Saarland University
Faculty of Natural Sciences and Technology I
Department of Computer Science

Master’s Thesis

Fusing AIDA Means Table With Google Dictionary

submitted by
Sandra Alshabben

June 15, 2013

Supervisor:
Prof. Dr.-Ing. Gerhard Weikum

Reviewers:
Prof. Dr.-Ing. Gerhard Weikum
Dr. Klaus Berberich
Statement in Lieu of an Oath

I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis.

Furthermore, I hereby confirm the congruence of the contents of the printed data and the electronic version of the thesis.

Sandra Alshabben .................. Saarbrücken, June 15, 2013

Declaration of Consent

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Sandra Alshabben ................. Saarbrücken, June 15, 2013
Acknowledgements

My sincerest thanks go to Prof. Dr.-Ing. Gerhard Weikum, who granted me to discover a for me new research area of Computer Science by writing my master’s thesis in the Databases and Information Systems Group at the Max-Plank Institute for Computer Science in Saarbrücken.

Furthermore, I am grateful to Dr. Klaus Berberich who accepted to be my second reviewer.

In addition, I want to thank my supervisor Mohamed Amir Yousef, who supported me in any situation and gave me advice whenever I needed it.

Finally, I owe my deepest gratitude to my husband who gave me consolation and cheered me up whenever I was down.

Last but not least, it is an honour for me to thank my mother who always makes me smile.
Dedicated to my beloved family.
Contents

1 Introduction ................................................................. 7
  1.1 Motivation .............................................................. 7
  1.2 Related Work .......................................................... 10
  1.3 Proposed Approach .................................................. 11
  1.4 Overview ............................................................. 11

2 Basics of Named Entity Disambiguation ............................... 13
  2.1 Named Entity Disambiguation (NED) .............................. 13
      2.1.1 History and Definition of NED ............................ 13
      2.1.2 Entity Linking with Wikipedia ............................ 14
  2.2 Common Knowledge Bases ......................................... 16
  2.3 NED Methods ......................................................... 17
      2.3.1 Record-Linkage Methods ................................. 17
      2.3.2 Corpus-based Methods .................................. 17
      2.3.3 Collective Learning Methods ......................... 17

3 Introduction to AIDA ..................................................... 19
  3.1 Concept .............................................................. 19
  3.2 Architecture ........................................................ 20
      3.2.1 Detecting Mentions by Using Stanford NER Tagger .... 20
      3.2.2 Collecting and Scoring Candidate Entities ............ 22
      3.2.3 Matching Mentions to Named Entities ............... 25
  3.3 AIDA Means Table and its Drawbacks ......................... 26

4 Google Dictionary and its Fusion with AIDA ....................... 29
  4.1 Introduction to Google Dictionary ............................... 29
      4.1.1 The Starting Point ....................................... 29
      4.1.2 Processing the Data .................................... 30
  4.2 Comparison between AIDA and Google Dictionary .............. 33
      4.2.1 Language Matching ...................................... 34
      4.2.2 Scoring Google Dictionary .............................. 35
  4.3 Filtering Google Dictionary ..................................... 37
      4.3.1 Filtering by Use of Thresholds ....................... 37
Abstract

In today’s life of social media and unlimited internet access, search engines have a special precedence by providing any information needed on any time at any space all around the world. One essential challenge that they have to deal with is natural language understanding since words in search queries can be ambiguous. The process of disambiguating words (mentions) to unique concepts (entities) is known as Named Entity Disambiguation (NED). Many NED systems have been developed that aim to establish mappings between mentions (words) and entities (concepts). One of those systems is AIDA, an online tool for automatically disambiguating mentions in a given text by URL’s of Wikipedia articles. Mappings from mentions to entities are normally stored in so-called means tables. AIDA Means Table contains around 27822 mention-entity mappings but also suffers from drawbacks like inconsistency, high complexity and expensive computational costs.

The aim of this thesis is to explore a new data resource for AIDA that opens possibilities for the enrichment of AIDA Means Table by adding new, accurate and non-noisy data.

Index Terms
1. Introduction

1.1 Motivation

During the early development of the World Wide Web in the 1990th, the Web was designed for scientists who found a need in sharing and accessing data all around the world. At that time, only a few organizations like universities or libraries were interested in the Web [2]. After creating the first easy Web browsers like Mosaic\(^2\) and Netscape Navigator\(^3\) in 1993, the Web became more than a simple academic tool [3]. It became a modern way of communication and knowledge gathering. The Web arouse increasing public interest and with its extremely fast growth, users

---

1. http://www.wordle.net, last opened: 15.05.2013
1. Introduction

were suddenly presented by a huge collection of websites and online documents. They aimed to provide millions of users with any kind of information. But soon, this aim changed into a challenge, caused by the increasing number of websites, the enormously data arise and the local organized structure of the Web.

One of the first approaches to put order into the Web was by using so-called web directories which tried to categorize websites in hierarchically or by topic ordered catalogs [4]. Early catalog systems attempted to create browsable databases that were generated by including all file names of the directories of a website and make searchable indexes from them. The occurring problem at that time was that catalogs had to be manually administrated and hence were limited in the amount of data that they could include [4]. Since the amount of online available data increased with high speed all around the world, there was on the one hand no possibility to organize the Web in its entire structure and on the other hand there was the users’ boosting wish of unlimited information access from the Web. As a reason of fact, there was a need to develop automatic systems that retrieve information from the Web without depending on limited catalogs or any kind of classification.

In order to make exploring the Web more comfortable for the users, the first information retrieval systems for the Web were invented. Their intention was to allow users to search and find any information in the Web by simply typing a text into a field on a website. The text, also called search request or query, could have been anything that the user was looking for, like a person’s name, an address, the translation for a foreign word or the definition of a term.

Such systems are known as today’s search engines. One of the first search engines that provided full text search was WebCrawler⁴, an engine developed in 1994. Two years later, the nowadays famous Google⁵ search engine was invented but became first popular in 2000, after having innovated a new interactive algorithm called PageRank. Their algorithm supposed that high ranked websites are the ones that are more often linked to. With this discovery, Google became able to provide better query results to its users than any other engines at that time and thus opened a new era of search engines [5].

Nowadays, search engines like Google are a normal part of everyday’s life. Google for example answers milliards of search requests every day, always intended to satisfy the users’ need of information. Search engines are today not only the most common information resources, they also replace hardback lexicons and dictionaries. Looking up terms on Wikipedia⁶ is a typical procedure that is done by everybody worldwide to gain as much as knowledge that can be achieved in the smallest possible time span.

Despite of being provided with a huge variety of different full text search engines,

⁵http://www.google.com/about/company/ , last opened: 10.06.2013  
⁶http://en.wikipedia.org/ , last opened: 05.06.2013
1.1. Motivation

finding the right information can still be a difficult task for today’s Web users since searching correctly does not only require a properly working search engine, but might also depend on the ability of the users to type in a query that is understood by the engine. The key issue here is the human language that allows user requests to be vague or incomplete or words to be ambiguous. For that reason, one essential aspect that search engine developers have to deal with is natural language understanding which explores the capability of computer systems to derive the meaning of user inputs. But how can search engines know what the text means that the user inserted?

When analyzing the search behaviour of different users it can be seen that in general the user is searching for a fact that is described by a name. The name itself normally consists of a single word or a sequence of words that can have various meanings. Which of the meanings the user is actually looking for can not easily be known by the search engine. Normally, the engine provides links to websites that contain the search term and are often visited by users or often linked to by other websites. Since modern website developers headline their pages with names that describe the content of the particular site, the first few results that this procedure provides are usually the ones that the user intended to find.

A main problem occurs when search queries get longer, like whole sentences, text passages or complete documents. Then the search engine might not be able to distinguish the context in which a name is used and might provide query results to a wrong topic. Here, a useful aspect of the Web is that most names on a website stand in context with other information on the site which may express the meanings of the name like a person, an institution, an organization, or a geographical information etc. In the field of natural language understanding these meanings are called named entities. Since the goal of good search engine development is to provide optimal query results to the users, the entities belonging to a name as well as the context in which a name occurs have to be known. Therefore, search engines are supported by large databases that contain mappings from the names to their entities. The process of illustrating these mappings is known as named entity disambiguation (NED) [6].

NED is nowadays a research field itself. Lot of NED systems have been proposed to find the best disambiguation method for natural language text. In general the systems obtain their named entity mappings by harvesting named entity pairs from any sites and knowledge bases in the Web like DBpedia7, Freebase8, Wikipedia or YAGO2 [8]. Adding data from external sources into their databases also means to include the noisy data that is part of a source. Hence, one of the main issues that NED systems have to deal with is to obtain robust and clean means tables.

One NED system that was invented by the Max-Plank Institute for Computer Science in Saarbrücken is AIDA [6][7] which includes data from the YAGO2 knowl-

---

7http://dbpedia.org/, last opened: 05.06.2013
8http://www.freebase.com/, last opened: 05.06.2013
1. Introduction

edge base and contains various manually added inputs. Its means table consists of mappings from names to Wikipedia articles that disambiguate them. AIDA aims to provide an efficient and accurate NED method and performs with low error rate. Nevertheless AIDA Means Table has potential for improvements as it suffers from noise that was generated by anchor texts from Wikipedia and contains only a small amount of entries.

In order to increase the number of useful entries in AIDA Means Table, new accurate data could be added to the existing system. The data could be gathered from different knowledge base sources like Freebase for example. In May 2012, Google released a huge data collection of disambiguation mappings [9] which may be considered as possible data resource for AIDA Means Table.

The approach in this thesis tries to enhance the quality of AIDA Means Table by including new and non-noisy data from Google Dictionary.

1.2 Related Work

The main paper that is related to this thesis is the work of Spitkovsky and Chang [9] in which the Google Dictionary is shortly presented. After the release of Google’s dictionary only a small number of research papers has been published in the field of named entity disambiguation that concern the data of the dictionary. A selection of papers that describe the possible usage of the Google Dictionary will be mentioned in the following section.

The first paper that should be named here is the paper of Steiner et al. [10] which uses the Google Dictionary to justify that matching names to entities works well without considering the context in which a name appears. Google Dictionary is a completely context-free collection of mappings between names and entities and Steiner et al. [10] adopt this idea in their approach. Their new method is modeled on the basis of Google’s Knowledge Graph invention and aims to improve the results of Web searches by including conceptual information from microposts in social networks like status updates, likes, or recommends. For this, they developed a new browser extension that links the concepts that were found in a post to the Google Knowledge Graph entities for that an user searched for on Google.

Another interesting paper that refers to the Google Dictionary is a paper by Singh et al. [11] in which an automatically generated cross-document co-reference corpus is introduced. The method used for establishing the corpus is based on generating mappings from anchor texts to Wikipedia links that were taken by a web crawl. This approach is similar to the one used to create the Google Dictionary but according to Singh et al. [11] their approach differs from Google by providing information that gives hints about the context like URLs and offsets.

The last paper that is related to this thesis is a paper by Dalton and Dietz [12] that is concerned about the classification of disambiguation context for entity linkage. Dalton and Dietz [12] claim to present a model that uses relevance feedback for the above mentioned classification. To evaluate their approach, they use data of a Wikipedia dump from Freebase as well as data from the Google Dictionary that
provides external web links.

As far as known, no further approaches have been developed that refer to the Google’s dictionary in the time this thesis has been written.

1.3 Proposed Approach

The aim of this thesis is to improve the performance of AID A by establishing a new data resource for AID A Means Table. Before new entries could be added to the system, AID A Means Table has to be evaluated. Hence, a particular concern of this thesis is to judge the existing system according to its accuracy and coverage as well as the data collection provided by Google. Since Google’s data has not been available in browsable version, the creation of a new database for Google’s dictionary and the calculation of score probabilities that rank each entry are also parts of this thesis. After having checked the entries in the Google Dictionary, it became clear that many of the dictionary’s entries are noisy. For that reason, different filter techniques for cleaning the Google Dictionary had to be applied and were evaluated. As a consequence, AID A Means Table and the filtered Google Dictionary have been merged in experimental setups. Accordingly, an important component of this thesis has been the estimation of the results. For better usability a Web-based browser for searching in the dictionary has been designed as well.

1.4 Overview

In this chapter (1) the motivation for the importance of this thesis is given. As a little recollection, the second chapter (2) will contain a short overview of the most important notations and concepts of named entity disambiguation related to this thesis. The third chapter (3) introduces the AID A system and points out why and how an improvement of its means table should be done. In chapter four (4) the setup of the Google Dictionary and the criteria of filtering the data to distinguish between good and bad entries will be explained. In addition, the fusion of AID A and Google Dictionary as well as the entire results of this thesis will be described. The design of a Web-based browser developed for better usability of the Google Dictionary will be discussed in chapter five (5). Finally, this thesis ends with a conclusion and a short overview of further research options.
2. Basics of Named Entity Disambiguation

In this chapter an overview of the key concepts that are needed to understand the context of this thesis as well as the design and methodology of the systems covered in this thesis is given.

2.1 Named Entity Disambiguation (NED)

2.1.1 History and Definition of NED

Caused by the political and financial increasing globalization around the world, the first interest in understanding the content of message texts arouse in 1987 when extracting information about defense related activities or company activities from unstructured text like newspaper articles was brought into focus by the American defense agency DARPA\(^1\). For this reason the Message Understanding Conferences (MUC) [13] were initialized in which all participating scientists had to present methods to extract specific types of information from texts, like temporal and spatial information or the agents and operations of an event.

In 1991 Lisa F. Rau [14] published the first research paper that described a system for extracting and recognizing company names in a text. With the time, the goal of identifying names in texts became a complete new information extraction task and was defined as the task of Named Entity Recognition (NER). Then in 1995 the Sixth MUC concerned about the NER task took place. The specialty of MUC-6 was not only to train data for six type classes (Person, Location, Organization, Time, Date, Money, Percent) but also to include semantic evaluation about three main tasks. Among them was the Word Sense Disambiguation (WSD) problem which had to find an explanation (sense) for each word in the text [13].

Nowadays a more common task that belongs to NER is Named Entity Disambiguation (NED) which is closely related to WSD but concentrates on finding meanings only for proper names and not for other word groups in the text. For better understanding, the difference between NER and NED is shown in figure 2.1.

The definition of a proper name was set up by the philosopher Saul Kripke in 1981 who assumed that a proper name is a rigid designator that refers to the same

\(^1\)The Defense Advanced Research Projects Agency is responsible for military technology development in the US.
2. Basics of Named Entity Disambiguation

In all possible worlds in which it exists [15]. Hence, a proper name can for example describe a person, an organization or a location.

In the field of NED proper names are also called *mentions*. For the rest of this thesis this notation is used accordingly. Moreover the meanings of the mentions that the NED task tries to discover are denoted as *entities*. The problem with mentions is that they can be ambiguous. The same mention can have more than a single meaning [6][7]. For example the mention "Michael" can stand for "Michael Schumacher" or "Michael Jackson" or "Michael Stonebraker" or a lot of other persons.

Considering these facts Named Entity Disambiguation can be defined as the process of mapping each mention in a text to an unique entity [6][7].

2.1.2 Entity Linking with Wikipedia

Since entities should provide the explanations of mentions, a common practice is to link each entity with the Wikipedia article that contains the entity's name in its title and gives information about the entity. Hence, in NED each Wikipedia article can be seen as an entity.

Wikipedia[^1] is the most popular and biggest multilingual free-content encyclopedia on the Internet. It contains over 30 million articles in 286 languages from which over 4.2 million articles are included in the English Wikipedia version. As a fact
the number of articles in Wikipedia is continuously growing [16].

Each Wikipedia article is uniquely classified by its title. The beginning of the title is always written in capital letters and spaces between words of complex titles are replaced by underscores. For example a typical article title could look like "Firstword_Secondword_Thirdword". In addition, each article is assigned to one or more categories that describe the topic to which the article belongs to. Currently Wikipedia includes around 60 thousand categories. Another important feature of Wikipedia is its entity linking structure which is created by anchor links contained in articles. Every author of an article can link words in his article to other articles that have already been written to explain the word. That means that many Wikipedia articles contain mentions of entities that already refer to the corresponding entity [17].

Accordingly, Wikipedia can be expressed as a many-to-many correspondence which means that it establishes relationships between its articles, so that one article can relate to many articles and many articles can relate back to one article or other articles [18]. In other words there are different mentions that refer to the same article on the one hand and there are equal mentions that refer to different articles on the other hand. Wikipedia handles this by using two relations: redirects and disambiguation pages[17].

For each alternatively written name that defines the same mention a redirect page to the entity article exists in Wikipedia. For example the article of the computer scientist "Donald_Ervin_Knuth" is redirected to the article "Donald_Knuth" because the notation "Donald Knuth" is more widely spread than the full name and means the same entity. In contrast, when the name of the title is ambiguous, the end of the article's title is further concatenated with an attribute in parenthesis that describes its disambiguation. For example the name "Bush" is contained in Wikipedia as "Bush_(brand)", as "Bush_(band)" and as "Bush_(1916_automobile)". Wikipedia lists all articles with the same name but different meaning in so-called disambiguation pages.

Figure 2.2 illustrates the different relationships between mentions and entities in redirects and disambiguation pages.

What NED wants instead is to obtain a many-one correspondence between mentions and entities in a text. That is, various mentions can refer to the same entity but not vice versa. For modern NED systems the use of Wikipedia features is one of the favoured methods to gain information about entities for their disambiguation processes.

Other repositories that can be used for disambiguation purposes are shortly illustrated in the next section.

\[^2\text{http://en.wikipedia.org/} , \text{last opened: 05.06.2013}\]
2. Basics of Named Entity Disambiguation

Figure 2.2.: Sets of Redirects and Disambiguation Pages [1]

2.2 Common Knowledge Bases

Nowadays various resources besides Wikipedia are known that deliver information about textual contents. Among them are WordNet, DBPedia, Freebase and GeoNames.

Wordnet\(^3\) is a lexical database that orders English nouns, verbs, adjectives and adverbs into groups of words that have a similar meaning, so-called synsets. Each synset represents a distinct concept and contains a definition as well as sentences that describe the use of the words contained in the particular synset. In addition, various semantic relations like synonymy, antonyms, hypernymy and more are drawn between them. In total, WordNet contains about 117,000 synsets and aims to provide a combination of thesaurus and dictionary that may be helpful for disambiguation tasks [19].

DBpedia\(^4\) is a huge knowledge collection that leverages common databases like Wikipedia. It aims to extract structured information from Wikipedia and provides them in the Web for users all around the world. DBpedia is available in various languages. Currently, its English version presents around 3.77 million things. Among them 764,000 persons, 573,000 places, 333,000 creative works, 192,000 organizations, 202,000 species and 5,500 diseases [20].

Freebase\(^5\) is an online database that is built from different resources and aims to provide a fast and globally accessible repository of structured data that is administrated in a graph-like structure. Freebase actually contains more than 37 million topics, 1,998 types, and 30,000 properties about real-world entities like people, places and things [21].

\(^3\)http://wordnet.princeton.edu/ , last opened: 05.06.2013
\(^4\)http://dbpedia.org/ , last opened: 05.06.2013
\(^5\)http://www.freebase.com/ , last opened: 05.06.2013
2.3 NED Methods

GeoNames\textsuperscript{6} is a free-available geographical database that provides spatial information about all countries of the world as well as 8 million names of places. In total, GeoNames contains around 10,000,000 geographical names that were harvested from different sources in the Web like Wikipedia and DBpedia \cite{22}.

Next to many other knowledge bases the above mentioned are frequently used in modern named entity disambiguation methods.

2.3 NED Methods

Various methods have been proposed that aim to solve the disambiguation problem. In this section the most popular NED methods are shortly presented.

2.3.1 Record-Linkage Methods

Record-Linkage is a NED approach that deals with the determination of equivalent records from different repositories that refer to the same entity. It is mainly used on unstructured data sets that are joint together from diverse data sources. The method tries to find similarities among the attributes of the data in order to allow annotations and relations between schematically unequal entries or to eliminate duplicates. For this it uses similarity measures like n-gram overlap or string edit distances \cite{7}. Notable papers that make use of record-linkage are among others the works of Cohen \cite{22}, Limaye et al. \cite{23} as well as the work of Naumann and Herschel \cite{24}. More recent research has adapted record-linkage to entity linkage from Wikipedia.

2.3.2 Corpus-based Methods

Another type of NED methods are corpus-based methods. They are based on harvesting entities from huge knowledge bases like FreeBase or DBpedia that provide a large number of additional entity features like semantic types of entities or relations among entities. Moreover knowledge bases like WordNet offer dictionaries or lexicons of synonyms which allow easy classification of entities. Their only drawback is that they contain such a big amount of data that the running time of corpus-based NED methods prone to be computationally expensive \cite{7}. Nevertheless, a lot of current NED systems use this method for disambiguation.

2.3.3 Collective Learning Methods

The last methods that should be mentioned here, are collective learning methods that transform the disambiguation problem into a stochastic graph problem where mentions and entities are defined as the nodes of a graph and their relations are denoted as edges. Their goal is to establish a joint mapping from mentions to

\textsuperscript{6}http://www.geonames.org/, last opened: 05.06.2013
entities in one single step. One famous approach that uses collective learning is the work of Kulkarni et al. that yields a mention-entity mapping by solving a linear program. Like the corpus-based methods the collective learning methods also suffer from expensive running times caused by the time-consuming graph modeling or linear programming[7].

In the next chapter a more efficient NED system is presented that performs a collective graph-based mapping for disambiguation in real-time, the AIDA system.
3. Introduction to AIDA

After the discussion of the most important concepts of named entity disambiguation, this chapter will introduce the disambiguation system AIDA which was invented by the Max-Planck Institute of Computer Science in Saarbrücken. In the following the system, its concept and architecture as well as the system’s drawbacks will be described in details.

3.1 Concept

AIDA is a system that was developed to obtain information from natural inputs on the Web. In particular it was designed to identify and disambiguate named entities on raw textual data. The main goal of AIDA is to provide an efficient and precisely in real-time working disambiguation tool for online applications which maps each mention in a given text to a single entity that is presented by a Wikipedia link [6][7].

When given an input text like

"Michael was the father of Ingres and Postgres, two relational database systems developed at Berkeley."

AIDA’s system should provide disambiguations for all mentions occurring in the text. For the example text above, AIDA returns the following disambiguation [26] (marked in [ ] brackets before the mention):

"[Michael Stonebraker]Michael was the father of [Ingres (database)]Ingres and [PostgreSQL]Postgres, two relational database systems developed at [University of California, Berkeley]Berkeley."

To obtain this result, AIDA works in three main steps. The first step is to detect the mentions of an input. The input can be any natural text, a HTML or XML formatted text or a table. The mentions on the text are classified by using a named entity recognition tagger. In the example above, the mentions to detect are Michael, Ingres, Postgres and Berkeley. The second step is to collect candidate entities by harvesting possible entities from the YAGO2 knowledge base. Naming only a few entities, possible candidates for the mention Michael in the example could be the racing driver Michael Schumacher or the singer Michael Jackson or the database specialist Michael Stonebraker. The third step is then to assign each
mention to the right candidate entity that fits to the context of the mention. For this a graph based approach is applied that uses prior probabilities, similarity measures as well as coherence to rank the candidate entities and to find the best entity among all candidates. Since the context of the above example includes databases like Ingres and Postgres, the best candidate is chosen to be Michael Stonebraker.

The following draft 3.1 shows the general process of AIDA’s named entity disambiguation method.

![Diagram](image)

Figure 3.1.: The three main steps of AIDA’s disambiguation process [1]

The details of each step will be explained in the next section.

3.2 Architecture

3.2.1 Detecting Mentions by Using Stanford NER Tagger

The first step of AIDA’s disambiguation process is to identify all mentions in the given text. For this task several pre-implmented methods could be used. Most methods that try to extract information about the names of things in a text are based on statistical hidden state sequence algorithms like Hidden Markov Models, Conditional Chain Models or Conditional Random Field Models. These models have in common that they use the Markov property to infer about a state [27]. Each state is associated with a category name that maps to some word in a text.
3.2. Architecture

By tracking the sequence of states the whole text can be generated and each word in the text will be labeled by its category. The problem here is that this method depends on a local structure that makes inferences about a state only in a small window around the state.

Since natural language is more complex and a word can depend on a wider range of words, the Stanford Natural Language Processing Group developed a non-local method that aims to provide a more consistent name labeling suited for NER tasks. Their so-called Stanford Named Entity Recognition Tagger [28] intends to mark names in a text by using a Conditional Random Field Model that includes constraints about long distances and consistency. For approximate inference they use Gibbs sampling which assures the method to be tractable and not locally dependent at the same time [27]. They trained their model by using different training data and provide different category classes. Included are a model trained for MUC containing 7 classes (Time, Location, Organization, Person, Money, Percent, Date), a model trained for CoNLL 2003 [29] containing 4 classes (Location, Person, Organization, Misc) and a model combined by these two models that contains 3 classes (Location, Person, Organization). AIDA uses the Stanford NER Tagger version with the CoNLL training data for automatically detecting the mentions in a given text [6][7].

Figure 3.2 shows a screen shot of the Stanford NER Tagger Demo which gives a little impression how the tagger should work for the example chosen in this chapter.

![Stanford Named Entity Tagger](image)

Figure 3.2.: Mention labeling produced by Stanford NER Tagger Demo [30]
As can be seen, the tagger labels the words Michael, Ingres, Postgres and Berkeley correctly as the mentions of the text.

3.2.2 Collecting and Scoring Candidate Entities

For disambiguation the detected mentions, adequate candidate entities have to be found and ranked according to their relevance in matching to the particular mention. This section is concerned about how possible candidates can be found and how they are compared with another.

Candidate Collection

Entities can usually be extracted from knowledge bases that store not only the entities themselves but also the type of the entity, their properties and the relationships among them. Such knowledge bases are called ontologies. Common ones are for example knowledge bases like Freebase, DBpedia or YAGO2.

YAGO2 [8][31][32] is a huge ontology that was built by deriving information from Wikipedia, WordNet and GeoNames. Each entity contained in YAGO2 represents an article in Wikipedia. In addition, type information about an entity are extracted from Wikipedia and YAGO2 links them to WordNet for obtaining the most common disambiguation of the entity. Relations among entities are manually defined and can be expanded to facts by extracting information from infoboxes and categories of Wikipedia. Each fact is a triple consisting of an entity, a relation and a second entity. Besides YAGO2 uses the GeoNames database to get advanced temporal and spatial information about entities additionally to the ones harvested from Wikipedia. Currently around 9.8 million entities with 447 million are contained in YAGO2 [8][31][32].

Since YAGO2 also includes a relation that expresses the meaning of an entity, facts that contain this so-called MEANS relation can be used for disambiguation purposes [8][31][32]. Accordingly, AIDA harvests candidate entities from YAGO2 that map to the mentions contained in the input text and makes use of YAGO2’s MEANS relation to obtain possible meanings.

Candidate Scoring

Although the MEANS relation in YAGO2 provides a huge set of meanings, each entity can be connected to more than a single meaning. To choose the right entity that fits to the context of the input, AIDA computes different features that judge the usefulness of each candidate entity in the occurring context. This section gives an overview of the features and their calculations.
• The Prior Probability

One efficient feature that AIDA uses to score candidate entities is the prior probability of an entity. The prior probability captures the prominence or popularity of an entity and is estimated by counting the number of occurrences of a mention in link anchor texts that point to the particular entity for which the probability is calculated [6]. In this way AIDA gets for each mention a probability distribution over all possible entity matches.

Looking again at the example in this chapter and using only the prior probability as scoring criteria, the following disambiguation would be provided by AIDA:

"[Michael (archangel)]Michael was the father of [Jean Auguste Dominique Ingres]Ingres and [PostgreSQL]Postgres, two relational database systems developed at [Berkeley, California]Berkeley."

It can be seen that all mentions have been wrongly classified except for the mention "Postgres". This is caused by the priors which take the most frequent entity as the most likely disambiguation without considering the entity's context. In the above example "Michael" refers to "Michael (archangel)" in 13% of all occurrences, in 78% "Ingres" refers to "Jean Auguste Dominique Ingres", in 64% "Berkeley" refers to "Berkeley, California" and in 100% "Postgres" refers to "PostgreSQL" [26].

Hence, popularity priors do not assure correct disambiguation. For this reason there is a need for AIDA to use further context-based measures that may improve the results.

• The Mention-Entity-Similarity

In order to enable a mapping from mentions to entities, the context of the mention and the context of the entity have to be conform with each other.

As context of the mention all words in the text are considered. Only the mention itself and stopwords are not part of the mention’s context. Each mention can accordingly be expressed as a set of coexisting words, also called phrases[6].

As context of the entity all phrases in link anchor texts, names of categories, citation titles or external links in the Wikipedia article that the entity refers to are considered. The phrases are harvested from the entity’s Wikipedia article and are also denoted as keyphrases. Each entity can accordingly be expressed as a set of keyphrases in which every word has statistical weights that are computed by using mutual information (MI) between the entity \( e \) and the keyphrase \( w \) [6]. The \( MI \) is defined as:
3. Introduction to AIDA

\[
MI(e, w) = \log \frac{p(e, w)}{p(e)p(w)} [33] 
\]  

(3.1)

where the joint probabilities \( p(e, w), p(e, \bar{w}), p(\bar{e}, w) \) and \( p(\bar{e}, \bar{w}) \) are calculated by counting the number of occurrences of the word \( w \) in the keyphrase set \( KP(e) \) of the entity \( e \) or in a keyphrase set of an entity that links to \( e \) (\( KP(e') \)) [6]. The equation is given by

\[
p(e, w) = \frac{|w \in (KP(e) \cup \bigcup_{e' \in IN_e} KP(e'))|}{N} [6] 
\]  

(3.2)

with \( N \) being the total number of entities.

In the example a possible keyphrase set for the entity "Michael Stonebraker" might be (computer scientist, relational database, Berkeley) whereas for the entity "Michael Schumacher" a set like (racing driver, Formula One, Germany) would fit. Stonebraker’s keyphrases would then have more accordance with the context words of the mention "Micheal" than the one of Schumacher.

For equalizing the context of both, mention and entity, the similarity measure between them can be calculated by accumulating over the set of keyphrases. Since in some input texts only parts of a keyphrase are present, partial matches are considered as well.

For each partial match \( q \) a score can be computed by

\[
score(q) = z \left( \frac{\sum_{w \in \text{cover}(q)} \text{weight}(w)}{\sum_{w \in q} \text{weight}(w)} \right)^2 [6] 
\]  

(3.3)

where the factor \( z \) is calculated as \( z = \frac{\text{no. matching words}}{\text{length of cover}(q)} \), the \( \text{weight}(w) \) is the \( MI \) weight of word \( w \) and the \( \text{cover}(q) \) is a window size that expresses the size of the smallest window that contains the maximal number of keyphrase words [6].

Mathematically seen the total similarity score between a mention \( m \) and an entity \( e \) is then computed as

\[
sim\text{score}(m, e) = \sum_{q \in KP(e)} score(q) [6] 
\]  

(3.4)

where \( KP(e) \) is the set of keyphrases.
3.2. Architecture

- **The Entity-Entity-Coherence**

Another important measure for finding the right candidate entity is the coherence between two entities. It expresses the distance between the entities’ semantic type or the entities’ YAGO2 relations [6]. For example the semantic type of the entity "Michael Stonebraker" (computer scientist) is the same as the type of the entity "Michael J. Franklin". Hence, both can be connected by the coherence measure and are shortlisted as candidates.

In detail the entity-entity coherence is calculated by the number of incoming links that the Wikipedia articles of the entities have in common.

Mathematically it is defined as:

\[
mw_{coh}(e_1, e_2) = 1 - \frac{\log(max(|IN_{e1}|, |IN_{e2}|)) - \log(|IN_{e1} \cap IN_{e2}|)}{\log(|N|) - \log(min(|IN_{e1}|, |IN_{e2}|))}
\]

where \( N \) is the total number of entities and \( IN \) is the number of incoming links for entity \( e_1 \) and \( e_2 \).

Since the content of usual texts only reports about one or a small number of topics, using the coherence measure allows AIDA to shrink the number of possible candidate entities to the ones that have the same context.

3.2.3 Matching Mentions to Named Entities

In order to establish a collective mapping between mentions and entities, AIDA converts the disambiguation task into a graph problem. For this AIDA constructs an undirected graph in which mentions and entities are defined as nodes and the edges are weighted [6]. The features described in the former section are used to compute the edge weights of the disambiguation graph. The graph contains two types of edges.

1. **Mention-Entity Edges**
   
   This kind of edges reflect the relationship between mentions and candidate entities and are labeled with edge weights that capture the similarity between the context of a mention and the context of a candidate entity [6]. In addition the weights can be defined as a combination of the prior probability and the similarity.

1. **Entity-Entity Edges**

   This edge type describes edges between different candidate entities and are weighted by a function that catches the coherence between two entities [6].

After having produced the disambiguation graph, AIDA tries to reduce the graph to a maximal weighted dense subgraph that consists of unique mention-entity
3. Introduction to AIDA

Figure 3.3.: AIDA’s Mention-Entity Graph - the thicker an edge, the higher the weighted degree [1][7]

edges. The density of the subgraph is measured by the minimum weighted degree among its nodes whereas the weighted degree is defined as the total weight of its incident edges. To obtain an optimal disambiguation, the subgraph is required to match each mention to a single entity [6].

AIDA computes the subgraph in two main steps [6]:

1. Detecting the entity node with the minimal weighted degree.

1. Deleting the node and its incident edges if it is not the last remaining candidate for one of the mentions, else disambiguation complete.

The subgraph with the maximal total edge weight yields then the disambiguation result.

Figure 3.3 shows a possible mention-entity graph for the used example in this chapter. All mentions were correctly disambiguated.

3.3 AIDA Means Table and its Drawbacks

AIDA has a repository in which AIDA’s ground truth data is saved. This so-called Means Table contains about 27822 mention-entity mapping in total, from which 6303 mappings are distinct [34]. Most mentions in the Means Table has been manually disambiguated [6]. Hence, AIDA depends on manual input which is on the one hand a possible error source since the disambiguations are subjective to the persons which manually performed the mappings and on the other hand the
3.3. *AIDA Means Table and its Drawbacks*

Number of entries is limited due to it.

Another issue that AIDA Means Table has to deal with is the inconsistency of its data. Mentions can be written in capital letters or in small letters, in singular or plural form, by use of abbreviations or different cases. Thus, there are many different ways to write a mention that has the same meaning. Even typos could occur. Unfortunately, manual inputs are roughly able to cover all variations of a mention. Therefore AIDA suffers from inconsistency. As an example, AIDA contains a number of mentions that refer to the entity "President_of_the_United_States" but does not include a mention for "President of the U.S."

Another drawback is the fact that AIDA contains a lot of useless entries, so-called noise. The noisy entries are mainly generated by anchor texts from Wikipedia that were badly assigned from any users. For example noisy mentions like the following ones map to the entity "Germany": ISO 3166-1:DE, size=100px, empire|28px [26].

A crucial point that makes the exploration of a new data resource for AIDA so important is its complexity and its expensive computationally costs. Having the right candidate sets is thus the basic level on which AIDA’s named entity disambiguation process should base on.

In summary, AIDA has the following drawbacks:

- Dependent on manual input
- Small number of data
- Noisy data
- Inconsistency
- High complexity
- Expensive computation

In consequence of facing these drawbacks a revision of the existing data in AIDA Means Table seems inevitable to improve the quality of AIDA’s disambiguation process. The next chapter introduces a corpus of mention-entity mappings that was newly released by Google and which could be used to enhance AIDA Means Table by adding its data to AIDA.
4. Google Dictionary and its Fusion with AIDA

After having discussed the drawbacks of AIDA Means Table in the last chapter, this chapter deals with the Google Dictionary, a huge data resource for mention-entity matching developed by Google. This chapter contains a description of the dictionary’s data as well as a comparison between AIDA and the dictionary. As an intermediate result, different filter methods are applied to shrink the number of entries that could possibly be added to AIDA Means Table to clean the noisy data from Google.

4.1 Introduction to Google Dictionary

4.1.1 The Starting Point

With the intention to achieve a new data resource for AIDA Means Table, the focus of this thesis has been set to the exploration of a huge corpus built by Google\(^1\) in May 2012 [9]. Looking at the origins of their release, the developers had the vision to create a collection of data that is freely accessable and fully compatible on any system world wide, filled with knowledge from the most popular online encyclopedia on the Web, namely Wikipedia.

Their creation is a cross-lingual dictionary that maps words to concepts. *Words* are represented by natural language text strings and *concepts* by URLs of Wikipedia articles [9]. Both are equivalent to the notations of mentions and entities that were described in the former chapters of this thesis. In the following the dictionary is referred to as *Google Dictionary*.

On the search for new data for AIDA Means Table, Google Dictionary turned out to be a good choice since both share the commitment of using Wikipedia for entity linking. That is to say, each word maps to an uniquely defined Wikipedia article. In this way Google Dictionary is able to disambiguate the meanings of words according to the definition of named entity disambiguation. Hence, Google Dictionary is perfectly suited for AIDA’s purposes. The extent to which Google Dictionary can be used to enrich AIDA Means Table will be discussed in the next sections.

Another point that made Google Dictionary an interesting source is its simplicity.

\(^{1}\)http://www.google.com/about/company/, last opened: 10.06.2013
4. Google Dictionary and its Fusion with AIDA

On the one site it contains millions of word-concept pairs but on the other site it was built by considering only a simple probabilistic measure based on raw counts of anchor hypertext links from Wikipedia articles, a conditional probability. This fact would it make easier to integrate data from Google Dictionary to AIDA in case of fusing both.

In the following the data provided by Google Dictionary will be described in details.

4.1.2 Processing the Data

The Google Dictionary was published as a set of zipped text files that can be downloaded at http://www-nlp.stanford.edu/pubs/crosswikis-data.tar.bz2/. In total seven files were released that cover a bi-directional data collection, one for mapping words to concept and one for mapping concepts back to words, a cross-lingual file for mapping non-English concepts to English ones and other files that are concerned about lexical issues like free-style typing and white-space handling.

To give an overview, all files are shortly described below [9]:

- **dictionary.bz2**: maps words to concepts
  - e.g., maps Germany to en.wikipedia.org/wiki/Germany
- **inv.dict.bz2**: maps concepts back to words
  - e.g., maps en.wikipedia.org/wiki/Germany to Germany
- **cross.map.bz2**: maps non-English to English concepts
  - e.g., de.wikipedia.org/wiki/Riesenkalmar to Giant squid
- **redir.map.bz2**: maps free-style titles to concepts
  - e.g., Bush Baby and Bushbabies to Greater galago
- **redir.log.bz2**: contains trace of all proposed cluster merges
  - i.e., a Wikipedia article can represent a cluster of other Wikipedia articles that have the same meaning, like redirects and disambiguation pages.
- **lnrm.forward.bz2**: maps words w to concepts in l(w) form
  - e.g., Bushbaby (lesser) to bushbaby lesser
- **lnrm.back.bz2**: maps l(w) back to w
  - e.g., bushbaby lesser to Bushbaby (lesser)
- **lnrm.dict.bz2**: maps aggregate l(s) to concepts

In this thesis the intention was delimited to the word-to-concept dictionary whose data is contained in the dictionary.bz2 file and whose mappings are closest to the ones in AIDA. In addition using only one file reduces the amount of data saved on the server. To have a bi-direction dictionary the program code was accordingly adapted to accept queries in both directions. In the following parts the notation of Google Dictionary refers to the word-to-concept dictionary only.

The word-to-concept dictionary was a 2.7 GB big compressed text file that ex-
panded to 20GB of data after unzipping the file. Each line of the dictionary’s text file is a mapping from a word to a concept and consists of four parameters, the word, a conditional probability, the url and a score. Moreover, all lines are alphabetically sorted and have the following line structure [9]:

```
<word><tab><cprob><space><url>[<space><score>]*
```

where the

```
<word>
```

is the natural text string that refers to the `<url>` in the same line. Words can be empty strings and can also contain white-spaces. For each word Google Dictionary may also include different types of punctuation, upper and lower cases as well as plural forms of a word.

They are harvested from four different sources [9]:

1. titles of English Wikipedia articles
2. anchor texts from English inter-Wikipedia links
3. anchor texts from non-Wikipedia pages into the English Wikipedia pages
4. anchor texts from non-Wikipedia pages into non-English Wikipedia pages for topics that have corresponding English Wikipedia articles

The

```
<url>
```

is the Wikipedia article to which the `<word>` refers, thus the disambiguation of the word. In a different way from normal Wikipedia links, the URLs are represented without the prefix "en.wikipedia.org/wiki/". Each URL can define a so-called cluster of Wikipedia articles. For example redirects and disambiguation pages usually belong to a cluster.

The

```
<cprob>
```

is the conditional probability of the `<url>` when given the `<word>`. It captures the ratio between the number of hyperlinks into the Wikipedia article with `<url>` that have the anchor text of the `<word>` and the total number of anchors with the `<word>`.

The following table 4.1 depicts how the conditional probabilities are calculated on an exotic example:

As can be seen, the raw counts to compute the conditional probability are contained in the `<score>` field. For example the probability for the mapping from
### 4. Google Dictionary and its Fusion with AIDA

<table>
<thead>
<tr>
<th>Word</th>
<th>Conditional Probability</th>
<th>URL</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germans,</td>
<td>0.37037</td>
<td>Germans</td>
<td>W:10/27 W08 W09 WDB</td>
</tr>
<tr>
<td>Germans,</td>
<td>0.296296</td>
<td>Germany</td>
<td>D W:8/27 W08 W09 WDB</td>
</tr>
<tr>
<td>Germans,</td>
<td>0.185185</td>
<td>Nazi_Germany</td>
<td>D W:5/27 W08 W09 WDB</td>
</tr>
<tr>
<td>Germans,</td>
<td>0.037037</td>
<td>Carpathian_Germans</td>
<td>W:1/27 W08 W09 WDB</td>
</tr>
<tr>
<td>Germans,</td>
<td>0.037037</td>
<td>German_Empire</td>
<td>W:1/27 W08 W09 WDB</td>
</tr>
<tr>
<td>Germans,</td>
<td>0.037037</td>
<td>German_cuisine</td>
<td>W:1/27 W08 W09 WDB</td>
</tr>
<tr>
<td>Germans,</td>
<td>0.037037</td>
<td>Germanic_peoples</td>
<td>W:1/27 W08 W09 WDB</td>
</tr>
</tbody>
</table>

Figure 4.1.: Conditional Probability on an exotic Example for word = "Germans,"

word = "Germans," to url = "Germany" is defined by the rational "8/27" which describes that in 8 out of 27 cases the word = "Germans," links to url = "Germany".

The `<score>`

is a collection of labels that expresses the attributes of the mapping from the `<word>` to the `<url>`. In earlier developed knowledge bases [35], the creators of the Google Dictionary have already introduced the first labels. At that time two types of labels existed, labels for booleans \(t\), \(c\) and \(d\) as well as labels for rationals \(W\) and \(w\).

Booleans have the format `<label>` whereas rationals have the format `<label><colon><numerator><slash><denominator>` [9].

The above mentioned labels have the following meanings [35]:

- **t** - title: `<word>` is the title of the page which is a cluster represented by `<url>`
- **c** - clarification: states that the title of a page in the cluster that is represented by `<url>` is `<word>` or `<word>` (...)  
- **d** - disambiguation: is a disambiguation page with title `<word>` that refers to a page in the cluster represented by `<url>`
- **W** - states that out of a total of `<denominator>` external Web links into the cluster represented by `<url>`, `<numerator>` many had anchor text `<word>`
- **w** - is analogous for inter-Wikipedia links into English Wikipedia articles
4.2. Comparison between AIDA and Google Dictionary

With the time more labels were added. In the current version of the Google Dictionary the score contains three different types of labels: rationals, booleans and integers. Integers have the format \(<label><colon><count>\) \([9]\). A complete list of all labels used in the Google Dictionary can be found in the appendix (A). Some of them will be described in further sections of this thesis.

To summarize the previous passages, a line in the Google Dictionary contains four fields, each separated by a tab (between the word and the rest) or white-spaces (between the other elements of the line). Accordingly a complete line in the data collection could look like this:

Germans, 0.296296 Germany D KB W:8/27 W08 W09 WDB

Since Google Dictionary has not been released in a browsable version, one part of this thesis was the integration of the data in a newly created database. The key data for the Google Dictionary database are as follows:

- Database name: *google_dictionary*
- Table name: *word_to_concept*
- Fields: `id`, `word`, `conditional_probability`, `url`, `score`
- Index search: yes, for each field
- Number of inserted entries: 297,073,139

After having set up the Google Dictionary database, the next step of exploring Google Dictionary could be done, namely comparing its data with the data of AIDA.

4.2 Comparison between AIDA and Google Dictionary

In the interest of finding a way to re-engineer AIDA's data repository, it has been necessary to evaluate the existing means table. For this sake, AIDA's table has been explored by focusing on its performance compared to the Google Dictionary. In principle, the comparison has been done by considering significant examples that have been checked for the following main attributes: number of useful and noisy entries, nicknames, disambiguation pages, redirects, anchors and consistency in lexical syntax. All of them are known as having effects on the efficiency of a disambiguation tool \([34]\).

Figure 4.2 shows the results of comparing AIDA with Google Dictionary according to the selected attributes. For example, it is clear that noisy entries should be excluded when it makes no sense to map them to a certain concept. In addition, it is well known that redirects and disambiguation pages are good choices for mappings since they represent exactly what disambiguation stands for. Nicknames are also highly wanted entries since they belong to the class of synonyms and should be included as well. As already mentioned in former parts of this thesis, anchor texts are the first choice of collecting entries since they are easily to detect and widely spread around the Web. Their drawback is only that manually generated anchors can also be noisy. Conclusively, all the above attributes have been checked to make statements about the goodness of entries contained in both databases.
4. Google Dictionary and its Fusion with AIDA

<table>
<thead>
<tr>
<th>Attributes</th>
<th>AIDA</th>
<th>Google Dictionary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Noise</td>
<td>Acceptable</td>
<td>A lot</td>
</tr>
<tr>
<td>Nicknames</td>
<td>Yes</td>
<td>Yes, more coverage</td>
</tr>
<tr>
<td>Disambiguation pages</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Redirects</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Anchors</td>
<td>Yes</td>
<td>Too many caused by noise</td>
</tr>
</tbody>
</table>

Figure 4.2.: AIDA versus Google Dictionary [1]

When looking at the results, replacing AIDA Means Table by the data of Google Dictionary is no option, since AIDA performs already good. The matter with AIDA Means Table is that it contains only a few number of entries but is at the same time so complex that adding not fine-tuned entries would cause its computationally costs to gain immeasurably. Furthermore, AIDA’s data is in contrast to Google Dictionary not consistent in case of punctuation, capitalization and pluralization. On the Google Dictionary site, it seems that Google Dictionary performs well in all categories but its big drawback is the huge amount of noisy entries that it contains. Integrating the complete data set into AIDA would hence not come into considerations.

For this reason, this thesis is conform with the idea of enriching AIDA Means Table by adding good, clear and non-noisy entries from Google Dictionary.

Figure 4.3 shows a little sketch on how the data distribution could look like after fusing AIDA with Google Dictionary.

Figure 4.3.: Data distribution of Google and AIDA [1]

4.2.1 Language Matching

In order to integrate data from Google into AIDA Means Table, the data has to be adapted to the current conditions of AIDA’s data. Since AIDA leverages YAGO2,
it only contains English Wikipedia concepts. In contrast, Google Dictionary links to English and non-English Wikipedia articles. Consequently, language matching is an issue that has to be dealt with. For fusing AIDA with Google the language only concepts of English Wikipedia articles should be considered. This problem can be easily resolved by using the score label information.

As mentioned before, the field "score" contains different labels, each associated with a specific attribute of the entry it belongs to. There are four labels that concern entity linking to English Wikipedia articles, $W$, $w$, $w'$ and $W_x$.

Figure 4.4 illustrates the entity linking described by those four labels. As a special case, $w'$ is analogous to $w$ but was obtained from a Google crawl in 2011 [9].

![Figure 4.4.: Wikipedia article linkage in Google and AIDA](image)

In this thesis all labels are considered except for non-English concepts. That means that entries with label $W_x$ should not be considered as addable resources. The clue here is that non-English articles can be parallel to the English ones. An entry belonging to this type of concepts is thus assigned with a $W$ and a $W_x$ label at the same time [9]. Such entries are also covered in this thesis.

### 4.2.2 Scoring Google Dictionary

Another important step to the data integration is to distinguish between good and bad Google Dictionary entries since only the good ones should be added to AIDA. For this reason, a method had to be found which makes it possible to judge the performance of a Google Dictionary entry.

The first idea that would come to one's mind when thinking about it, would be to use the conditional probability that is already contained in each mapping pair. The motive for not taking the conditional probability as scoring function for Google’s entries is that it is a probability per word and not per concept. As seen before, concepts are always useful since they describe a Wikipedia article whereas words
4. Google Dictionary and its Fusion with AIDA

can vary in their quality. Taking this into account, using a probability measure capturing the concept would be more appropriate.

Such a probability can be computed by using again the score label information of \( W, w, w' \) and \( Wz \). These rational labels contain hit counts and state how many links out of a total number of links into a Wikipedia article contained a specific word.

\[
\begin{array}{|c|c|c|c|c|c|}
\hline
\text{Word} & \text{URL} & W & W_x & w & w' \\
\hline
\text{Germany} & \text{Germany} & 45 & 10 & 5 & 3 & \sum \leq 63 \\
\text{German} & \text{Germany} & 24 & 6 & 3 & 2 \\
\text{Democratic Republic of Germany} & \text{Germany} & 15 & 4 & 1 & 3 \\
\text{Deutschland} & \text{Germany} & 30 & 6 & 3 & 2 \\
\text{Berlin} & \text{Germany} & 21 & 7 & 1 & 1 \\
\hline
\end{array}
\]

Score (Word - Germany) = 63 / 192 \approx 0.32

\Rightarrow 32\% \text{ of all links to the Wikipedia article with the URL = Germany contain the word Germany}

Figure 4.5.: Score Probability Calculation - on example url = "Germany" [1]

Figure 4.5 shows an example calculation of the so-called score probability for the url = "Germany". It is simply computed as a fraction of the sum over all labels for the specific word mapping to the given url divided by the sum over all label counts for all words mapping to the given url.

\[
\begin{array}{|c|c|}
\hline
\text{Word} & \text{Score Probability in \%} \\
\hline
\text{Germany} & 34.443 \\
\text{German} & 9.920 \\
\text{Allemagne} & 3.872 \\
\text{Duitsland} & 1.486 \\
\text{Jerman} & 1.159 \\
\text{Germania} & 0.800 \\
\text{Tyskland} & 0.646 \\
\text{Federal Republic of Germany} & 0.582 \\
\text{Duitse} & 0.550 \\
\text{allemand} & 0.493 \\
\hline
\end{array}
\]

Figure 4.6.: Top-10 Google Dictionary Results for url = "Germany" [1]

Figure 4.6 shows the top-10 results for url = "Germany" sorted by their calculated score probabilities. For comparison 4.7 shows the top-10 results for url = "Germany" in YAGO2.
4.3 Filtering Google Dictionary

After having found a measure that scores Google Dictionary entries, the next step would be to eliminate noisy entries from Google’s data set.

4.3.1 Filtering by Use of Thresholds

One may assume that noisy entries have small score probabilities. Hence, the first idea of filtering noisy entries from Google Dictionary would be to use the computed score probabilities as thresholds. All entries below the threshold could then be ignored. During the development of this thesis different threshold values have been manually tested, coming to the conclusion that thresholds between 0.0005 and 2% are suited best. In the range between 0.5 and 2% the results stop to vary and reach their maximum of usefulness containing at the same time the smallest number of entries.

<table>
<thead>
<tr>
<th>Threshold</th>
<th>Useful entries</th>
<th>Total number of entries</th>
<th>Percentage of gained entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0005</td>
<td>371</td>
<td>1060</td>
<td>35.00%</td>
</tr>
<tr>
<td>0.001</td>
<td>236</td>
<td>659</td>
<td>35.81%</td>
</tr>
<tr>
<td>0.005</td>
<td>113</td>
<td>286</td>
<td>39.51%</td>
</tr>
<tr>
<td>0.01</td>
<td>96</td>
<td>213</td>
<td>45.01%</td>
</tr>
<tr>
<td>0.05</td>
<td>53</td>
<td>83</td>
<td>63.85%</td>
</tr>
<tr>
<td>0.1</td>
<td>38</td>
<td>51</td>
<td>74.51%</td>
</tr>
<tr>
<td>0.5</td>
<td>10</td>
<td>14</td>
<td>71.43%</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>8</td>
<td>75.00%</td>
</tr>
<tr>
<td>1.5</td>
<td>3</td>
<td>3</td>
<td>100.0%</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>100.0%</td>
</tr>
</tbody>
</table>

Figure 4.7.: Top-10 YAGO2 Results for url = "Germany" [1]

Figure 4.8.: Results for url = "Germany" when filtering by thresholds [1]
4. Google Dictionary and its Fusion with AIDA

obtained by filtering all results for url = "Germany" according to the threshold value and after by manually labelling and counting the number of useful entries that was left after the application of the threshold filter. Using for example a threshold value of 0.0005% only 371 entries out of 1060 entries would be considered as useful leading to the fact that only 35% of the found entries for url = "Germany" would be acceptable to add to AIDA Means Table. With increasing threshold the usefulness of the entries increase whereas the number of addable entries decreases dramatically as well.

Accordingly, the former assumption has been wrong, noisy entries do not have small scores. The best example for this is the word "Wikipedia" that is part of each result set obtained for any query. It is often scored highly, among the top-results, although it is a noisy entry that does not disambiguate any concept except for url = "Wikipedia" itself.

4.3.2 Filtering by Combination of Thresholds and Patterns

Another idea of filtering for noise is to combine the threshold filter with a filter based on pattern matching. For this, the noisy entries had to be analyzed and frequent patterns in the words that occur over and over again had to be found in long lasting experiments. Any word containing one of the defined patterns will then be eliminated in the filter process.

The list below summarizes all used filter patterns and gives some examples accordingly.

Filtering for:

- **common words:**
  - wiki|pedia|read|source|site|page|more|about|article|
  - click|see|here|view|search|link|download|http|www|url

  Examples:
  - http://en.wikipedia.org/wiki/Germany
  - Germany - Wikipedia, The Free Encyclopedia
  - Click here
  - ..read more
  - View article on Wikipedia Â»

- **Strings smaller than 2 letters:**
  - Examples: 1, W

- **Strings between <>**
  - Examples: </SPAN>

- **Strings between []**
  - Examples: [COLOR=0000ff]Germany[/COLOR], [93]
4.3. Filtering Google Dictionary

- Entries where < or > occurs more than once
  Examples: Read Article Â»

- The pattern string=
  Examples: Flag of=, target="_blank">"

- The pattern string:string
  Examples: en:GermanyInfrastructure

- Strings containing ?
  Examples:
  Where did this come from?
  Where is Germany?

- open string window and new string window
  Examples: Open in a new browser window

- Point . followed by at least 2 letters
  Examples:
  Flag of Germany.png
germany.de

- Entries with a length larger than 30 letters
  Examples:
  LIKE who think today win gernany...
fuckyeahgermany: The fall of the Berlin Wall, 1989

- [digit and digit]
  Examples: [1, 4]

- digit% and digit %
  Examples: 90% white, 90 %

- digit.digit
  Examples: 1.1 Etymology

- digit at the beginning
  Examples: 2 Politics

Figure 4.9 gives the results for filtering by threshold and patterns. As can be seen, the number of useful entries reduced enormously whereas the percentage of useful entries that are amongst them increased a lot.

The example in this chapter were based on the concept "Germany" and proved to obtain a possible gain of entries for AIDA of at least 82%. In regions where 100% coverage is depicted, the entries seem to be restricted to concepts that are from non-English data but are parallel to the concept in English. All ignored entries
4. Google Dictionary and its Fusion with AIDA

<table>
<thead>
<tr>
<th>Threshold</th>
<th>Useful entries</th>
<th>Total number of entries</th>
<th>Percentage of gained entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0005</td>
<td>371</td>
<td>448</td>
<td>82.81%</td>
</tr>
<tr>
<td>0.001</td>
<td>236</td>
<td>284</td>
<td>83.1%</td>
</tr>
<tr>
<td>0.005</td>
<td>113</td>
<td>132</td>
<td>85.6%</td>
</tr>
<tr>
<td>0.01</td>
<td>96</td>
<td>109</td>
<td>88.1%</td>
</tr>
<tr>
<td>0.05</td>
<td>53</td>
<td>55</td>
<td>96.4%</td>
</tr>
<tr>
<td>0.1</td>
<td>38</td>
<td>38</td>
<td>100%</td>
</tr>
<tr>
<td>0.5</td>
<td>10</td>
<td>10</td>
<td>100%</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>6</td>
<td>100%</td>
</tr>
<tr>
<td>1.5</td>
<td>3</td>
<td>3</td>
<td>100%</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>100%</td>
</tr>
</tbody>
</table>

Figure 4.9.: Results for url = "Germany" when filtering by threshold and patterns [1]

have been checked as well and it can be said that only 2% of them seemed to be useful. Hence, using this filter strategy does not end in a significant loss of useful entries.

The only problem that occurs here is that with increasing threshold value the total number of entries decreases extremely.

4.3.3 Filtering by Use of Filter Patterns

To complete the search for an adequate filter, using only filter patterns without threshold has been tested. As figure 4.10 depicts, filtering by patterns optimises the number of useful entries proportional to the total number of entries. In the example, 81% of entries were judged to be useful and hence be addable to AIDA Means Table.

<table>
<thead>
<tr>
<th>Useful entries</th>
<th>Total number of entries</th>
<th>Percentage of gained entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>900</td>
<td>1109</td>
<td>81.15%</td>
</tr>
</tbody>
</table>

Figure 4.10.: Results for url = "Germany" when filtering by patterns [1]

4.4 Discussion of the Results

Summarizing the obtained results, it can be seen that score probabilities do not always open the way to good entries. Especially when entries were obtained from anchors that may have high frequencies. Then a larger threshold would have to be applied which would then cause a loss of many useful entries, upto one third of the entries.
4.4. Discussion of the Results

Using a combined filter of threshold and patterns led only to moderate results that were also accompanied with a higher loss of entries.

Filter patterns perform in case of percentage similar to thresholds but are more robust. They can easily be adapted and new patterns can be added. Moreover, they do not reduce the amount of data that could be newly included to AIDA.

By using patterns for filtering the noisy entries of Google Dictionary, it can not be excluded that Google Dictionary could become the new data resource for AIDA Means Table. But before, Google Dictionary’s data have to be explored in a second round where the quality of the entries is achieved in more detail.

To make the first steps into evaluating Google Dictionary easier and the accessing of the data more comfortable, a Web-based browser for Google Dictionary has been implemented that is presented in the next chapter (5).
5. Web-based Browser Design

In this chapter the Web-based browser developed for browsing the Google Dictionary will be described in details.

5.1 Program Flow

The browser has been implemented on base of Java, JavaScript, JQuery, Html and CSS.
5. **Web-based Browser Design**

Figure 5.1 shows the program structure of the browser in case of searching for concepts. The search for words is equivalent but skips the steps for calculating the score probability since these are already known by the conditional probabilities.

### 5.2 Google Dictionary Search Web Browser

Figure 5.2 shows the start view of the Google Dictionary Search when loading the page. It offers the possibility to search for any word or any concept in the complete data set of Google Dictionary or in the Fusion of the already denoised and filtered Google data with AIDA. The inserted entry should be case sensitive. In addition the browser offers different filter options, one for filtering by using a threshold value that displays the score probabilities and one for filtering by using filter patterns.

Figure 5.2.: Start view of the Google Dictionary Search [37]: 1 - search field; 2 - search direction, from words to concepts or vice versa; 3 - database selection, Google Dictionary only or Fusion of Google and AIDA; 4 - filter selection, by threshold, by patterns or both; 5 - helpbox with usage instructions;

Detailed Instructions on the usage of the Google Dictionary Search can also be read in the helpbox that is provided when going with the mouse over the "questionmark" button. Furthermore, the browser provides tooltips that pop up when staying with the mouse a few seconds on an element of the page.

Figure 5.3 shows the helpbox as well as the tooltip provided for the search query field.

Figure 5.4 shows the browser after having sent the request for the concept "Germany" in the entire Google Dictionary. The results are displayed in a table consisting of three columns. In the current example the columns are "Word", "Score Probability" and "Score Attribute" from which the third column is actually hidden. The probabilities are in percentages and the score attributes can be shown by clicking on the "Show Score Attributes" button if necessary. In this way it can also be hidden by clicking on "Hide Score Attributes".

The browser allows site-wise navigation through the found results by using a usual
5.2. *Google Dictionary Search Web Browser*

Figure 5.3.: Tooltips and Helpbox of the Google Dictionary Web Browser[37]: 1- tooltip for the search field; 2 - helpbox with instructions;

paginate function. In figure 5.4 another feature of the browser can also be seen, namely the option of downloading the results as .CSV file by clicking on the "Download Results" button.

Figure 5.5 shows the opened "Score Attributes" column. For this a special feature has been added, the score can be filtered for different labels as can be seen on an example in figure 5.6 that illustrates filtering for the label "w". Moreover, the dropdown menu of the labels provide tooltips so that it will be easier to reflect on the meaning of a particular label.

The next figure 5.7 draws an example where the fusion of Google and AIDA has been selected as database. Special is here that all entries that Google and AIDA share are written in black, whereas entries that are only in AIDA are written in red and entries only available by Google are in green. In this way it can easily be seen which entries AIDA would gain when accepting to add datza from Google to its Means Table, namely the green ones. As can be seen, the score probabilities for the red entries are not applicable (N/A) since Google Dictionary does not contain the probabilities for AIDA. Here is place for further improvements, the probabilities for AIDA could for example be taken by its prior probability values in YAGO2.

All functionalities of the browser have in detail been tested on Firefox but a sketchy test on Google Crome has also been succesfull.
Figure 5.4.: Result Table of the Google Dictionary Web Browser - after search for concept in entire dictionary [37]: 1 - site-wise navigation; 2 - hidden third column; 3 - column "Word"; 4 - column "Score probability"; 5 - download link for .CSV file
## 5.2. Google Dictionary Search Web Browser

![Figure 5.5: Result Table with opened "Score Attribute" column [37]: 1 - "Score Attributes" column; 2 - dropdown for label filtering;](image)

<table>
<thead>
<tr>
<th>Word</th>
<th>Score Probability in %</th>
<th>Score Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Germany</td>
<td>34,357460022</td>
<td>KB W:156792/197339 W:88 W:99 W:DB</td>
</tr>
<tr>
<td>German</td>
<td>9,989212036</td>
<td>KB W:42393/175856 W:88 W:99 W:DB</td>
</tr>
<tr>
<td>ドイツ</td>
<td>1,603067875</td>
<td>KB W:38304 W:88 W:99 W:DB</td>
</tr>
<tr>
<td>Deutsch</td>
<td>1,478073835</td>
<td>KB W:67120 W:88 W:99 W:DB</td>
</tr>
<tr>
<td>독일</td>
<td>1,338743210</td>
<td>KB W:9301187 W:88 W:99 W:DB</td>
</tr>
<tr>
<td>Germany - Wikipedia, the free encyclopedia</td>
<td>1,311183333</td>
<td>KB W:423094380 W:88 W:99 W:DB</td>
</tr>
<tr>
<td>言語</td>
<td>1,179647446</td>
<td>KB W:128135 W:88 W:99 W:DB</td>
</tr>
<tr>
<td>German</td>
<td>1,53644277</td>
<td>KB W:214484 W:88 W:99 W:DB</td>
</tr>
<tr>
<td>Germania</td>
<td>0,795618236</td>
<td>KB W:3405906 W:88 W:99 W:DB</td>
</tr>
<tr>
<td>Wikipedia</td>
<td>0,760541975</td>
<td>KB W:269996010521 W:88 W:99 W:DB</td>
</tr>
<tr>
<td>Tyskland</td>
<td>0,642229319</td>
<td>KB W:106221 W:88 W:99 W:DB</td>
</tr>
<tr>
<td>Топонимы</td>
<td>0,6143909493</td>
<td>KB W:155181 W:88 W:99 W:DB</td>
</tr>
</tbody>
</table>

*Figure 5.5: Result Table with opened "Score Attribute" column [37]: 1 - "Score Attributes" column; 2 - dropdown for label filtering;*
Figure 5.6.: "Score Attributes" column filtered for label "w" [37]: 1 - different filter labels;
5.2. Google Dictionary Search Web Browser

![Figure 5.7. Result Table of the Google Dictionary Web Browser - after search for concept in Fusion Google and AIDA [37]: 1- black: entry in Google and AIDA; 2 - green: entry only in Google; 3 - red: entry only in AIDA; 4 - N/A for not applicable probability;](image-url)
6. Conclusion

The purpose of this thesis was the exploration of a new data resource for AIDA Means Table. For this extent, the already existing data in AIDA had to be evaluated. Among many other possible repositories, the data resource of choice in this thesis has been set to a newly released data collection by Google, named Google Dictionary.

After having understood the structure of the data provided by the Google Dictionary, a new database has been initialized to cover full access to the data. AIDA Means Table and Google Dictionary were then compared by using different attributes that could judge the performance of their entries. The following main attributes were checked: number of useful and noisy entries, nicknames, disambiguation pages, redirects, anchors and consistency in lexical syntax.

As an result, AIDA and Google performed both well but were proved to have drawbacks. The main drawback of Google Dictionary was its large number of noisy entries. Whereas the drawbacks of AIDA Means Table lied in lexical inconsistency, high complexity and expensive computational costs while containing only few entries at the same time. For this reason, having clean and accurate data is important for enriching AIDA’s disambiguation processes. Since both databases have their disadvantages, replacing AIDA Means Table by Google Dictionary did not become an option. Instead, adding good non-noisy entries from Google Dictionary has been considered.

Therefore, different filter techniques have been applied to eliminate noisy entries. In total, three filter approaches have been tested: the first - using filtering by threshold values, the second - using filtering by filter patterns and the third - being a combination of both. After several experiments with significant examples and manual evaluation, it turned out that filtering by patterns give the best results for denoising the data of Google Dictionary. For giving the possibility of further exploring the Google Dictionary, a Web-based browser has been developed additionally.

The concepts of this thesis could be optimized in multiple ways as shown in the next chapter (7).
7. Further Work

Providing a correct and robust means table for AIDA’s named entity disambiguation is a key issue to assure the system’s accuracy. For this reason, it is important to continue the ambition for finding new data resources for AIDA Means Table. Further research is necessary in the following areas:

- **Confirming Experimental Results by User Poll**
  In order to assure the correctness of the experimental results obtained from the author of this thesis, an user poll could be initialized in which people can browse the Google Dictionary and mark each entry obtained from the results as useful or not. In this way it can be seen which disambiguation of particular mentions imitate more likely the natural language understanding of human beings. Hence, the data of Google Dictionary could be explored in a more general way instead of relying on the subjective judgement of a single person.

- **Training Classifier for Assigning Entries to Classes**
  When sending a search request in the direction from concepts to words, among the resulting words there are often results that express attributes about a real-world fact concerning the search word. For example when inserting the concept "Michael Jackson" the term "singer" would occur in the result table. This is the profession of the concept describing a person. Another example would be to look for "Berlin" and to obtain the word "Germany" which is the location of the city or the word "capital city" that describes the fact it is a city and in addition the capital city of Germany. This information could be used to apply class relations among words and concepts. An idea would be to use data from Google Dictionary and train a classifier that assigns entries to classes, similar to the relations in YAGO2.

- **Using Score Labels for Further Classification**
  As already mentioned, labels describe entries in detail, like referring to their origin, their occurrences in an article, their language and other attributes. In addition to point two this feature of Google Dictionary’s data could then be used to train a classification model. As an introductory approach, redirects and disambiguation pages of AIDA Means Table could be used as ground truth data, so that all entries with labels r (redirect) and d (disambiguation page) could be added to AIDA as they are assumed to be clean entries. Another idea is to use the label t which refers to entries that are explicit links, such that the title of the Wikipedia article is equal to the word itself.
7. Further Work

One possibility would be to use only these entries and see how many new data would be added to AIDA Means Table.

• **Improving the User Interface**
  Since the Google Dictionary contains a large dataset of entries which might be ambiguous or hard to remember, it might be useful to provide a more comfortable browsing environment for the users. For this reason, an auto-complete function could be added to the dictionary which would enable the user to search faster and more accurate by offering possible search query entries that the user might be looking for. These entries should be shown immediately while the user is typing. Unfortunately, average auto-complete methods are not able to browse through large amounts of entries in a few milliseconds. Therefore, the best way to avoid runtime problems would be to aim for a more optimized auto-complete method.

• **Exploring Other Data Resources**
  Besides Google Dictionary there are other data resources whose data could be used to enrich AIDA Means Table. Examples of such repositories are Freebase, WordNet or the newly established co-reference corpus of Singh et al. that is similar to Google Dictionary but uses additional data from Wikipedia for context-based issues. For whichever the decision falls the new data source should provide clean, accurate and consistent data that can reduce the complexity and expensive computation of AIDA.
Appendices
A. Score Attributes

In the following all labels of the Score Attributes column and their meanings are listed [9] [38]:

- **rationals**: `<label>:<numerator>/<denominator>
  - W states that out of a total of `<denominator>` external (non-Wikipedia) web links into the article’s (English) cluster, `<numerator>` had hyper-text `<string>`
  - Wx is similar, but for links into articles in other languages that are parallel to ones in English
  - w is analogous, for inter-English-Wikipedia links, based on data from several older Wikipedia dumps
  - w’ is another estimate of w, based on a recent crawl
  - dl expresses that this anchor was the `<numerator>`th link (out of a total of `<denominator>` hyper-links) in a line of text of a Wikipedia disambiguation page
  - dt is analogous, but measured in numbers of tokens

- **integers**: `<label>:<count>`
  - ds states that a line in the disambiguation page from which the link was extracted started with `<count>` stars
  - dc is also analogous (to dt), but measured in characters, i.e., the link’s position in a line (after any stars)

- **booleans**: `<label>`
  - D indicates that at least one article in the cluster represented by `<url>` is a disambiguation page
  - l analogous, but is a list-of... page
  - m also analogous, but is a meanings-of... page
  - L if `<url>` itself is a list-of... page
  - M if `<url>` itself is a meanings-of... page
A. Score Attributes

- **KB** indicates that `<url>` is in the knowledge-base associated with the 2009-11 TAC-KBP EL/SF tasks
- **W08** one of the cluster’s articles is known from a 2008 Wikipedia dump
- **W09** one of the cluster’s articles is known from a 2009 Wikipedia dump
- **WDB** one of the cluster’s articles is known from a DBpedia dump
- **UNK** is for unknown `<url>` values (very rare)
- **RWB** indicates that `<url>` redirects on the web
- **R08** states that it was a redirect in the 2008 dump
- **R09** in the 2009 dump
- **RDB** in the DBpedia dump
- **NR** states that even though a redirecting representative `<url>` was chosen, there is at least one non-redirect in its cluster (which was dispreferred for other reasons)
- **CHRON:CENTURY** indicates that an article is about a century
- **CHRON:DATE** indicates that an article is about a date
- **CHRON:DECADE** indicates that an article is about a decade
- **CHRON:MILLENIUM** indicates that an article is about a millennium
- **CHRON:YEAR** indicates that an article is about a year
- **c** is for clarification, either `<string>` or `<string>_(...)` is the (English) title of some page in the cluster
- **f** another clarification: `..._(<string>)` is the title
- **t** if `<string>` itself is the title.
- **h** is for hash: there exists a(n English) link of the form `...<string>` into one of the pages in `<url>`’s cluster
- **H** indicates the existence of the actual internal anchor
- **r** a(n English) page with title `<string>` redirects into this cluster
- **d** a(n English) disambiguation page with title `<string>` links to some page in the cluster represented by `<url>`
- **p** is also on if `d` was signaled by the noisy WDB resource
- **x** for cross, indicates that `<string>` was obtained from non-English data (and would differ from Wx>0 if we had also mined non-English titles, etc.)
Bibliography


Bibliography

lands.


[34] Additional material from the author's supervisor.


[37] Screen shots from the Google Dictionary Search Interface, created by the author of this thesis.