer science research focusing on using Wikipedia in a similar way to construct knowledge systems.
- General users that want to look up different names and find associated readings.

1.2 Research Questions

RQ1 To which degree can the Japanese Wikipedia be suitable as the basis for building knowledge databases such as the one proposed?

RQ2 To which degree are relatively simple natural language processing techniques sufficient for extracting the data required for such a database?

RQ3 Will the database generated have an acceptable amount of data when compared to similar systems that contain information on Japanese names?

1.3 Objective

The objectives of the thesis are the following:

1. Create a set of natural language processing algorithms that can sufficiently separate Wikipedia articles on Japanese persons from articles on on either non-Japanese persons or non-persons.

[2] https://ja.wikipedia.org/

2. From these articles create a set of algorithms that can extract information regarding these persons, namely both their given and family names written in Kanji and Hiragana, and their year of birth.

3. Use this data to populate a database for the readings of Japanese names and make the data available to the public through the use of the Internet.

1.4 Research Approach

The thesis uses the following research approach:

1. Prototyping the suggested architecture and software to prove the concept as feasible.

2. Test article analyser accuracy and compare to manually prepared samples (answer key), improving on the methods that work and abandoning methods that do not give positive results.

3. Use results from manual testing as basis for further planning of additional algorithms in order to improve accuracy.

Prototyping Prototyping is a research method in the field of software engineering. The idea is to first build an initial version of the software required, and when finished, analyse which part of the prototype worked successfully and which did not, and then build the next prototype, improving on what was successful and eliminating what did not work as expected (Bischofberger and Pomberger 2012). Although there is dispute within the software community whether a prototype should consist of a simpler version of the entire planned program or purely the core concept, the focus is nonetheless on rapid development and proof-of-concept, rather than usability and eliminating bugs.

Prototyping is used both in industry and research. Within the field of research it is especially used for determining and demonstrating the feasibility of a proposed software system (Gregg, Kulkarni, and Vinzé 2001) (proof-of-concept). This thesis focuses on using the prototype research method by creating a working prototype of the proposed software, and proving both its

feasibility and potential usefulness.

1.5 Rationale for Choosing Wikipedia

When starting the research, one of the central choices was to choose the data source.The Japanese Wikipedia was chosen, although it has both positive and negative sides for research of this type.

Benefits of using Japanese Wikipedia as a data source

1. Wikipedia data is easily accessible as a downloadable XML-file available through Wikimedia, whereas other sources of data might either require writing a web-crawler or in the case of physical dictionaries manually inputting data.

2. Wikipedia has a strict enforcement of the principle that articles in Japanese should state both a person's name written in Kanji and in Hiragana, meaning that all articles on persons would indeed contain the data desired.

3. Wikipedia is in a constant state of growth, meaning that as time passes, more data will be available. From 2008 until 2016, the Japanese Wikipedia has doubled in size according to the English. Wikipedia[3]

Limitations of using Japanese Wikipedia as a data source

1. Wikipedia editors are usually non-professionals, and especially the Japanese Wikipedia has more anonymous editors than any other major language Wikipedia[3], meaning that editors to a large degree are not held accountable for any inaccurate content provided.

2. Although the articles to a large degree are uniform, a significant amount of articles deviate from standards sufficiently that natural language methods may not be as efficient as with a completely standardized copula, and regular expressions need to be complicated to be able to parse the texts.

[3]https://en.wikipedia.org/wiki/Japanese_Wikipedia

3. The XML-dumps that WikiMedia supply do not tag the types of the Japanese Wikipedia articles, meaning developers using the dumps as data would have to correctly identify relevant and irrelevant types before performing methods on the data. More on Wikipedia article types in Chapter 3.1.

Alternatives that exist and might be used for either this or similar research includes but were ruled out in favor of Japanese Wikipedia:

GeoNames Japan[4] – A Japanese website containing data on geographical locations. This site was not chosen as data source due to the data being exclusively about geographical locations and not containing any information on persons and names

DBPedia[5] – A website dedicated to making Wikipedia articles available in databases. This site's content is identical to that of Wikipedia and therefore would be more of a choice concerning workflow. It was determined that the XML files supplied by WikiMedia were sufficient, especially given that WikiExtractor allows the stripping of unnecessary metadata that DBPedia has excess amounts of

1.6 Thesis Structure

The thesis uses the following structure:

1. **Introduction** – Defines the task, objectives, and the planned approach to achieve stated objectives.

2. **Related Works** – A brief description of related earlier works and how they are relevant for this paper.

3. **Background** – Explains the background of the thesis, both from a technical and a linguistic point of view.

[4]http://geonames.jp

[5]http://ja.dbpedia.org/

4. **Method** – Describes the method and scientific principles of the research as well as information on the data and how it was used.

5. **Implementation** – Describes in detail how certain methods and principles were implemented.

6. **Results** – Describes and explains results obtained throughout the research, including both efficiency tests for separate parts of the system as well as the entire system.

7. **Discussion** – Discussions concerning the system, the results, shortcomings during research and recommendations for future work based on this paper.

8. **Conclusion** – A short conclusion based on the work done and what it achieved compared to the goals of the thesis.

Chapter 2

Related Works

This chapter lists and describes related works sorted by category. Some of these works are older and more fundamental in nature, while others are state-of-the-art articles on recent development within relevant fields. As this thesis combines several fields, certain texts might only be relevant to very specific parts of this thesis. The chapter consists of the following sections: Section 2.1 discusses earlier work using Wikipedia as input data for various software research; Section 2.2 discusses the often-occuring problem of entity recognition when dealing with Wikipedia, a problem this thesis also deals with; Section 2.3 discusses works done on extracting content from text sources written in Japanese.

2.1 Wikipedia as Data

As the system uses Wikipedia as input data, the first and most basic related work is that which proves Wikipedia as a functioning and solid source of valid data for any real world application. Wikipedia as a quality source of information comparable to more traditional encyclopedias has already been established (Giles 2005). The question that remains is whether or not Wikipedia is suitable as data for software, or more specifically, for natural language processing tasks.

Zesch, Gurevych, and Mühlhäuser 2007 points to some of the strengths

and weaknesses of Wikipedia as a source of data for computer systems. Wikipedia's strength lies mainly in its vast size and the degree to which it is continuously updated and corrected by its users. Wikipedia also has links to articles about topics relevant to any given article, is capable of accepting numerous surface forms for finding correct articles, and can present users disambiguation pages in the cases a surface form can refer to several topics.

Wikipedia's main weakness lies in its inconsistent editing by its users, and can lead to entity articles being referred to by numerous different surface forms even within Wikipedia itself, as well as articles seemingly similar in content varying significantly in article layouts.

Zesch, Gurevych and Mühlhäuser concludes that Wikipedia is suitable as a data source, due to the vast amount of knowledge available, but that due to its structure will require some effort from researchers to become directly useful as data in NLP research.

2.2 Wikipedia and Entity Recognition

A major part of the system focuses on recognizing the topics of entity articles extracted from the Japanese Wikipedia. The following articles focus on using Wikipedia as data for entity recognition tasks.

Cucerzan 2007 focuses on using Wikipedia article data to solve the problem of accurately determine the subject of an ambiguous reference in a written text. An example named in the article is that news media would often use the word *China* to refer to the government of the People's Republic of China, whereas searching for *China* in Wikipedia would by default forward the user to an article about Chinese culture and history. Similarly, the text mentions the difficulty of understand the reference *the Alliance for Democracy in Mali* as a single entity, and not a group of three separate entities in sequence.

The method included using a a double data base; one that contained all entities related to a surface form (the example in the text being the word *Columbia*, having *Columbia University*, *Columbia Space Shuttle* and the cities of *Columbia, California* and *Columbia, Maryland* etc. as related entities). The second database consists of the entities with related tags (in

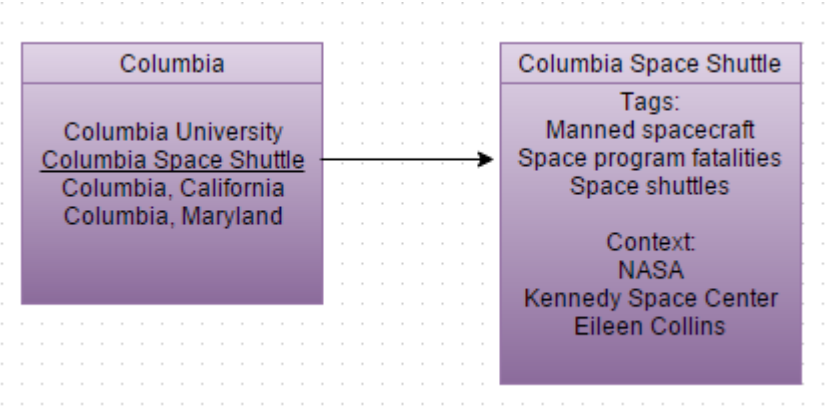

Figure 2.1: Cucerzan Database Visualisation (based on figure in text)

practice short descriptions) and contexts (in which the related entity would be mentioned). The text gives the entity *Columbia Space Shuttle* as an example with the tags being *Manned spacecraft*, *Space program fatalities* and *Space shuttles*, and the contexts being *NASA*, *Kennedy Space Center* and *Eileen Collins*.

Since the task at hand focused on disambiguating texts, the baseline was determined by the following procedure: Search Wikipedia for the baseform given; if an entity whose name matches exactly the baseform is found, use that; if an exact match is unavailable, use the most commonly used entity within the list of ambiguities.

The baseline was determined at 51.7% while Cucerzan shows that training on Wikipedia data allowed her system to achieve an accuracy of 91.4%, clearly indicating that Wikipedia is suitable as data for training NLP disambiguation systems.

Kazama and Torisawa 2007 focused on using the first sentence of the English Wikipedia to extract information about entities and using this to categorize entity articles. Similarly to Cucerzan, Kazama and Torisawa also use the Wikipedia entity article titles as entity names. They then used the word succeeding the copula (after a set of stopwords that would not be used) as the category for the article. The text exemplifies this with the article about Franz Fischler which has the following first sentence:

Franz Fischler (born September 23, 1946) is an Austrian politician.

In this case, Kazama and Torisawa's system extracts the words Franz Fischler and politician (extracting the first word succeeding the copula, while ignoring the stopword *an* and any word starting with a capital letter (in this case, *Austrian*) putting the entity *Franz Fischler* into the category *politician*.

The method of using Wikipedia to train their NLP system did show an increase in the F-score of the final model's named entity recognition, by 3.03 points to an 88.02 F-score, and that the somewhat structured form of Wikipedia articles make it appropriate as input data for relatively simple NLP algorithm training.

The conclusion does, however, point out that future works would have to resolve problems with ambiguity, one of the reasons being that the percentage of ambigious surface forms is steadily rising. Problems with surface form ambiguity seems to be a recurring theme for work related to data extraction from Wikipedia.

2.3 Japanese-Language Content Extraction

Somewhat similar to the subject of this thesis, Kanada 1999 describes a developed system that scans articles in the Japanese World Encyclopedia for (more or less ambiguous) geographical locations, matches these to an internal list, and returning to the user a list of accurate geographical locations described in the text. The key feature is therefore accurately pinpointing somewhat inaccurate references to geographical locations in the article texts.

The most similar part of the systems describes in the paper and this thesis is the system of detecting location information in a text: By either looking at Kanji-suffixes that clearly indicate that the word is a geographical location (such as the Kanji-suffix 県, meaning "prefecture") and a dictionary list of commonly occurring geographical locations (such as アメリカ, meaning either America or the United States depending on the context). A crucial difference is that the system described in the paper uses these methods throughout the articles analysed while the system described in this thesis uses these two methods only for title analysis. The purpose is also different: Kanada wants to include only geographical locations in his work while this system completely excludes geographical entity articles from the database.

Kanada achieved a worst-case precision of about 95% during testing. Kanada stressed the importance of clearing ambiguities, a recurring theme for NLP-related work that uses encyclopedia-based data, and argues that the worst-case results were lower than others due to unresolved ambiguity problems.

Chapter 3

Background

This chapter describes background information crucial for understanding either the principles or the implementation of the system. The chapter consists of the following sections: 3.1 describes the different categories of Wikipedia articles; 3.2 explains how Kanji functions and why the pronunciation of Kanjis present a challenge; 3.3 explains F-score testing which was used to determine the accuracy of the system developed; 3.4 describes the NLP-task tokenization; 3.5 explains the common NLP-algorithm Bag of Words.

3.1 Wikipedia Article Types

As a part of detecting whether Wikipedia articles are relevant or not for the database, it is not only important to classify the category of the content of the article, but also the category of the article itself. Cucerzan 2007 operates with four defined types of Wikipedia pages that this thesis also uses. The four types are:

1. Entity Pages - According to Cucerzan 2007, "An entity page is an article that contains information focused on one single entity, such as a person, a place, or a work of art."
2. Redirecting Pages - For entities with several surface forms, redirecting

pages are set up to ensure that only a single article exists on a single subject and that Wikipedia users that use alternative surface forms to find an entity article will be redirected to the correct page. In other words, redirecting pages are simply meant for references to the correct entity page for its subject matter.

3. Disambiguation Pages - Several entities may share a surface form. In this case, a disambiguation page is used to refer to various entity pages that share a surface form, along with a short description of the topics of each article. For instance, searching for *Norge* on the English Wikipedia will direct the user to a disambiguation page refering to articles on the country Norway, the Norwegian TV-channel TVNorge and the passenger ship SS Norge, among other topics.

4. List Pages - Contains lists of articles concerned with certain topics. A regularly recurring example is the Year List: The article title would be simply a year and then lists various articles about events and incidents that took place during that year.

This paper chooses to use the same system for categorizing articles.

3.2 Kanji

A Kanji is a single non-phonetic symbol meant to represent a concept. While Japanese does indeed have phonemes (symbols that carry phonetic information) as well, namely the Hiragana and Katakana scripts, knowledge of Kanji is absolutely critical to Japanese literacy. Kanjis are a diverset set of symbols, and while some are partly phonetic and others can be considered pictograms, most Kanji symbols are neither. Instead, they represent various abstract and non-abstract concepts, ranging from physical objects as the sun or water to more man-made concepts as law and theories.

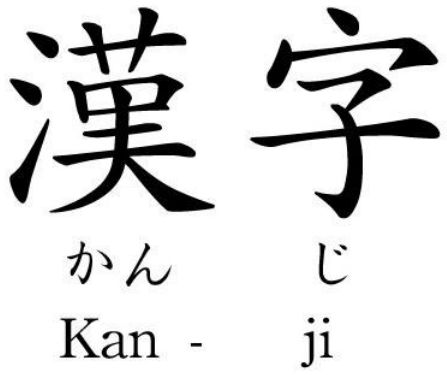

Figure 3.1: The two Kanjis used in the Japanese word "Kanji"

The easiest way to explain the concept of Kanji would be to consider the use of Arabic numerals in western languages. The symbols themselves contain no phonetic information of how to pronounce them, but contain important conceptual meanings. Anyone familiar with Arabic numerals will understand that the symbol "2" implies the value of two, regardless of the pronunciation of the word for "two" in their respective language. Since Arabic numerals are used throughout Europe, the number symbols are readily understood by speakers of various European languages. Conversely, Europeans who do not share a common language will not have any verbal way of communicating the same concept. In this case, people of different languages may communicate directly through the written numerals to convey any value precisely, since the numerals themselves contain only meaning, not pronunciation.

The usefulness of Arabic numerals stretches beyond cross-language communication. Consider mathematics, and reading numbers such as "three-thousand two-hundred and four plus one-thousand five-hundred and eighty-three" vs "3204 + 1583". Numerals are effective symbols for using in mathematics, and it has been implied that mathematics using numerals and other contemporary mathematical symbols could by itself be considered a separate special language independent of traditional languages (Adams 2003).

Kanjis work similarly. Kanji also has symbols for the concepts of numbers. Starting with the simple 一, 二 and 三, meaning respectively one, two and three, numeric Kanjis are widely understood immediately by people in East Asia. However, where the use of symbols are rather limited in western

languages, Kanjis take the philosophy of representing concepts with symbols much, much farther than merely mathematics.

Kunyomi and Onyomi Historically, the Japanese language had no writing system until the ancient Chinese civilization started introducing it sometime in the 5th century AD (Frellesvig 2010). Japanese scholars that mastered the Chinese characters would also learn the Chinese readings, effectively importing Chinese words into the Japanese language. However, native Japanese words for most of the imported vocabulary already existed, meaning acquisition of Chinese loanwords was at first done by the scholars and the elite (not entirely unlike Latin's position in Renaissance Europe (Ong 1959)).

As the Chinese characters became more and more widespread throughout feudal Japan, the concept of "Kunyomi" and "Onyomi" developed. **Kunyomi** means the Japanese reading of a Kanji, based on the native Japanese word for the concept the character represents, while **Onyomi** is the imported pronunciation based on the Chinese language (it must be remarked, however, that the Onyomi pronounciation in actuality often differs significantly from the actual Chinese pronounciation of a certain character).

Since the introduction of Kanji, Japan has gone through several phases of internal fragmentation and external interaction with various parts of East Asia. Sailors from the southern islands of Kyushu would interact with people from Canton (modern day Guangdong province in China) and the Ryukyu Kingdom (modern day Okinawa province in Japan), while traders from the north of Japan would interact with people from Manchuria (northern parts of modern day China) and Koryo (modern day Korea). All these people would use Chinese characters for communicating in writing, but with different words correlating to the same characters. Due to this variation, the Japanese language acquired a wide variety of Onyomi for each Kanji, and the Japanese language has today a much greater variation in pronounciation for each character than either the Korean language or any Chinese dialect.

In addition, dialect differences between for instance the northern part and the south-western part of Japan are sufficiently complex that native speakers of either dialect will be unable to understand common words of the other

dialect (Gottlieb 2005). Thus, Japanese is a language with tremendous variation considering the ways words and Kanji are pronounced.

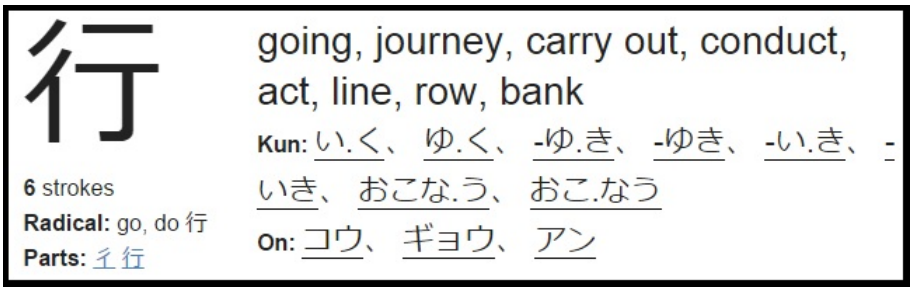

Figure 3.2: Onyonmi and Kunyomi example from the Jisho dictionary

Figure 3.2 shows the entry for the Kanji 行 in the online dictionary Jisho[1]. A total of 11 reading entries exemplifies the relative complexity of knowing how to pronounce this single character. Ignoring grammatical derivatives, the Kanji has 6 unique readings, which according to standard, contemporary romanization of Japanese characters can be written as: iku, yuku, okonau, kou, gyou and an.

Learning the readings of Kanji is a complex linguistic challenge, to the point that both psychologists and neuroscientists frequently use Kanji acquisition as the basis for studies focusing on human learning (Yamazaki et al. 1997) (Sakurai et al. 2000). Kanjis used for names often have the most variety in the number of readings, and is therefore one of the most complicated challenges learners of Japanese face.

3.3 F-Score Testing

For evaluating the accuracy of the system, F-score testing (Goutte and Gaussier 2005) was applied. F-score testing is a standard testing procedure within natural language processing, and functions as an indicator of how well the program functions.

[1] http://jisho.org

	Algorithm	Solution	Assessment
True Positives	*in*	*in*	**correct**
False Positives	*in*	*out*	*incorrect*
True Negatives	*out*	*out*	**correct**
False Negatives	*out*	*in*	*incorrect*

Table 3.1: Explaining F-score (algorithm assessment)

Simply put, F-score requires the developer to categorize a test set into two categories: in or out. These categories function as the solution that the algorithm ideally should achieve in case the program is perfectly accurate.

Running the algorithm and comparing each decision it made to the solution set, decisions are labeled as either:

Since True Positives and True Negatives indicate that the algorithm is making decisions that match the solution set, it is therefore intuitively correct that maximizing the number of True Positives and True Negatives, and conversely, minimizing the number of False Positives and False Negatives, is desirable.

The next step is to find the Precision and Recall of the system. Precision is found by the following formula:

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

Similarly, Recall is found using the following formula:

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives}$$

There are several types of F-score that depend on whether Recall or Precision should be considered more desirable for the system being developed (Rijsbergen 1979). In our case, we use the standard F1, indicating an equal preference for both Recall or Precision. For finding the F1-score, the following formula is used:

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

The F1 score will be a number between 0 and 1, with 1 being the best case and 0 the worst case. Since F1 is a harmonic mean, it means manipulation of either Recall or Precision on the cost of the other will not necessarily improve the score.

3.4 Tokenization

A central technique in natural language processing is the tokenization of sentences. Tokenization, also known as lexical analysis, has the job of dividing a sentence into meaning-bearing separate words, or tokens. In the case of English, a central technique is segmenting by whitespaces since English words are often (but not always) enclosed by spaces when used in a sentence. English is therefore the most common subject language of natural language processing, and its appropriateness for tokenization has been pointed out to create an illusion of the simplicity of tokenization techniques for other languages (Webster and Kit 1992).

However, current focus of research lies beyond the scope of tokenizing English. While some research focuses on multilanguage tokenization (Fagan et al. 1991), an increasing amount of research focuses on tokenization of non-English languages, including Arabic (Attia 2007)(Habash and Rambow 2005) and East Asian languages such as Chinese (Huang et al. 2007).

3.4.1 Tokenization of Japanese

Compared to English, Japanese language is relatively difficult to tokenize due to the lack of whitespaces to separate linguistic entities (words). For reference, the same difficulty is found in natural language processing of other germanic languages such as Norwegian and German where compound nouns are written without any whitespace between the compound words. Japanese is on the rather extreme end, where whitespaces are normally only inserted after punctuation symbols.

3.4.2 Kuromoji

For tokenizing Japanese sentences Kuromoji[2] was used, developed by Tokyo-based Atilika[3]. Atilika is a search technology company founded by Christian Moen, a graduate from the University of Oslo, which focuses on natural language processing techniques for use in information retrieval, with the Japanese language as their main focus.

Kuromoji was chosen as it is not only very precise in the separation and tokenization of Japanese words, but also includes syntactic information about the words, such as classifying whether the word is a noun or a verb etc. A typical Kuromoji output, in this case for the word 物理学 (physics), would look like this:

物理　名詞,一般,*,*,*,*,物理,ブツリ,ブツリ

学　　名詞,接尾,一般,*,*,*,学,ガク,ガク

Let's take a closer look at the above example. In this specific case, we see that the three-character word is split into two tokens. The first token means physics itself while the second token roughly translated to "the study of", or "science". We see that the token 物理 has two part of speech tags, namely 名詞 ("noun") and 一般 ("common" or "normal"). We can conclude that this token represents a noun that is commonly used in modern Japanese. The

[2]https://github.com/atilika/kuromoji

[3]http://www.atilika.com/en/products/kuromoji.html

Position#	1	2	3	4
Type	Token	POS Tag 1	POS Tag 2	POS Tag 3
Example	学	名詞	接尾	*
Position#	5	6	7	
Type	POS Tag 4	Conjugation Type	Conjugation Form	
Example	*	*	*	
Position#	8	9	10	
Type	Base Form	Reading	Pronunciation	
Example	学	ガク	ガク	

Table 3.2: Explaining Kuromoji output

third and fourth part of speech tags are unused and the token has neither a conjugation type nor form. The third to last field is the dictionary form, which is identical to the token itself. In the last two fields we see that the reading and pronuncations are identical: ブツリ ("butsuri").

For the second token (see table 3.2), we see it has three part of speech tags, namely 名詞 ("noun"), 接尾 ("suffix") and 一般 ("common" or "normal"). We can again conclude that this is a commonly used noun, but also that it is usually used as a suffix. The word roughly translates into "the studying of" or "science", and is often put after words to indicate the pursue of knowledge within that field. The Kanji itself directly translates to "learning". Again, we see that there are neither a conjugation form nor type, and that the dictionary form is identical to the token itself. The reading equals its pronounciation: ガク ("gaku").

An attentive reader might notice that included in the token are readings, and then question the necessity of this thesis. The reader should however note that given and family names often use different readings than what would be consider general readings.

Kuromoji is included into the project by using Maven. In the project's pom.xml file, the following lines ensured inclusion of the up-to-date version of Kuromoji:

```
<dependency>
    <groupId>com.atilika.kuromoji</groupId>
    <artifactId>kuromoji-ipadic</artifactId>
    <version>0.9.0</version>
  </dependency>
</dependencies>
```

After this, Kuromoji is able to handle input from the Wikipedia article files (after removing the XML-markup from the files).

3.5 Bag of Words

One of the most common and basic techniques in natural language processing is the Bag of Words technique. Originally a strictly linguistic approach to languages, Bag of Words is not only used in natural language processing but is also utilized in other fields such as image recognition and categorization software (Tirilly, Claveau, and Gros 2008).

Despite the Bag of Words having become a quite common method for analyzing text, the technique is criticised for eliminating important linguistical information such as word order (Wallach 2006). Moschitti and Basili 2004 demonstrated that advanced natural language processing techniques often incur a much higher computational cost with accuracy only barely increasing with algorithms that are much more complex and processor-demanding.

Bag of Words was used as the basis for one of the central token analyzation techniques used in this thesis, using the tokens generated by Kuromoji as input. Since Kuromoji also effectively creates tokens that signify grammatical senses and other valuable information, the method in my system was named "Bag of Tokens" to illustrate that it is in fact tokens that was used as meaning-bearing entities in the system.

The algorithm works by finding the most common tokens occurring in articles about persons and articles about nonpersons in the training set, and giving value to tokens based on the frequency of tokens occurring in either type. The algorithm then analyzes texts using these values and determines whether or not an article is about a person or not depending on what value it finally gives to the text.

Chapter 4

Method

This chapter describes methods and principles on which the research was conducted, as well as the foundation for the system itself. The chapter consists of the following sections: Section 4.1 describes how Wikipedia was used as data for the system; Section 4.2 describes the training sets that were created during development and used for improving the accuracy of the system; Section 4.3 describes the test set that was used for testing.

4.1 Data

There are various ways to fetch data from Wikipedia (Zesch, Gurevych, and Mühlhäuser 2007): Doing both manual and programmatic search on the Wikipedia site itself (the latter is discouraged by the founders of Wikipedia); doing either on a privately hosted Wikipedia; or using an XML dump available from WikiMedia[1]. For this thesis, the XML method was chosen, although it poses another challenge: The dumps are vast in size and impractical for direct use. One solution is the WikiExtractor[2] tool that allows a user to extract articles from a WikiMedia dump into smaller files easily readable for machines.

[1] http://dumps.wikimedia.org/

[2] https://github.com/bwbaugh/wikipedia-extractor

During the development of the software, approximately 1/8th of the entire Japanese Wikipedia per 2nd of June 2015 was used as data. The file used during the development was downloaded from the Wikimedia dump service[3] (close to 1.2 GB uncompressed in XML format and contains 74,939 pages). The compressed XML file was then exported, and WikiExtractor was used to further extract the data into more easily handled files. For the exact procedure, see Appendix B.

Wanting consistency between all parts, an algorithm to find the minimum number of articles in each file was written and found to be 49 (on average, the files contained 198 articles and the file containing the highest number was 393, clearly illustrating a significant gap in the number of articles per file).

In order to keep the articles somewhat random, each training set and testing set was set to contain one article from every file, meaning each set of either type contains 378 articles by semi-random distribution (arguably, it is not random, and a more complicated algorithm for choosing articles might be recommended for any further work within this subject).

Training sets were chosen incrementally from the first, meaning that the first training set contained the first article of every file while the second contained the second article of every file et cetera. Similarly, test sets were chosen from the 49th article of every file, then proceeding in reverse order. The flaw of this method would be that the articles would eventually overlap, meaning that a training set would be identical to a test set, but since the scope of this thesis is rather limited, this was never a realistic issue.

In both sets, articles determined to be about any person were manually marked, but persons concerning non-Japanese had a special tag attached. The reason for this is that non-Japanese were desirable in the automatic training and testing of the Bag of Tokens algorithm, but undesirable in the testing of the entire system. For instance, articles about American individuals would be useful for teaching the Bag of Tokens algorithm to correctly identify person category entities, but would preferably be excluded from the

[3]http://dumps.wikimedia.org/jawiki/20150602/jawiki-20150602-pages-articles1.xml.bz2

rest of the system by other algorithms capable of detecting articles about non-Japanese persons, such as the Title Characterset Analysis algorithm, described in section 5.2.

4.2 Training Set

This section will shortly list the relevant stats for each training set. The tables 4.1, 4.2 and 4.3 use the following acronyms: Not about a person (NP); Not about a Japanese persons (NJP); About a Japanese person (JP).

Training Set 1

	Articles	Persons Total	JP	NJP	NP
Quantity	378	60	34	26	318
Ratio	100%	15.8%	9.00%	6.88%	84.1%

Table 4.1: Relevant data for training set 1

Training Set 2

	Articles	Persons Total	JP	NJP	NP
Quantity	378	62	44	18	316
Ratio	100%	16.4%	11.6%	4.76%	83.6%

Table 4.2: Relevant data for training set 2

Training Set 3

	Articles	Persons Total	JP	NJP	NP
Quantity	378	64	35	29	314
Ratio	100%	16.9%	9.26%	7.67%	83.1%

Table 4.3: Relevant data for training set 3

4.3 Test Set

For the part of the software recognizing relevant articles, a test set of 378 articles was created. Each file produced by Wikiextractor was analyzed to find the number of articles included per file. On average, the files contained 198 articles with the lowest number of articles in a file being 49; the highest 393.

Due to wanting consistency with other parts of the program, article no. 49 was chosen from each of the wikiextractor output files to form the test set. Each article was written to a separate text file, as was the tokenization of each article. Finally, each article was manually checked to determine which of the three category it belonged. See table 4.4

Test Set

	Articles	Persons Total	JP	NJP	NP
Quantity	378	54	37	17	324
Ratio	100%	14.3%	9.79%	4.50 %	85.7%

Table 4.4: Relevant data for test set

After this categorization, the content of the test set was never checked again, as it was meant to be an objective analysis of the efficiency of the article categorization system. Using the content within the test set to improve the system would of course improve the result, but would make the test set obsolete for testing, defeating the purpose of having a separate test set to begin with.

Chapter 5

Implementation

This chapter describes in relative detail the implementation of the system. The chapter consists of the following sections: Section 5.1 describes the overall architecture of the system and the pipeline processing of the input data; Section 5.2 describes the primary algorithms that analyzes the article text for determining the article type; Section 5.3 describes the algorithms that uses the tokenized text; Section 5.4 describes the Bag of Tokens implementation; Section 5.5 describes the final algorithms used for extracting accepted input to SQLite format.

5.1 Pipeline Architecture

The system functions by processing the input data through a pipeline of exclusion methods. Each method tries to detect certain patterns associated with articles that are not considered relevant, and if one is detected, the article currently being evaluated is discarded and removed from the pipeline processing.

An article that passes each test is assumed to be about a person and is then sent to the Person class where regular expressions are used to extract information regarding the person in question and creating a Person object containing that information. This information includes both family and given names written in Kanji and Kana as well as year of birth.

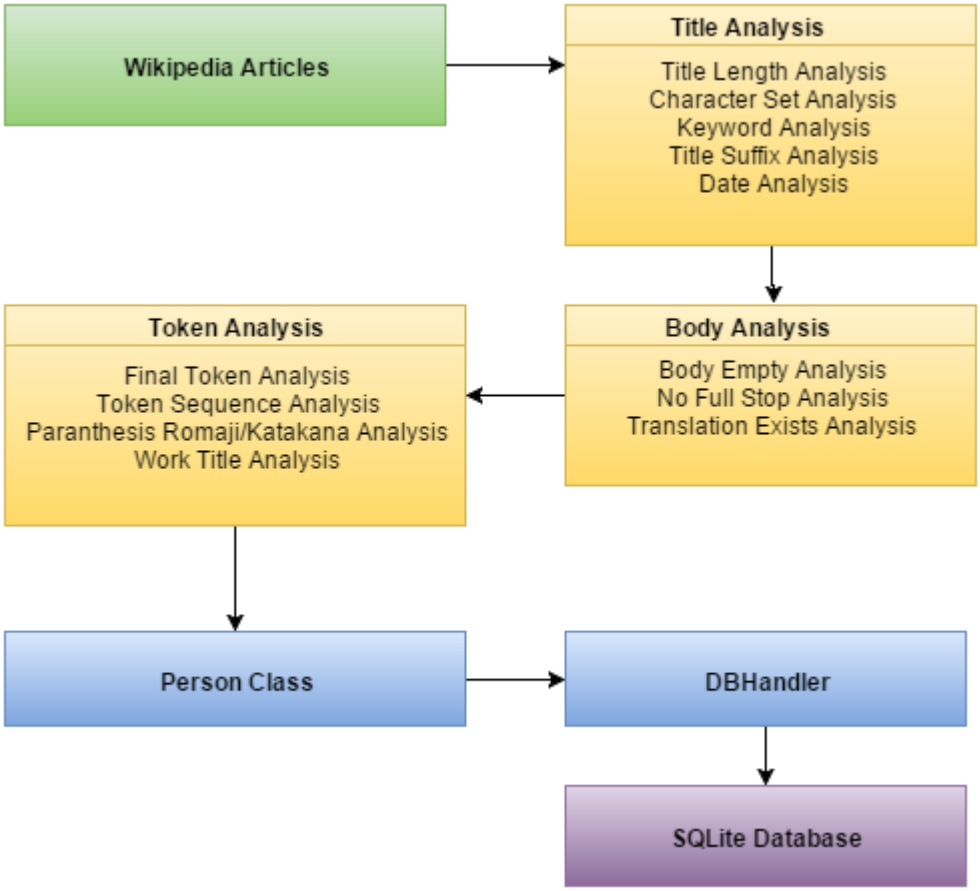

Figure 5.1: System Pipeline Architecture

Finally, all Person objects are sent to the DBHandler that extracts the information from each object and inserts it into SQLite Databases. For a visual representation of the pipeline flow, see Figure 5.1.

5.2 Simple Article Analysis

Since the program developed uses the entire Japanese Wikipedia as its data source, as few steps as possible were used to analyse the data. The first step

of analyzing incoming articles is to go through a series of analysing methods that require a minimum amount of processing power. In the case these methods were not sufficient, more demanding algorithms were used to filter out relevant articles.

The methods used are described in the order in which they are used in the program. However, note that if any article is disqualified at any point by any of the algorithms, it is automatically discarded and is not analysed at any further and more complex steps.

5.2.1 Title Analysis

The analysis of the articles started with evaluations of the title. The following four methods are used to analyse the title of any given article:

1. Title Length Analysis - The first analysis simply checks the length of the title. The minimum length is set to a single character, as no Japanese individuals have a single character as their name (although full names consisting of only two characters do occur semi-regularly). Similarly, a maximum threshold was set at 10 characters, as any title as long as this would either not be a bout a person or be about a person that does not use Kanji in his or her name.

2. Character Set Analysis - Since we are only interested in individuals who use Kanjis in their names, a simple analysis is to check whether the title at all contains Kanji. If it does not, the article is disqualified. However, some individuals write their names partly with Kanji and partly with Kana (Hiragana and/or Katakana). It is desirable for these individuals to also be included in the database, and therefore this method does not simply discard titles that contain non-Kanji characters.

3. Keyword Analysis - A set of words that would never be used in a given name, such as 電車 (train) and 漫画 (manga, Japanese comics). Other types of words that are included are Katakana words that may escape the previous character set analysis, such as コーヒー (coffee) and プログラミング (programming), and words that are only in extreme obscure cases used as a given name such as 大学 (”university”). The

entire list can be found in the titlekeywords.txt file[1] (total of 215 entries).

4. Title Suffix Analysis - A large amount of Kanjis might occur in a persons name, but when used as a suffix will almost never indicate a person. One of the most regular examples is 省, which is often used as the first character of a name, but as a suffix always means "Ministry" (of Defence etc.). The entire list can be found in the titlesuffixkeywords.txt file[2] (total of 19 entries).

5. Title Date Analysis - Certain articles were found to be about either specific years or dates. For this case, a simple algorithm was written to determine whether the title of an article consisted of at least one number and at least one of the three keywords 年 (year), 月 (month) or 日 (day).

5.2.2 Body Analyzer

The second step of the algorithm analysed the article of the body itself. Since the body would contain the entire article, content-wise analysis was left for the token analyser, as only the first sentence would be tokenized and saved per article.

In Wikipedia, group of articles are stored in category articles. These do not contain any actual information but rather links to other articles related to the category subject. Although difficult to find through title analysis, they are relatively simply to find by analysing the bodies.

1. Body Empty Analysis - Category articles often contain no text whatsoever, and any content is found by linking to other articles and fetching their information. Because of this, the output of Wikiextractor would render these articles completely void of content aside from the title. A simple algorithm was therefore written to determine whether or not the article contained any text or not, ruling out articles with empty bodies.

[1] https://github.com/larsdood/masterprosjekt/blob/master/src/titlekeywords.txt

[2] https://github.com/larsdood/masterprosjekt/blob/master/src/titlesuffixkeywords.txt

2. Body No Full Stop Analysis - Some category articles would have a single or a few words in its body, then following with a list of articles relevant to the category subject. Finding that these articles generally did not have a full stop meant that finding them was relatively easy: Scan the text for a full stop, and if none is found, rule out the article.

3. Body Translation Exists Analysis - If an article concerns itself with a subject of foreign origin, whether person or not, there will be a reference to the way the subject matter is written in the original matter. This algorithm searches for such words that implicate that the subject is not Japanese.

5.3 Token Analyzer

The token analyzer uses the tokens generated by Kuromoji as input for determining whether or not the article is relevant for the database. In early experimental phases, the entire article was tokenized before evaluating its content, but it was found that it was more than sufficient to only tokenize and analyse the first sentence of every article.

5.3.1 Final Token Analysis

Articles on the Japanese Wikipedia often end the first sentence in a word that indicates the category of the article. This algorithm checks if the last (or second-to-last word, in the case the article ends in the coppula である) word matches one of recurring article topics. The entire list can be found in the finaltokenkeywords.txt file[3] (total of 57 entries).

5.3.2 Token Sequence Analysis

In Japanese, certain words are only used for inanimate objects, while other words are only used for animate objects. Since the article recognition system works through a system of exclusion, a set of token sequences that will only be used with inanimate objects were tested for by using this algorithm.

[3]https://github.com/larsdood/masterprosjekt/blob/master/src/finaltokenkeywords.txt

The tokens used are the following:

- とは - A grammatical component often used for explaining what an inanimate entity is (short for というは, roughly translated as "the thing called...").
- にある - A sentence ending that implies an inanimate entity exists at a certain location (a sentence describing a living being would use the ending にいる).
- にあった - A sentence ending that implies an inamiate entity historically existed at a certain location (a sentence describing a living being would use the ending にいた).

5.3.3 Token Parenthesis Romaji/Katakana Analysis

Articles concerning non-Japanese whose names are written with Kanji will regularly write their readings in either Romaji or Katakana. This method finds the first parenthesis in a sentence and checks whether or not the next word is written using either Romaji or Katakana. If this is the case, the article is disqualified. This is particularly useful for detecting articles on Korean and Chinese persons.

5.3.4 Token Work Title Analysis

Articles on creative works are often put between the Japanese brackets 『 and 』, similarly to the usage of the character " in English. This method simply checks whether or not the first token of the body text is the character 『. The reason this is done during the token analysis and not during title analysis is that article titles do not use this character for encapsulating the title of the creative work, whereas the actual article texts regularly does this.

5.4 Bag of Tokens

The bag of token methods searches for tokens known to regularly occur in articles that either deal with persons or non-persons. Originally, it was written to analyse the entire article bodies but this proved to be inefficient, with the

method being rewritten to simply analyse the first sentence in each article.

For creating the set of tokens known to associate with either persons or nonpersons, a total of 378 articles was checked and double-checked manually to determine whether or not the content dealt with persons or nonpersons. While most articles could easily be determined as either, there were several borderline cases, such as the following:

Duos/Groups - While duos and groups can certainly not be considered to be equal to persons, a lot of the similar tokens were used to describe these articles as articles concerning people. However, since they will not be used as input to the database, they were chosen as basis for not relevant tokens. Duos and groups are therefore considered to not be about persons.

Fictional Persons - Fictional characters pose a challenge in research related to actual use of names in a language - however, since names used often reflected real names and the considerable similarity in tokens used, fictional characters were decided to be included as relevant input. Fictional characters are therefore considered to be about persons.

Disambiguation Articles about Persons - Though these articles often do have limited content about actual persons who do use Kanjis for names, these articles themselves are only holders of meta-data, linking to the actual entity articles that concern themselves with persons. Due to the limited content of information in these articles, as well as the risk of double or multiple duplicate entries once the data set is sufficiently large, these articles were therefore considered to not be about persons.

The Bag of Tokens was trained using the training sets described in 4.2. In the initial training of the Bag of Tokens, a method to find the most common tokens for both person articles and nonperson articles was written and executed. The results of this trial can be found in 6.1.

Near the end of development it was found that Bag of Tokens did not contribute positively to the F-score of the system, and that the rest of the

system performed better without it. For more information on this see 7.1. Due to this, it was scrapped from the article processing pipeline and further development of the algorithm was discontinued. Arguably, improvement on the code could have led to better results, but this was concidered too time-consuming given the limited scope of this thesis. The source code for the algorithm is still available in the github repository (see Appendix B).

5.5 Database Extraction

Once articles about persons were extracted from Japanese Wikipedia, they were put into Person objects containing five fields: Kanji characters for family name, Hiragana characters for family name, Kanji characters for given name, Hiragana characters for family name and the year the person was born.

In order to extract this information, a series of regular expressions were used. Due to Japanese Wikipedia articles being quite inconsistent in style and format, several regular expressions to be used in series were necessary.

A DBHandler class was created to handle the Person output and put the information into a SQLite database. Two databases were created: PersonBase.db which contains all information about the persons detected by the system, and NameBase.db which contains a list of Kanji names and Hiragana readings. The reason for creating two databases is to allow users to quickly cross-reference any name reading they find in the NameBase by comparing it to PersonBase. It is also possible for the user to search for the person on the Japanese Wikipedia site once this information is available.

The databases are significant in data. Both databases used the entire Japanese Wikipedia as of 2nd of December 2015 as input data. A total of 261,090 unique persons were identified and 65,933 unique names and readings were extracted from these persons. When repeating the experiment, be mindful that both methods require several hours of processing on an average computer.

Finally, the databases were converted to HTML using SQLite Studio[4]. For accessing either the SQLite database files or the HTML pages, please refer to Appendix A.

[4]http://sqlitestudio.pl/

Chapter 6

Results

The results chapter contains results from various system runs during the development of the software. Some of the results, such as the bag of tokens training chapter, describes development results not necessarily relevant for the final system itself, but as data for important design decisions. Nonetheless, as they are as important to the design of the system, they are included in this chapter.

The chapter consists of the following sections: Section 6.1 contains results obtained during training of the Bag of Tokens algorithm; Section 6.2 contains results given by the Bag of Tokens algorithm during isolated testing; Section 6.3 contains the results achieved by testing each single algorithm in isolation (except the Bag of Tokens algorithm); Section 6.4 contains the results of the entire article recognition system when run at different times during development; Section 6.5 contains results from the testing of the entire system including the database generation algorithms when using a subset of Wikipedia; Section 6.6 contains the results of the final execution of the program as a whole when generating the database that was the objective of this thesis.

6.1 Bag of Tokens training

Tables 6.1 and 6.2 indicate the most commonly occurring tokens in person and nonperson categories during the initial training of the Bag of Tokens. The results show that many of the same words were shared across the two categories, somewhat increasing the complexity of using the Bag of Tokens method to categorize articles.

Token	Meaning	Occurrences	Ratio of articles
ある	To exist	20	37.0%
日本	Japan	13	24.1%
年	Year	12	22.2%
する	To do	11	20.4%
県	Prefecture	9	16.7%
日	Day	8	14.8%
月	Month	8	14.8%
から	From	6	11.1%
市	City	6	11.1%
者	Person	5	9.3%

Table 6.1: Most commonly occurring person tags

Token	Meaning	Occurrences	Ratio of articles
ある	To exist	131	40.4%
年	Year	87	26.9%
する	To do	71	21.9%
月	Month	64	19.8%
日	Day	58	17.9%
から	From	48	14.8%
県	Prefecture	46	14.2%
日本	Japan	44	13.6%
こと	Thing	35	10.1%
れる	”Passive”	25	7.7%

Table 6.2: Most commonly occurring nonperson tags

6.2 Bag of Tokens Testing

This subchapter describes testing results of the Bag of Tokens algorithm in isolation from other parts of the system.

The Bag of Tokens algorithm was tested on the test set with initial value threshold set to 0.5 with the results presented in Table 6.3.

True Positives:	34	False Negatives:	5
False Positives:	138	True Negatives:	201
Precision:	19.8%	Recall:	87.2%
F-Score:	32.2%		

Table 6.3: Bag of Tokens initial testing

Although the Recall was certainly satisfactory, the Precision was deemed to be completely insufficient. Experimenting with different threshold values for maximizing the F-score concluded with the ideal threshold set to 1.2 and the results shown in Table 6.4.

True Positives:	24	False Negatives:	15
False Positives:	26	True Negatives:	313
Precision:	48.0%	Recall:	61.5%
F-Score:	53.9%		

Table 6.4: Bag of Tokens high threshold testing

Having the Bag of Tokens-algorithm able to hit an F-score of 0.54 after this initial tests was very positive. After the ideal threshold for the algorithm in isolation was found to be 1.2, experiments with a strict excluding policy were run: Tokens found in both categories were strictly excluded during a single run. This experiment ended with 0 positives, and the threshold value was reset to the original 0.5 for the results shown in Table 6.5

True Positives:	1	False Negatives:	38
False Positives:	7	True Negatives:	332
Precision:	12.5%	Recall:	2.56%
F-Score:	4.25%		

Table 6.5: Bag of Tokens strict exclusiveness

The results clearly show that mutual exclusion for the tokens is ineffective, and the idea was scrapped.

Although only tested for the purpose of experimenting, the logic should nonetheless be obvious: The Bag of Tokens algorithm already corrected for

tokens that are strongly associated with one category and weakly with the other. Thus, an exclusion system is not necessary.

6.3 Isolated Parts Testing

In order to ensure that individual algorithms within the system were in fact efficient by themselves, all the basic algorithms were tested in isolation to see their efficiency. This testing was done after the training of training set 3, and was performed on test set 1.

Note that the default mode for these methods is to accept the article entity to be about a person, therefore the recall will necessarily be high for all algorithms. Precision, on the other hand, was rather low, and it was assumed that while some algorithms are efficient on larger data sets, being able to correctly label a single article as not about a person was considered a success during this testing. On the other hand, incorrectly labelling an article about a person as not about a person was considered undesirable.

The baseline precision, by accepting every article as an article about a person, was found to be 10.3%, with an F-score of 18.7% (see Table 6.9). The entire system at the point of testing had a precision of 62.3%, a recall of 97.4% and an F-score of 76.0% (see Table 6.11). Tables 6.6, 6.7 and 6.8 show the results of respectively Title Analysis algorithms, Body Analysis algorithms and Token Analysis algorithms in isolated test sessions.

Algorithm	Precision	Recall	F-Score
Title Length Analysis	10.8%	100.00%	19.5%
Character Set Analysis	13.2%	100.00%	23.3%
Title Keyword Analysis	15.3%	100.00%	26.5%
Title Suffix Analysis	10.9%	100.00%	19.7%
Title Date Analysis	10.7%	100.00%	19.4%

Table 6.6: Title Analysis Isolated Testing Results

Algorithm	Precision	Recall	F-Score
Empty Body Analysis	10.5%	100.00%	19.0%
No Full Stop Analysis	10.8%	100.00%	19.5%
Translation Analysis	10.8%	100.00%	19.5%

Table 6.7: Body Analysis Isolated Testing Results

Algorithm	Precision	Recall	F-Score
Final Token Analysis	12.9%	100.00%	22.8%
Towa Analysis	12.7%	100.00%	22.5%
Niaru Analysis	11.9%	100.00%	21.2%
Niatta Analysis	10.6%	100.00%	19.2%
Parenthesis Romaji/Katakana Analysis	15.8%	97.4%	27.2%
Work Title Analysis	11.0%	100.00%	19.8%

Table 6.8: Token Analysis Isolated Testing Results

The results were interpreted in the following way:

1. All algorithms had a precision higher than the baseline of 10.3% , mean-

ing every algorithm was able to identify **at least one** article about a person in a set of only 378 articles (see Section 4.3).

2. Most algorithms had a precision barely above the baseline, but when used together in combination had the baseline of 62.3%, indicating that although each algorithm by itself could be considered rather inaccurate, they were able to detect disqualifying patterns in different types of texts.

3. The only algorithm that in this case introduced recall problems was the "Parenthesis Romaji/Katakana Analysis" algorithm. This is most likely due to articles about Korean and Chinese being considered relevant at the time of classifying data. If the experiments were to be repeated, it is crucial that this classification is changed. On the other hand, this same algorithm had the highest precision score and this drop in recall was therefore a tolerable trade-off for a higher precision.

Another experiment with only the "Parenthesis Romaji/Katakana Analysis" algorithm disabled was run, resulting in a precision of 53.4%, a recall of 100.00% and an F-score of 69.6%. An increase of F-score from 69.6% to 76.0% was therefore considered an acceptable trade-off for a slight drop in recall.

6.4 Article Recognition System

During development, tests were run at different times, with different parts of the system discussed in Section 5.2 being implemented at each part. Each paragraph will shortly give both results and which parts of the program were implemented at the time of testing. All tests discussed in this section uses the test set described in Section 4.3.

The baseline of the system was given by accepting all articles as person as the system functions by exclusion methods rather than inclusion methods. The baseline is given by Table 6.9. Results for test sessions 1, 2, 3 and 4 are given respectively by Tables 6.10, 6.11, 6.12 and 6.13. Which algorithms used is mentioned after each table (the baseline uses no algorithms as it simply accepts all articles as relevant).

Baseline

True Positives:	39	False Negatives:	0
False Positives:	339	True Negatives:	0
Precision:	10.3%	Recall:	100.00%
F-score:	18.7%		

Table 6.9: Baseline results for test set 1

Test session 1 (using training set 1)

True Positives:	36	False Negatives:	3
False Positives:	116	True Negatives:	223
Precision:	23.6%	Recall:	92.3%
F-score:	37.6%		

Table 6.10: Test set run 1

Algorithms used: Title length analysis, Characterset analysis, Title stopword analysis & Title suffix stopword analysis

Test session 2 (using training set 1 & 2)

True Positives:	38	False Negatives:	1
False Positives:	40	True Negatives:	299
Precision:	48.7%	Recall:	97.4%
F-score:	64.9%		

Table 6.11: Test set run 2

Algorithms used: All algorithms used in test session 1, in addition to: Title date analysis, Body empty analysis, Body no full stop analysis, Body translation exists analysis, Final token analysis, Token sequence analysis

Test session 3 (using training set 1, 2 & 3)

True Positives:	38	False Negatives:	1
False Positives:	23	True Negatives:	316
Precision:	62.3%	Recall:	97.4%
F-score:	76.0%		

Table 6.12: Test set run 3

Algorithms used: All algorithms used in test session 2, in addition to: Parenthesis Romaji/Katakana analysis, Work title analysis

Test session 4 (using training set 1, 2 & 3)

True Positives:	29	False Negatives:	10
False Positives:	16	True Negatives:	323
Precision:	64.4%	Recall:	74.4%
F-score:	69.0%		

Table 6.13: Test set run 4

Algorithms used: All algorithms used in test session 3, in addition to: Bag of Tokens Analysis

As test session 4 shows, the inclusion of the Bag of Tokens analysis algorithm reduced the accuracy of the system. This finding and its implication on the development of the system is discussed in Section 7.1.

6.5 Database Generation Test Run

After the methods for matching regular expressions and the database handler were implemented, a final test run with the original 1/8th of Wikipedia was run. The resulting SQL file had a total of 13,944 person entries from a total of 74,939 articles. This means that the program identified 18.6% of all articles in this set as articles about Japanese persons. The training and test sets had an average ratio of 9.92% articles about Japanese persons to total number of articles.

Manually going through the entire set of 74,939 articles is a task of such scope it would take months to complete. For this reason, it is not possible to reach a conclusion of the precision of the system. However, if we assume that the the ratio of relevant articles to be 9.92% for the entire set, this would give the system a best-case precision of 53.3%, with the best-case describing a situation with no false negatives. While this is unrealistic, the system has in tests made very few false negative judgements, meaning that this simple speculation is not completely without merit. For comparison, a precision of

62.3% was found during test session 3.

6.6 Database Generation Final Run

The final database generation used the entire Japanese Wikipedia as input. Although the file size was roughly eight times the size of the test set (9.04GB to 1.81GB), it contained 13 times as many articles (992,812 to 74,939). This indicates that the partitions of the entire data available at WikiMedia do not necessarily evenly distribute articles of similar size. From this, it was assumed that article content types as well might be unevenly distributed between the partitions.

During the final run, a total 261,090 unique persons were found. This means that roughly 26.6% of all articles in the final set were identified as articles about Japanese persons. This number is significantly higher than both the test run and the training sessions would indicate, meaning a significant portion of these articles might be mislabeled. However, since the meta-data on the partitions suggest an uneven distribution of articles between files, any conclusion can not be reached without analysing a large portion of the entries in the database.

From these 261,090 persons, 65,933 unique names were extracted. The persons can be found in the database `PersonBase.db`, and the names can be found in the database `NameBase.db`. Both can also be viewed in HTML-format. For accessing these files, please refer to Appendix A.

Chapter 7

Discussion

This chapter discusses the results and its implications, and also discusses certain problems encountered during development and how they can be improved. The chapter consists of the following sections: Section 7.1 discusses the development of Bag of Tokens algorithm and its ultimate failure; Section 7.2 discusses problems of determining which articles should be included and excluded from the article recognition system; Section 7.3 is an overall discussion of the work done, including a short discussion concerning whether or not the system is efficient for populating name databases.

7.1 Bag of Tokens

While Bag of Tokens was originally showing signs of decent performance in isolation, it was shown in Section 6.4 not to improve significantly on precision when used in combination with the other algorithms, instead showing a significant decline in the recall performance of the system. Due to this, it was determined that while Bag of Tokens by itself had merit as a useful algorithm for determining article categories, it contributed negatively to the result when combined with the other, more specialized algorithms. Due to this, bag of tokens was finally deactivated as a part of the processing pipeline for the system.

I believe that if more time was used on extending its training base and

used in a situation of differentiating between several types of categories, it would probably perform better than in this situation. However, given a single binary choice of person or not person and a limited training set, algorithms written specifically to recognize these kind of articles based on the training set were proven to significantly outperform the Bag of Tokens algorithm.

7.2 Person Classification Issues

When the system was first being developed, it was deemed that being able to simply identify articles about persons concerning East-Asian people would be sufficient, and because of this articles about not only Japanese, but also Chinese and Koreans were tagged as positive in both the training sets and the test set, and the algorithms were written to reflect this.

However, at a very late stage of the development, it was found that the "Parenthesis Romaji/Katakana Analysis"-algorithm was efficient at detecting articles concerned with persons of Korean and Chinese identity, even though it was not designed to do so. Since this discovery was made at a very late part of the project development, there was no time to retag the training and test sets. This led to the system results having a slightly higher rate of False Negatives than it would have if qualifiers for a relevant article was correctly set during initial development and the tags were correctly set to reflect this.

It is therefore recommended that any work or experiment that wants to repeat or build upon this thesis is aware of this problem, and makes a good judgement whether or not to intend to include articles about non-Japanese persons.

7.3 Overall Discussion

A lot of the work done focused on creating a system for detecting whether or not an article was about a person or not. About 20% of the development time was used on building a Bag of Tokens algorithm, which by itself performed fine but when used cooperatively with the other algorithms barely

increased precision while drastically lowering recall. In retrospect, if less time was used on developing probability-based NLP-methods and rather on further developing absolute determination methods, the project would have progressed faster and allowed time for building an application for using the data. Another idea could be to focus research purely on building a set of algorithms for classifying text data, which could then later be used for research as proposed by this thesis.

Comparing the system developed to existing name dictionaries built using more traditional methods, a popular online dictionary for names, Onamae Jiten[1], contains 171,455 unique names. This shows that the system, with a total of 65,993 unique names identified, certainly has merit as a very efficient way to populate a database of names.

It must be noted, however, that other name dictionaries contain both names written with and without Kanji while the database generated by this system contains only names that are written using Kanji, meaning that the amount of names are not necessarily directly comparable.

[1] http://name.m3q.jp/

Chapter 8

Conclusion

A system for extracting persons and their respective names from the Japanese Wikipedia was created. Although the Japanese Wikipedia is challenging to use due to inconsistent editing and styling, the significant size of the automatically generated databases for persons and names shows that the Japanese Wikipedia certainly has merit as a data source for NLP tasks that require large amounts of data. Two of the strongest benefits of using Wikipedia as a data source are the readily available WikiMedia dumps that radically simplify the process of obtaining the data in a manageable way, and the amount of research that's already been done on the subject (see Chapter 2).

The chapter consists of the following sections: Section 8.1 reviews the research questions introduced in 1.2; Section 8.2 suggests future work that can use the developed software or improve upon it; Section 8.3 contains the final words of this thesis.

8.1 Review of Research Questions

Refering to Section 1.2 This section reviews to which degree the research questions have been answered.

RQ1 To which degree can the Japanese Wikipedia be suitable as the basis for building knowledge databases such as the one proposed?

Conclusion: The Japanese Wikipedia proved to be a sufficiently suitable data source for building the databases proposed. Problems encountered during development were for the most part related to inconsistent editing and format, but once these obstacles were overcome it was straightforward to extract the massive amounts of data used to populate the database from the Japanese Wikipedia.

RQ2 To which degree are relatively simple natural language processing techniques sufficient for extracting the data required for such a database?

Conclusion: In the final system, only simple natural language processing techniques were used. The results presented in Chapter 6 shows that such algorithms used in succession were more accurate than the Bag of Tokens algorithm, and that the Bag of Tokens algorithm contributed negatively to the accuracy when introduced as a part of the program pipeline. During test runs, the highest F-score obtained was 76.0%, indicating the accuracy of the system was moderate. The results in Section 6.3 indicate that each algorithm worked by eliminating only a small portion of irrelevant articles while ensuring none or close none relevant articles were incorrectly labelled as irrelevant. Although each algorithm by itself scored poorly, when used in succession the algorithms proved to be much more efficient, and that this design principle should be continued for further research.

RQ3 Will the database generated have an acceptable amount of data when compared to similar systems that contain information on Japanese names?

Conclusion: As mentioned earlier in this chapter, a contemporary and popular online dictionary for Japanese names contains 171,455 unique names. Comparing this to 65,993 unique names detected by this system shows that it was able to find roughly 2/5 as many names as an established name dictionary. However, unless the database is scrutinized it is difficult to determine how many of the 65,993 names detected are in fact names. On the other hand, as Wikipedia grows along with its data, this system can be used to automatically generate larger databases without any other work being necessary. The data generated is of an acceptable size, with concerns being during

initial development that the amount of names detected would range in the hundreds.

8.2 Future Work

Further development and improving of the system is necessary to improve its accuracy. Finding persons mentioned in texts that do not have their own Wikipedia entries can also significantly improve the results, although that is quite a challenging task. This, along with the natural user-driven expansion of the Japanese Wikipedia, suggests that the system in the future might become an efficient way to generate name Kanji dictionaries. This section will not present specific future work based on this system.

8.2.1 Detection of Persons in Article Texts

In the Japanese Wikipedia article texts, a large amount of persons are mentioned that do not have their own articles. Writing a system for detecting these and extracting their information could radically improve the size of the data to be used for populating the databases. This is, however, a more complicated task than simply identifying whether a given article is about a person or not. If such research is to be undertaken, one has to carefully consider the pros of having a larger data set to work with versus the cons of decreasing the accuracy of the system in total.

8.2.2 Contextual Information About Name Readings

One of the original ideas for the implemented system was to include information such as year of birth and birthplace to be able to track certain uses of a name in given contexts. Readings of name Kanji often vary by region and epoch, and having a system that could further contextualize the use of names would be useful for linguists studying origins of or the dynamicity of Kanji readings.

Taking this idea a step further, it would be possible to extract other information regarding the person, such as profession, to determine whether certain names were associated with certain social status etc. Such a system could be useful for research in the field of linguistic anthropology.

8.2.3 Companies/Locations Databases

Although this research focuses on names of people, many of the methods can be altered to recognize either companies or geographical locations rather than persons. Given the work that has already been finished, a database containing all companies in a certain region listed on Wikipedia, along with founding date and other information readily available on Wikipedia, should be a natural and progressing step forward from this research.

8.3 Final Words

The research was able to answer the research questions raised and is therefore considered successful, although the system itself certainly needs to be improved upon if it is to be used for populating name dictionaries. Learning Japanese has been a passionate hobby for many years, and having the chance to write a thesis in Computer Science concerning it has been very inspiring, although challenging.

Thanks to my supervisors Rune Sætre and Björn Gambäck for all assistance and a great deal of patience. Thanks to Christian Moen and Gaute Lambertsen of Atilika for inspiration for coming up with the thesis idea. Thanks to my father for helping with writing academic English and educated opinions regarding the format of the thesis itself.

Appendix A

Accessing the Generated Data

In total, two database files was generated: `NameBase.db` is the final result and contains names and readings in Kanji and Hiragana. `PersonBase.db` contains names and readings in Kanji and Hiragana as well as year of birth for all persons detected. Both files are available at the project github repository:
`https://github.com/larsdood/masterprosjekt`

The databases are located in the folder `databases\`.

In addition, the databases are available in HTML format at:
`http://folk.ntnu.no/larsalex/NameBase.html`
`http://folk.ntnu.no/larsalex/PersonBase.html`

The HTML files can also be downloaded for local viewing at github:
`https://github.com/larsdood/masterprosjekt/tree/master/htmls`

The HTML-files are relatively big (8.42 MB and 55.1 MB) and it is recommended to download and use the database files is possible. The HTML-pages were generated using SQLiteStudio[1].

[1] http://sqlitestudio.pl/

Appendix B

User Manual

The entire program source is available through github: `https://github.com/larsdood/masterprosjekt`. In addition to obtaining the system from github, the following steps are necessary to use the system:

Input data set-up

1. Download a partition of or the entire Japanese Wikipedia. Go to `http://dumps.wikimedia.org/jawiki/` and choose the latest dump. Relevant files are written on the format `jawiki-YYYYMMDD-pages-articles#.xml.bz2`, where `YYYYMMDD` is the date and `#` is either the number of the partition or null (null in the case of the file that contains all articles).
2. Extract the XML-file using WinRAR or similar software.
3. Use WikiExtractor to extract articles from the XML files. WikiExtractor is available from `https://github.com/bwbaugh/wikipedia-extractor` and requires Python to run. Various commands and arguments may be valid, however the following is recommended:

   ```
   WikiExtractor.py -b 1M -o target jawiki-YYYYMMDD-pages-articles#.xml
   ```

WikiExtractor will strip files of metadata and partition the articles into 1MB sized files in various folders in the directory /target/. When extracting, please note that WikiExtractor will use roughly 5-20 minutes to execute

on an average computer, depending on the size of the input. Once extraction is complete, the articles are ready to be used directly as input during development.

Configure config.properties In the project folder, open the config.properties file and make sure the `datafilespath` points to the output directory of WikiExtractor (target). Note that the system uses the `basefolder` property as a pretext for all except the database paths. Should you choose to use this, set all other paths to the desired relative path. If not, set `basefolder` to empty and set all other properties to desired absolute paths. Running the program with `-config reset` as arguments will reset the value to default.

At this point, all methods that do not require a functional database can be run. Some methods require other methods to be run first. For instance, the method for testing F-Score requires first a test set to be generated, and then an `isperson.txt` file to be created. This file is used as an answer key for the F-score testing. For correct notation when creating such a file, see Appendix D.

Database Set-up In order to extract name information to a database, a database has to first be created. The DBHandler in this system uses SQLite as database system.

1. Name Database For creating a name database, follow these steps:

1. Create an SQLite database file (at the location of your choice)
2. Create the table Name with the fields ID (INT and Primary Key), KANJI (TEXT), HIRAGANA (TEXT). All fields should be set to Not NULL.
3. Edit config.properties and change the nameDBPath to the path of the database you've created. Note that unlike the other paths, the database paths in config.properties are ABSOLUTE PATHS.
4. Run the program with the argument `-extractNames`.

2. Person Database For creating a person database, follow these steps:

1. Create an SQLite database file (at the location of your choice).
2. Create the table PERSON with the fields ID (INT and Primary Key), FAMILYKANJI (TEXT), FAMILYHIRAGANA (TEXT), GIVENKANJI (TEXT), GIVENHIRAGANA(TEXT), BORNYEAR(INT). All fields should be set to Not NULL.
3. Edit config.properties and change the personDBPath to the path of the database you've created. Note that unlike the other paths, the database paths in config.properties are ABSOLUTE PATHS.
4. Run the program with the argument `-extractPersons`

Appendix C

Arguments list

The following is a list of valid arguments. Note that the arguments other than extractNames and extractPersons are included solely for supporting any developers wanting to build on this project. The mark * means the algorithm requires predefined answer key files to be genereated (see Section D).

-extractNames	Creates a name database based on the input data
-extractPersons	Creates a person database based on the input data
-genArticleSets #	Generates a set of text files containing article and tokenized texts from 0 to the range #
-genSingleArticleSet #	Generates text files for the #th article per input file
-populateBag	Populates the bag of tokens *
-countArticles	Outputs number of articles per input files
-extractTestSet	Extracts a test set based on the input data
-trainingSetTest	Tests the system accuracy against the training sets *
-fScoreTest	Tests the system accuracy against the test set *
-resetConf	Resets the configuration file to default

Table C.1: Valid System Arguments

Appendix D

File Answer Key Notation

In order to run tests on test sets, a file with the answer key must be written for the system to know which ones should be identified as a person and which should not. However, the process is slightly more complicated. Due to bag of words testing relying on finding ALL persons and some methods searching for East Asians rather than only Japanese, the `isperson.txt` should be created in the following way:

1. If the article is about a person, write the number of the file associated with the article directly into the text file.
2. If this person is a Japanese person, simply proceed to the next line.
3. If the person is not Japanese, add `C` for Chinese, `K` for Koreans and `NJ` for any other nationality.
4. After the last person entry has been added, do not press enter (avoid having an empty line at the bottom).

Save the file in the same folder as the article and tokenized files using the filename `isperson.txt` .

Appendix E

Stopword List Samples

Tables E.1, E.2 & E.3 contain the first ten entries of the stopword lists used in the article recognition system. For more information, see Chapter 5. Each tables shows the first ten entries of each list. Note that the title suffix list contains the entry 者 which means person. This Kanji is used to describe occupations and social standings, not individuals, therefore it is excluded from the list.

Word	Meaning	Word	Meaning
町	Town	市	City
学校	School	問題	Problem
プログラミング	Programming	リスト	List
政治	Politics	事件	Incident
作物	Product	局	Bureau

Table E.1: Sample from titlekeywords.txt

Word	Meaning	Word	Meaning
省	Ministry	式	Ceremony
県	Prefecture	学	Learning
者	Person	州	State
化	Change	賞	Prize
仏	Buddha	団	Group

Table E.2: Sample from titlesuffixkeywords.txt

Word	Meaning	Word	Meaning
こと	Thing	総称	General term
示す	To exemplify	れる	Grammatical Passive
一つ	One (thing)	基	Radical/Base
インターチェンジ	Interchange	小説	Novel
映画	Movie	漫画	Japanese Cartoon

Table E.3: Sample from finaltokenkeywords.txt

Bibliography

[1] W.R. Bischofberger and G. Pomberger. *Prototyping-Oriented Software Development: Concepts and Tools*. Monographs in Computer Science. Springer Berlin Heidelberg, 2012. ISBN: 9783642847608. URL: https://books.google.no/books?id=XMWqCAAAQBAJ.

[2] Dawn G. Gregg, Uday R. Kulkarni, and Ajay S. Vinzé. "Understanding the Philosophical Underpinnings of Software Engineering Research in Information Systems". In: *Information Systems Frontiers* 3.2 (2001), pp. 169–183. ISSN: 1387-3326. DOI: 10.1023/A:1011491322406. URL: http://dx.doi.org/10.1023/A:1011491322406.

[3] Jim Giles. "Internet encyclopaedias go head to head". In: *Nature* (Dec. 2005), pp. 900–901. DOI: 10.1038/438900a.

[4] Torsten Zesch, Iryna Gurevych, and Max Mühlhäuser. "Analyzing and accessing Wikipedia as a lexical semantic resource". In: *Data Structures for Linguistic Resources and Applications* (2007), pp. 197–205.

[5] Silviu Cucerzan. "Large-Scale Named Entity Disambiguation Based on Wikipedia Data". In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (2007), pp. 708–716. URL: http://www.aclweb.org/anthology/D07-1074.

[6] Jun'ichi Kazama and Kentaro Torisawa. "Exploiting Wikipedia as external knowledge for named entity recognition". In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Prague, Czech Republic, 2007, pp. 698–707.

[7] Yasusi Kanada. "A Method of Geographical Name Extraction from Japanese Text for Thematic Geographical Search". In: *Proceedings of the Eighth International Conference on Information and Knowledge Management*. CIKM '99. Kansas City, Missouri, USA: ACM, 1999, pp. 46–54. ISBN: 1-58113-146-1. DOI: 10.1145/319950.323229. URL: http://doi.acm.org/10.1145/319950.323229.

[8] Thomasenia Lott Adams. "Reading Mathematics: More than Words Can Say". In: *The Reading Teacher* 56.8 (2003), pp. 786–795. ISSN: 00340561. URL: http://www.jstor.org/stable/20205297.

[9] Bjarke Frellesvig. *A History of the Japanese Language*. Cambridge, 2010. ISBN: 0521653207.

[10] Walter J. Ong. "Latin Language Study as a Renaissance Puberty Rite". In: *Studies in Philology* 56 (1959), pp. 103–124.

[11] Nanette Gottlieb. *Language and Society in Japan*. Cambridge, 2005.

[12] M. Yamazaki, A. W. Ellis, C. M. Morrison, and M. A. L. Ralph. "Two age of acquisition effects in the reading of Japanese Kanji". In: *British Journal of Psychology* 88 (1997), pp. 407–421.

[13] Yasuhisa Sakurai, Toshimitsu Momose, Makoto Iwata, Yasuhiko Sudo, Kuni Ohtomo, and Ichiro Kanazawa. "Different cortical activity in reading of Kanji words, Kana words and Kana nonwords". In: *Cognitive Brain Research* 9 (2000), pp. 111–115.

[14] Cyril Goutte and Eric Gaussier. "A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation". In: *Advances in Information Retrieval*. Ed. by DavidE. Losada and JuanM. Fernández-Luna. Vol. 3408. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, pp. 345–359. ISBN: 978-3-540-25295-5. DOI: 10.1007/978-3-540-31865-1_25. URL: http://dx.doi.org/10.1007/978-3-540-31865-1_25.

[15] C. J. Van Rijsbergen. *Information Retrieval*. 2nd. Newton, MA, USA: Butterworth-Heinemann, 1979. ISBN: 0408709294.

[16] Jonathan J. Webster and Chunyu Kit. "Tokenization as the Initial Phase in NLP". In: *Proceedings of the 14th conference on Computational linguistics* 4 (1992).

[17] J.L. Fagan, M.D. Gunther, P.D. Over, G. Passon, C.C. Tsao, A. Zamora, and E.M. Zamora. *Method for language-independent text tokenization using a character categorization*. US Patent 4,991,094. Feb. 1991. URL: https://www.google.com/patents/US4991094.

[18] Mohammed A. Attia. "Arabic Tokenization System". In: *Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources*. Semitic '07. Prague, Czech Republic: Association for Computational Linguistics, 2007, pp. 65–72. URL: http://dl.acm.org/citation.cfm?id=1654576.1654588.

[19] Nizar Habash and Owen Rambow. "Arabic Tokenization, Part-of-speech Tagging and Morphological Disambiguation in One Fell Swoop". In: *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. ACL '05. Ann Arbor, Michigan: Association for Computational Linguistics, 2005, pp. 573–580. DOI: 10.3115/1219840.1219911. URL: http://dx.doi.org/10.3115/1219840.1219911.

[20] Chu-Ren Huang, Petr Šimon, Shu-Kai Hsieh, and Laurent Prévot. "Rethinking Chinese Word Segmentation: Tokenization, Character Classification, or Wordbreak Identification". In: *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. ACL '07. Prague, Czech Republic: Association for Computational Linguistics, 2007, pp. 69–72. URL: http://dl.acm.org/citation.cfm?id=1557769.1557791.

[21] Pierre Tirilly, Vincent Claveau, and Patrick Gros. "Language Modeling for Bag-of-visual Words Image Categorization". In: *Proceedings of the 2008 International Conference on Content-based Image and Video Retrieval*. CIVR '08. Niagara Falls, Canada: ACM, 2008, pp. 249–258. ISBN: 978-1-60558-070-8. DOI: 10.1145/1386352.1386388. URL: http://doi.acm.org/10.1145/1386352.1386388.

[22] Hanna M. Wallach. "Topic Modeling: Beyond Bag-of-words". In: *Proceedings of the 23rd International Conference on Machine Learning*. ICML '06. Pittsburgh, Pennsylvania, USA: ACM, 2006, pp. 977–984. ISBN: 1-59593-383-2. DOI: 10.1145/1143844.1143967. URL: http://doi.acm.org/10.1145/1143844.1143967.

[23] Alessandro Moschitti and Roberto Basili. "Complex Linguistic Features for Text Classification: A Comprehensive Study". In: *Advances in Information Retrieval*. Ed. by Sharon McDonald and John Tait. Vol. 2997. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, pp. 181–196. ISBN: 978-3-540-21382-6. DOI: 10.1007/978-3-540-24752-4_14. URL: http://dx.doi.org/10.1007/978-3-540-24752-4_14.