# Exploiting DBpedia for Graph-based Entity Linking to Wikipedia

Master Thesis

presented by
Bernhard Schäfer

submitted to the
Research Group Data and Web Science
Prof. Dr. Christian Bizer
University Mannheim

May 2014

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Motivation

Unlike for human beings, for a computer it is very challenging to tell which sense a word in a sentence refers to. This can be illustrated by the following sentence:

*Napoleon was defeated at Waterloo by Wellington.*

Due to existing knowledge, a human immediately knows that in this context, *Waterloo* refers to the battle which took place in Belgium, and not the city in Canada. Likewise, *Napoleon* refers to Napoleon Bonaparte, and not one of his descendants with the same name. In the same manner, *Wellington* does not refer to the city in New Zealand, but to Arthur Wellesley, 1st Duke of Wellington, the commander of the allied army in the battle of waterloo.

The task in which a computer tries to derive the senses of words is called word-sense disambiguation (WSD), an open problem in natural language processing (NLP). With the increasing popularity of Wikipedia in the 2000s decade, the task of named entity linking (NEL) emerged, with the goal of linking words or phrases in text (surface forms) to their respective entities in a knowledge base. In case Wikipedia is used as the knowledge base, the surface forms are linked to their matching Wikipedia article. The task of linking entities to Wikipedia is also called Wikification [29].

The most successful approaches to entity linking, to date, are based on supervised learning methods [9]. Hereby, the article texts in Wikipedia and their containing interlinks, along with the anchor texts, are commonly used as training data. The first publication on supervised entity linking to Wikipedia was published in 2007 by [29]. At the time of writing, all state of the art performing entity linking systems still depend to a large extent on the features explored in the publication from 2007.

For a big step forward to the accuracy of a human linker, it is essential to also investigate on other approaches to entity linking. Here, the attention could be more directed towards knowledge-based approaches, since they are closest to

mimicking the approach of a human linker. This circumstance is an advantage of knowledge-based approaches, since a human being has an accuracy close to 100%. For example, a human might exploit the fact that Napoleon Bonaparte was the commander of the french troops in the Battle of Waterloo to reason that the surface form Napoleon refers to Napoleon Bonaparte in the previous example. This exploited fact can also be found in the infobox of the *Battle_of_Waterloo* Wikipedia article[1], and thus, it could be used by a computer in a similar fashion.

Earlier attempts to exploiting a knowledge source for disambiguation mostly focussed on the task of WSD and used WordNet as a knowledge base (cf. [32]). However, there are only a few attempts at exploiting Wikipedia for knowledge-based entity linking (cf. [41]). This might be due to the fact that in contrast to sources such as WordNet, the Wikipedia infobox statements are not structured and contain errors, which makes their usage tedious. Fortunately, in more recent years community efforts such as DBpedia have been founded, which extract structured information from Wikipedia and make this information available on the Web [2, 3]. To this end, the community created an automated extraction framework that creates structured datasets in the form of RDF triples, which can essentially be interpreted as a giant graph.

The mentioned facts lead to the conclusion that entity linking could substantially benefit from further research on exploiting DBpedia for graph-based entity linking to Wikipedia.

## 1.2 Problem Description

The present thesis addresses two main research problems. The first topic is centered around exploring DBpedia for novel approaches to graph-based entity linking to Wikipedia. To set graph-based linking into context with the bigger picture of entity linking, the second aspect is concerned with incorporating the graph-based methods into state of the art supervised approaches, with the aim of combining the strenghts of both.

The work on graph-based linking is broken down into four main phases. To enable graph-based entity linking based on DBpedia, at first a graph suited for disambiguation is constructed from the DBpedia datasets. This involves determining the appropriate datasets and figuring out which data from DBpedia is irrelevant for linking (e.g. image-related triples), and thus should be excluded from the graph representation. After the DBpedia graph has been constructed, methods are developed to traverse the graph efficiently. This involves various steps such as figuring out if the graph should be interpreted as directed or undirected, and the determination of an appropriate traversal depth limit. To further improve the expressiveness of connections in the graph, another aspect is the weighting of semantic relations to emphasize especially relevant predicates or predicate-object combinations. Thereby, different measures from existing research are evaluated (cf. [36]).

[1] http://en.wikipedia.org/wiki/Battle_of_Waterloo

To link the surface forms in text to their entity references, there are numerous graph-based connectivity measures. Therefore, different graph-based algorithms for entity linking are evaluated. For the similar task of word-sense disambiguation (WSD), different local and global graph algorithms have been already proposed by [32]. Accordingly, the present work looks at these very same algorithms, as well as extends this line of work to other algorithms, attempting to mitigate the drawbacks exposed from the evaluation of these algorithms.

For forming a federated linker that combines the graph-based with state of the art supervised techniques, DBpedia Spotlight is used as the supervised system. To assess the strengths and drawbacks of both linkers, an error analysis based on a random sample of documents is conducted for both. Based on the results, different methods for combining both linkers are developed and evaluated, with the goal of exploiting the synergies between both to enhance the state of the art statistical approach.

The remainder of this work is organized as follows: The next chapter discusses the state of the art in entity linking (Chapter 2). Next, the methodology of the present work is discussed, focussing on the various aspects in constructing the graph-based linker and how to integrate this linker with state of the art supervised methods (Chapter 3). This is followed by an extensive evaluation (Chapter 4). The evaluation serves not only to compare the performance of graph-based and other state of the art approaches, but also to use actual data to justify the numerous design decisions involved in constructing the graph-based linker. Finally, some conclusions are drawn from this work (Chapter 5).

# Chapter 2

# Related Work

The goal of this chapter is to outline the research that has happened in entity linking to Wikipedia and discuss how the graph-based approach of the present work fits in. Since the first papers on entity linking to Wikipedia have been published in 2007 (cf. [10, 29]), the topic has gained a lot of attention from the research community. The majority of works on entity linking propose and evaluate full entity linking systems. Among others, these systems can be characterized along two dimensions:

- *Entity Linking Process*: The steps that the entity linker performs to transform a raw document input to a document with annotations to Wikipedia.
- *Entity Linking Features*: The methods used within each step of the process. The features used within each step of the process can be best categorized and distinguished by their underlying *data sources*. Hereby, the vast majority of systems solely rely on features which are directly or indirectly drawn from Wikipedia.

Figure 2.1 summarizes the detected relationships between the entity linking *process*, *features* and *Wikipedia data sources* that have been drawn from the surveyed related work.

The remainder of the chapter is structured as follows: The next section addresses the entity linking process from a high-level perspective (Section 2.1). This is followed by a discussion of the different data sources that aid throughout the entire process (Section 2.2). Overall, this section provides a foundation for the rest of the chapter, since all features in each step of the linking process are based on a subset of these sources. The elaboration of the data sources is followed by dedicated sections for each step within the linking process. Thus, a discussion about techniques spotting (Section 2.3) is followed by a dedicated section for candidate selection (Section 2.4). Next, the disambiguation phase is discussed (Section 2.5). Since the focus of the present thesis is placed on the disambiguation step, this section is discussed in a detailed manner, including a comparison of all features used within entity linking systems and an analysis of which and how each of those features is utilized within state of the art these systems.

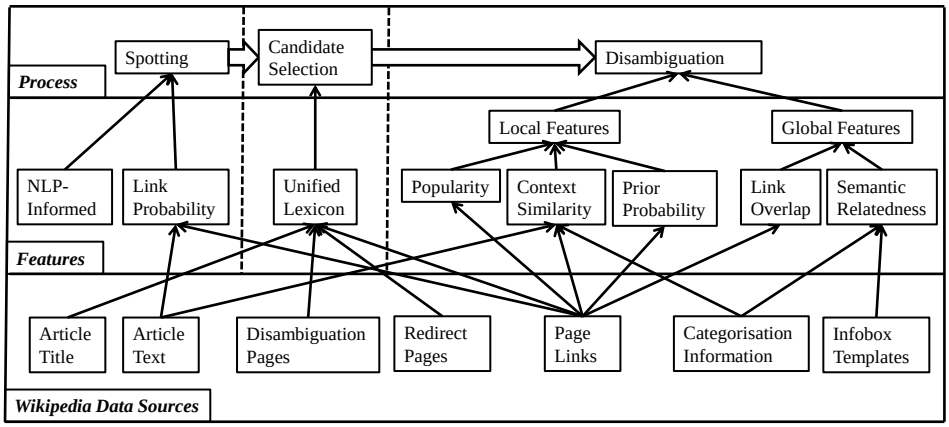


Figure 2.1: Entity Linking Process-Features-Data-Sources Relationships

## 2.1 The Entity Linking Process

The entire entity linking process can be split up in multiple steps. The first works on entity linking distinguished between two steps, *detection* and *link disambiguation* [29].

The first step, which is commonly referred to as *spotting* (cf. [28]), *detection* (cf. [31]) or *keyword extraction* (cf. [29]), involves identifying the terms and phrases from which the links should be made. Throughout research, these terms or phrases are called *spots*, *mentions*, or *surface forms*. The techniques for spotting are further elaborated in a dedicated section (2.3).

The link disambiguation phase is concerned with finding the correct Wikipedia article that should be linked to a keyword or phrase. Follow-up works have further split up this step into the two phases *candidate selection* and *disambiguation* [28]. The candidate selection step is concerned with generating potential entity candidates in the form of Wikipedia articles for each prior identified surface form. The next step, commonly referred to as *disambiguation* (cf. [31, 28]) or *candidate ranking* (cf. [8]), deals with finding the best matching candidate entity for each surface form. Hereby, candidate ranking is a generalisation of the disambiguation process, since disambiguation merely returns the candidate with the highest score, whereas ranking produces an ordered list of candidates, optionally enriched with confidence scores.

More recent work also identified a subsequent step to deal with *NIL* surface forms [8, 11]. This step is based on the assumptions that not all surface forms have corresponding articles in Wikipedia. Thus, if a surface form has no matching article, it should be mapped to a NIL entity. Since the most commonly used datasets for evaluating entity linkers, discussed in Chapter 4, do not contain annotations to NIL entities, this step does not play a major role in this work.

Throughout this thesis, the focus is placed on the three step process employed

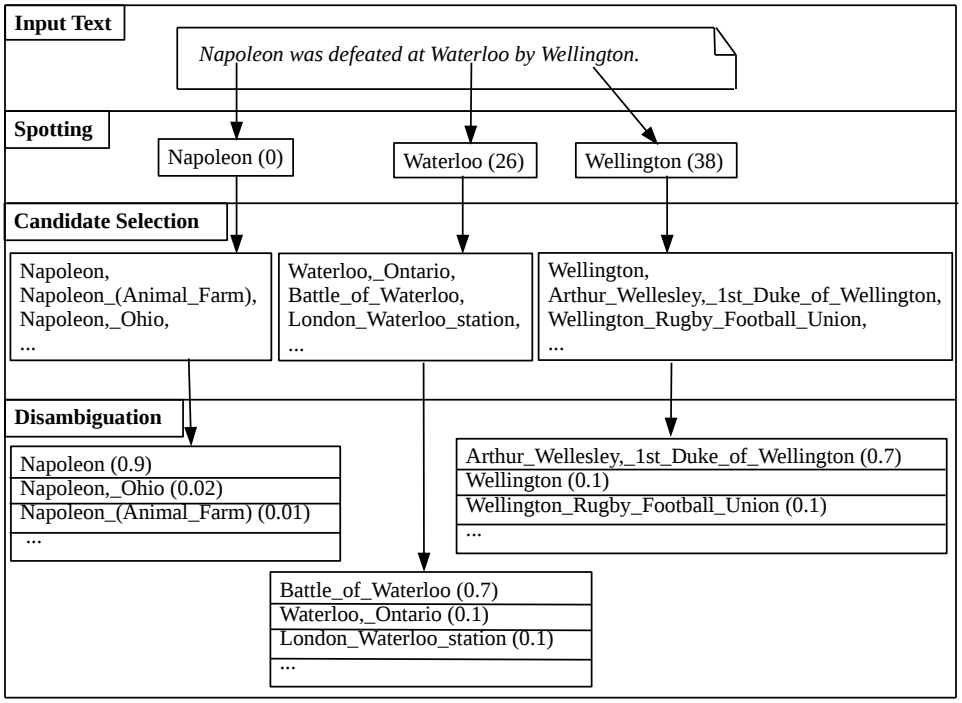


Figure 2.2: Entity Linking Process

by [28]. An overview of this process, along with an exemplary sentence, is shown in Figure 2.2. In the illustrated spotting phase, the numbers in parenthesis depict the character offset of the surface form in the overall document. This illustrates that there can be multiple surface forms with equal character strings, which can only be distinguished by their offset within the document.

## 2.2 Entity Linking Data Sources

The majority of data sources used within state of the art entity linking systems are directly or indirectly based on Wikipedia. The indirect usage happens through third-party projects such as DBpedia, a community effort to make the information of Wikipedia available in a structured manner. Since the graph-based approach of the present thesis is also focussed on DBpedia, this chapter is structured as follows: In the next section, the different data sources contained within Wikipedia are outlined (Section 2.2.1). This is followed by a dedicated section which covers the DBpedia project (Section 2.2.2).

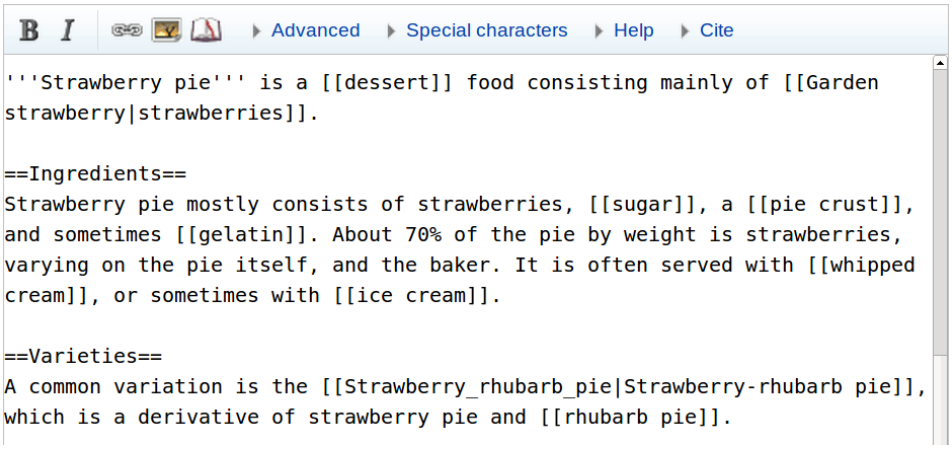


Figure 2.3: Strawberry Pie Wikipedia Article Markdown

### 2.2.1 Wikipedia

There are multiple data sources contained within Wikipedia, which are individually discussed in the remainder of this section. While doing so, the Wikipedia article of *Strawberry_Pie*[1] is used as an example to illustrate the different sources.

#### Article Text

The most straightforward data source from Wikipedia to use is the text of the Wikipedia articles. Since the articles contain an explanation of its main subject and relationship to related concepts, the words contained in the article indicate a certain relatedness to the article. This data source has been utilized by the first works in entity linking to Wikipedia [6, 10].

There are various context-based local features which leverage the article text; they are further explained in Section 2.5.1.

#### Page Links

The most prominent data source for entity linking to Wikipedia are the page links contained within the Wikipedia articles.

The Wikipedia articles are written in *wiki markup*[2], a markup language which is translated to HTML. As part of the language specification, references to other Wikipedia concepts can be made through double square brackets. Optionally, these references contain a vertical bar. In this case, the text left of the bar denotes the Wikipedia concept whereas the text on the right is the anchor text of the link. As illustrated in Figure 2.3, an exemplary anchor contained in the Strawberry Pie article is `[[Garden strawberry|strawberries]]`. Since the Wikipedia

[1] http://en.wikipedia.org/wiki/Strawberry_pie
[2] http://en.wikipedia.org/wiki/Wiki_markup

page articles are freely accessible[3], these double square brackets can be easily exploited to identify interlinking Wikipedia concepts. Through third-party projects such as DBpedia, these page links are also conveniently accessible in a structured and semantic format[4].

The page links and anchor texts within Wikipedia are used as basis in the spotting and candidate selection step, and also for many local and global features in Section 2.5.1.

### Infobox Templates

Infoboxes in Wikipedia contain core facts about an article[5]. They are displayed as panels that appear on the top right of the default view of many Wikipedia articles, or at the start for the mobile version.

Wikipedia uses a hierarchy of infobox templates that can be used for articles, where each template contains a set of properties for the respective domain. In the *Strawberry_Pie* example the template *Infobox prepared food*[6] is used, which contains properties such as *Main ingredient(s)* and *Variations*.

Overall, the infobox templates are utilized as global features within entity linking, commonly through the usage of third-party projects such as DBpedia [2] and YAGO [39], which generate structured data from these infoboxes.

### Categorisation Information

In Wikipedia, articles also contain a set of categories. These categories are roughly organized as a tree with the exceptions of several cycles.

The *Strawberry_pie* example lists the following categories: *Sweet_pies*, *German_cuisine*, *Pie_stubs*. Hereby, the *Sweet_Pies* category is a subcategory of *Pies*, which has the supercategories *Foods*, *Baked_Goods*, and *Dessert*.

This category information is exploited as local and global disambiguation features.

### Non-article pages

In addition to pages for Wikipedia articles and categories, there are other pages in Wikipedia, for example dedicated to redirects, disambiguations, lists, or images.

The redirect and disambiguation pages are useful for entity linking, especially regarding candidate selection (cf. Section 2.4). Therefore, they are shortly described in the following:

- *Redirect pages*: The Wikipedia articles commonly have a set of *redirect pages* that link to them. These redirect pages use synonyms or alternative

[3] http://en.wikipedia.org/wiki/Wikipedia:Database_download
[4] http://wiki.dbpedia.org/Downloads39#wikipedia-pagelinks
[5] http://en.wikipedia.org/wiki/Infobox#Wikipedia
[6] http://en.wikipedia.org/wiki/Template:Infobox_prepared_food

surface forms, including common misspellings and acronyms. For example, the entity *Michael_Jordan* has more than 20 different redirect pages, ranging from the full name depiction *Michael_Jeffrey_Jordan* to nicknames such as *His_Airness*.

- *Disambiguation pages*: These pages exist throughout Wikipedia for ambiguous surface forms. They contain a list of references to entities that are typically mentioned using this surface form.

### 2.2.2 DBpedia

DBpedia is a community effort to extract structured information from Wikipedia and to make this information available on the Web [2, 3]. To this end, the community created an automated extraction framework that creates structured datasets in the form of RDF triples, based on the data sources mentioned in Section 2.2. Each RDF triple consists of a subject, a predicate, and an object. Thereby, the subjects and predicates are IRIs, whereas the object is either an IRI or a literal. The notion of IRI (International Resource Identifier) is a generalization of URI (Uniform Resource Identifier), allowing non-ASCII characters to be used in the IRI character string [24]. Throughout this work, the URI prefixes commonly used within the Linked Data community are applied[7].

The DBpedia dataset has several advantages compared to the raw data from Wikipedia for the entity linking task. Besides the more convenient access to the underlying Wikipedia data sources, it also provides a higher data quality. The higher quality is due to the fact that the extraction framework embeds multiple steps commonly found in data mining applications, such as duplicate removal in the form of mapping redundant infobox properties to the same DBpedia property. Therefore, the present work is focussed on using DBpedia for entity linking to Wikipedia.

The entire DBpedia is split up into different datasets, which can be downloaded in different format such as N-Triples[8]. Since the goal of the present thesis is the creation of a graph based on DBpedia suitable for entity linking, in the following the datasets that are created as part of the extraction are discussed. Thereby, the datasets are grouped according to their Wikipedia data sources explained in Section 2.2.

**Article Text** There are various datasets containing elements of the article text, such as the titles[9] and short abstracts[10].

**Page Links** The page links, extracted from the internal links between Wikipedia articles, are also accessible as dataset in DBpedia[11].

[7] http://prefix.cc/

[8] http://wiki.dbpedia.org/Downloads39

[9] http://wiki.dbpedia.org/Downloads39#titles

[10] http://wiki.dbpedia.org/Downloads39#short-abstracts

[11] http://wiki.dbpedia.org/Downloads39#wikipedia-pagelinks

**Infobox Templates** The datasets extracted from the infobox templates represent the key facts about an entity. There are two types of information that are extracted from the templates, the type of template that is used for the article and the properties that are used within the template.

The name of the template itself is used to map the overall article towards an ontology type. The resulting *mapping-based types* dataset, also referred to as *instance-types*, contains *rdf:type* statements about DBpedia entities[12]. Hereby, the entities are linked to ontology classes, which are part of the overall DBpedia ontology[13]. DBpedia maintains a set of manual mappings between the templates in the template hierarchy and the appropriate ontology class. The ontology classes are arranged in a tree and range from broad classes such as *dbo:Place* and *dbo:Person* to fine-grained classes such as *dbo:Guitarist*[14]. For interlinking the ontology with other Linked Open Data datasets, the datasets also contain mappings to other respective classes, such as *foaf:Person*[15]. As an example, in the current version of DBpedia, *dbr:Michael_Jordan* is mapped as *dbo:BasketballPlayer*. Consequently, he is also mapped towards all classes from this class up to the root class, which makes him an *dbo:Athlete*, *dbo:Person*, *dbo:Agent*, and *owl#Thing*.

The properties of the infobox templates are extracted into various infobox datasets. These infobox datasets deal to a different extent with the inconsistencies and errors in infobox templates. The raw infobox properties[16] dataset, which is in the */property/* namespace, maps almost all infobox properties to their matching DBpedia property. Thus, this dataset has a high coverage but is less clean and more error prone. On the opposite, the *mapping-based properties* dataset[17], which is in the */ontology/* namespace, has a much higher quality, since different Wikipedia infobox properties that correspond to the same concept, are mapped towards one relation in the ontology. For the entity *dbr:Michael_Jordan*, an exemplary mapping-based triple is:

*dbr:Michael_Jordan dbo:birthPlace dbr:Brooklyn.*

Additional to the mentioned infobox derived datasets, there are domain-specific datasets such as *person data*[18] or *homepages*[19].

**Categorisation Information** DBpedia also contains datasets regarding the

[12] http://wiki.dbpedia.org/Downloads39#mapping-based-types
[13] http://wiki.dbpedia.org/Ontology39
[14] http://mappings.dbpedia.org/server/ontology/classes/
[15] http://xmlns.com/foaf/0.1/Person
[16] http://wiki.dbpedia.org/Downloads39#raw-infobox-properties
[17] http://wiki.dbpedia.org/Downloads39#mapping-based-properties
[18] http://wiki.dbpedia.org/Downloads39#persondata
[19] http://wiki.dbpedia.org/Downloads39#homepages

Wikipedia category system. Overall, DBpedia lists 995,911[20] different categories in their newest release 3.9 at the time of writing. The category URIs all share the prefix *http://dbpedia.org/resource/Category:*, which is commonly abbreviated as *category:*. The category information is split up into three different datasets, the *article categories*[21], the *skos categories*[22], and the *category labels*[23]. The article category dataset contains category statements about articles, using *http://purl.org/dc/terms/subject* as triple *predicate*. The skos category dataset contains the category hierarchy represented as statements with categories as subject and object. For the entity *dbr:Michael_Jordan*, DBpedia lists 38 categories, ranging from *category:Basketball_players_from_New_York* to very broad categories such as *category:Living_people*.

## 2.3 Phrase Spotting

*Phrase spotting* involves identifying the terms and phrases from which the links should be made [31]. Throughout research, these terms or phrases are called *spots*, *mentions*, or *surface forms*. There are two categories of spotting techniques which can be distinguished based on their data sources: spotting based on *Link probability*, which utilizes Wikipedia article texts and page links as a data source, and *NLP-informed* spotting, which uses the characteristics of tokens and phrases or pre-built language-dependent models. The remainder of this chapter discusses both categories in dedicated sections. Since DBpedia Spotlight makes extensive usage of both categories, their interplay as used in this system is discussed throughout the sections.

### 2.3.1 Link Probability

Throughout literature, the most common approach to spotting is called *link probability* or *annotation probability* detection. This approach has first been proposed by Mihalcea and Csomai in 2007 using the term *keyphraseness* [29]. Formally, the link probability of a phrase is defined as the number of Wikipedia articles that use it as an anchor, divided by the number of articles that mention it at all. Therefore, is is solely based on the article text (cf. Section 2.2.1) and page links (cf. Section 2.2.1) data sources. The detection approach works by gathering all n-grams for a document and retaining those surface forms whose link probability exceeds a certain threshold. When link probability is used as the only spotting method, overlapping surface form candidates can then be resolved based on the higher probability value.

[20] Unique categories in the Category (Labels) dataset (http://wiki.dbpedia.org/Downloads39#categories-labels)

[21] http://wiki.dbpedia.org/Downloads39#articles-categories

[22] http://wiki.dbpedia.org/Downloads39#categories-skos

[23] http://wiki.dbpedia.org/Downloads39#categories-labels

| Spot | Annotation Count | Total Count | Link Probability |
|---|---|---|---|
| *Napoleon* | 9,206 | 18,330 | 50.22% |
| *Waterloo* | 4,301 | 7,541 | 57.03% |
| *Wellington* | 11,994 | 19,278 | 62.22% |

Table 2.1: Link Probability Counts for Napoleon Sentence

For the following sentence, the respective annotation and total counts, generated by DBpedia Spotlight, are listed in Table 2.1:

*Napoleon was defeated at Waterloo by Wellington.*

Apart from the spots *Napoleon*, *Waterloo*, and *Wellington*, there are no n-grams within these sentences that have been used within an anchor in a Wikipedia article. Therefore, there are no further potential spots.

The link probability measure is unreliable for marginal cases where the total occurrence count is very low, since it only considers the relative frequency of phrases within anchor texts. Therefore, most entity linkers define a minimum total occurrence frequency threshold $t_{occ}$ for a phrase to be considered, such as $t_{occ} = 5$ as used by [29].

### 2.3.2 NLP-informed Spotting

To improve the accuracy of the *link probability* method, follow-up works experimented with incorporating natural language processing (NLP) informed methods. There are various methods based on NLP features.

A simple method for identifying further noun phrases is to determine *capitalized token sequences* such as "President Barack Obama" [11]. Another approach to improve spotting precision is to filter out phrases with common words based on a predefined set of stopwords, to avoid linking plain English words [27].

Apart from these relatively simple techniques, more advanced NLP techniques such as *part-of-speech (POS) tagging*, *phrase chunking*, or *named entity recognition (NER)*, can be applied [23]. These techniques require a prebuilt language-dependent model which has been trained on a large text corpora. Moreover, the tokenization of the input text is a prerequisite for these techniques to function properly. *POS tagging* marks tokens with their corresponding word type based on the token itself and the context of the token. There is an established set of word types commonly used throughout research, ranging from plural nouns (NNS) and adverbs (RB) to sentence closers. A *phrase chunker* uses these POS-tagged sentences to separate and segment a sentence into phrases such as noun, verb, and prepositional phrases. The phrase chunks that DBpedia Spotlight considers relevant for spotting are Noun Phrase chunks (NP), Prepositional Phrase chunks (PP) and Multi Word Units (MWUs) [11]. Apart from phrase chunking, *named entity recognition (NER)* is another NLP technique used for spotting. NER identifies and classifies phrases to types such as *People*, *Location*, or *Organization* [27].

Overall, DBpedia Spotlight combines link probability, capitalized token sequences detection, common phrase filtering, phrase chunking and named entity recognition in a two-step spotting process. In the first step, each of the mentioned methods generates a set of surface form candidates, forming a global set union of candidates. In the second step, the best candidates are selected from this set union. To this end, overlaps in the candidates are resolved based on which method generated them. Hereby, the more fine-grained and scientific approaches, such as NER, are preferred to methods such as plain capitalized token sequences detection. As the last step of the spotting process, the candidates whose link probability falls below a constant threshold value are dropped [11].

For the mentioned NLP tasks that require a pre-built model, DBpedia Spotlight relies on the open-source project *Apache OpenNLP*[24] [11].

## 2.4 Candidate Selection

Candidate generation is concerned with constructing a set of candidate entities for each spotted surface form. At the time of writing, most state-of-the-art systems rely on the approach proposed by the Cucerzan in 2007 [10]. Cucerzan uses four sources to extract possible surface forms for entities [10]:

**Article title** The article title is the most obvious surface form for an entity. For ambiguous surface forms, the article title contains a trailing appositive, either within parenthesis (e.g. *Napoleon_(Animal_Farm)*) or followed by a comma (e.g. *Napoleon,_Ohio*). These appositives are eliminated from the surface form representation.

**Redirect pages** The labels of the redirect pages, which have been discussed in Section 2.2.1, are all utilized as surface forms.

**Disambiguation Pages** The labels of the disambiguation pages (cf. Section 2.2.1) are also considered as surface forms for each listed entity. Thereby, the disambiguation suffix is stripped (e.g. *Michael_Jordan_(disambiguation)* $\rightarrow$ *Michael Jordan*).

**Page Links** As illustrated in Section 2.2.1, the anchor texts of the page links represent a rich set of surface forms for an entity.

Overall, the surface forms collected from these data sources can be seen as a community created set of commonly used labels, and thus, they have a low error rate. Based on the generated lexicon, which maps an entity to a set of surface forms, a reverse lookup can be created to map each surface form to a set of candidate entities.

DBpedia Spotlight follows the approach of Cucerzan (2007) by utilizing the respective datasets in DBpedia corresponding to the mentioned four data sources [28].

[24] http://opennlp.apache.org/

## 2.5 Disambiguation

The disambiguation phase, briefly described in Section 2.1, is concerned with determining the correct entity for each surface form, given a set of candidate entities.

There is a variety of features used for disambiguation within research. Therefore, the next section discusses all features used within state-of-the-art entity linking to Wikipedia systems (Section 2.5.1). Since, apart from the features themselves, the feature combination techniques are also crucial to the disambiguation accuracy, this section is followed by a discussion of selected state-of-the-art systems (Section 2.5.2).

### 2.5.1 Features

Based on the data sources drawn from Wikipedia (cf. Section 2.2), researchers have proposed various features that aid in linking the correct entity to a surface form. Throughout this section, it is distinguished between local and global features for entity linking, as proposed by [34]. Therefore the remainder of the section discusses all local features that have been utilized in state-of-the-art systems, followed by a discussion of the respective global features.

#### Local Features

Local approaches to disambiguation link each surface form to an entity, independent from the assignment of other surface forms to entities. There are three different categories of local features commonly used: *context similarity*, *prior probability*, and *popularity*. Each feature is described in the following:

- *Context Similarity*: The context similarity describes the similarity between a candidate entity in a document and its surrounding words. There are many ways on how to construct a context similarity feature, which can be based on the article text (cf. Section 2.2.1), the page links (cf. Section 2.2.1) or the categorisation information (cf. Section 2.2.1) data sources.

  Context similarity was already incorporated within early works on named entity disambiguation in 2006 [6]. In their work, the authors experiment with two different approaches. The first approach is based on a context-article cosine similarity, which compares the words within the sized window of the query document context and the article text. The latter approach uses a support vector machine (SVM) based on a taxonomy kernel using the Wikipedia categories. Further improvements to these approaches were achieved by [10].

  In one of the first works targeted at entity linking, Mihalcea & Csomai's (2007) best performing approach extracts different features from the phrase and its surrounding context [29]. To model feature vectors, the authors distinguish, among others, between a local context window of three words

around the ambiguous word and a global context. The global context is implemented through surface form-specific keywords, which are determined as a list of at most five words occurring at least three times in the contexts of a surface form. All feature vectors are then combined through a Naive Bayes classifier [29].

Later approaches to disambiguation also incorporate the context of page links to the bag-of-words representation for a context. To this end, Mendes et al. (2011) preprocessed Wikipedia articles and extracted the paragraph of each page link in Wikipedia [28]. Based on the article text, the page link paragraphs and the anchor texts, the authors generated a weighted bag-of-words representation for each entity.

Also in 2011, Ratinov et al. (2011) further experimented with different weighting schemes for the article and context [34].

Overall, the different approaches used within research indicate that there is a large set of options for modeling a context-based similarity between surface forms and entity candidates.

- *Prior Probability*: The prior probability, which is also known as commonness of an entity, is the most prominent feature in entity linking. This feature has been introduced in 2008 [25]. Formally, the prior probability $P(s|e)$ of an entity $e$ given a surface form $s$ is estimated as the fraction of times $e$ is the target page for $s$. This single feature is a quite reliable indicator of the correct disambiguation, and is therefore often used for baseline comparison. The importance of this feature is also reflected by the fact that it is contained within all state of the art entity linking systems discussed in Section 2.5.2.

  An example regarding prior probability is discussed in the subsequent Methods chapter, in Table 3.7.

- *Popularity* A global popularity feature of entities can also be exploited for disambiguation. To this end, Daiber et al. (2013) count the number of inlinks for each article in Wikipedia and normalize by the total number of links in Wikipedia [11]. Throughout research, this feature is not utilized frequently, since it does not perform as good as the previously discussed prior probability feature, which takes into account the surface form itself.

### Global Features

In contrast to local features, global features try to find an assignment of entities for a set of surface forms that maximizes the coherence among them. Throughout literature, global features are therefore often referred to as *entity coherence* or *relatedness*.

Ideally, global approaches to disambiguation define a *coherence function* that finds the best assignment of entities to surface forms within all combinations of

possible assignments. Since this global optimization problem is NP-hard, the common approach is the definition of a pairwise entity relatedness function and a heuristic that finds a satisfactory solution over a disambiguation context [34].

The *heuristic function* is commonly integrated with other local features into an overall disambiguation algorithm. Therefore, in the present work the heuristics are discussed in the same section that describes the overall disambiguation algorithms used by the state-of-the-art systems (cf. Section 2.5.2).

The notion of a *disambiguation context* has first been defined by [10] as the "union of disambiguation candidates for all the named entity mentions in the input document". However, follow-up approaches have reduced this context, since in large documents surface forms outside of the current paragraph are often unrelated and thus considered noise. Overall, there is no consensus of what the optimal disambiguation context looks like, since this also depends on the applied heuristic function. For instance, [31] use all unambiguous surface forms as context, whereas [12] use a window of five surface forms prior to and after the surface form in question.

Throughout research, there are two different categories of Wikipedia data sources that are exploited to estimate the pairwise relatedness between entities:

1. *Wikipedia page links*: This category is commonly referred to as *link overlap*, since it uses the overlap in the set of articles that link to each entity.

2. *Wikipedia infobox and category information*: The utilization of the infobox and category information results in a relatedness on a more semantic level; therefore, the latter category is depicted as the *semantic relatedness*.

The remainder of the section discusses both of these categories individually.

- *Link Overlap*: Link overlap has been proposed as the first global feature by Milne & Witten in 2008 [30]. It is estimated as the overlap between the inlinking articles of two entities. Milne & Witten (2008) use the *Normalized Google Distance (NGD)* for estimating the relatedness between two Wikipedia articles. Follow-up works also experimented with using in- and outlinks, and with using other measures such as *Pointwise Mutual Information (PMI)* [34].

  In the majority of the literature, the link overlap feature is referred to as *relatedness* (cf. [31, 34]) or *entity-entity coherence* (cf. [17]). Since other global features also estimate entity-entity coherence or relatedness, throughout this work the term *link overlap* is used to distinguish it from other global features.

  The first works on entity linking to Wikipedia all followed the approach of balancing the *commonness* (i.e. prior probability) of an entity with its relatedness to the surrounding context [31]. The techniques on how to combine these features are further addressed in Section 2.5.2.

- *Semantic Relatedness*: The semantic relatedness between entities can be estimated by exploiting the infobox (cf. Section 2.2.1) and category information (cf. Section 2.2.1) in Wikipedia.

  Throughout this work, the term *semantic relatedness* is used to refer to techniques that estimate the relatedness between entities using the mentioned data sources. This differs from the terminology in other works, where *semantic relatedness* is used when referring to link overlap (cf. [30]) or context-based similarity (cf. [15]). Moreover, there are works that use the term *semantic similarity* to refer to what the present work addresses as *semantic relatedness* (cf. [37]).

  Most works regarding semantic relatedness between Wikipedia concepts are not directly targeted at entity linking or word-sense disambiguation, but merely to estimate entity-entity relatedness. Strube & Ponzetto (2006) were the first to compute measures of semantic relatedness using Wikipedia [38]. For their approach - WikiRelate - they ported and modified techniques they had successfully applied on WordNet to Wikipedia, using the Wikipedia category structure. Their best performing approach uses a path-length measure adapted from [21]. A path-length measure has also been exploited by Schuhmacher & Ponzetto (2014) for the task of entity linking to Wikipedia [36]. Thereby, the authors determine the cheapest path between two entities, where the edges in the graph are weighted based on various information gain measures. Unlike most entity linking systems, that are targeted at natural text, the authors use their system for linking subject-predicate-object triples, obtained from an information extraction system.

  The LINDEN linker, proposed by [37], uses a semantic similarity feature that is also based on information gain. However, unlike [36], the authors estimate the relatedness based on a YAGO ontology tree distance measure. Thereby, for a pair of entities, the tree distance of their most similar $k$ super classes are determined, where the path distance is determined by weighting the edges based on information gain.

  Targeted at entity linking, the Illinois Wikifier 2.0 exploits direct connections between the candidates of the surface forms using statements from DBpedia [8]. This can be seen as connections in the DBpedia graph with a path length equal to one. They apply this feature on a very small disambiguation context of two surface forms that share a linguistic relation.

  The graph-based linking approach of the present thesis also fits into the category of semantic relatedness. Unlike the path-length measure approach, most disambiguation algorithms in the graph-based scenario are based on finding all paths within a maximum length to express the relatedness between two entities. This technique has been frequently used in the context of word-sense disambiguation, where typically the knowledge base is not based on Wikipedia, but rather on a lexicon with a linguistic focus, such as WordNet

[32]. However, for the task of entity linking, to the author's best knowledge AGDISTIS is the only system that exploits this methodology [41].

The usage of semantic relatedness in state of the art systems is further discussed in Section 2.5.2.

### 2.5.2 Systems

This section discusses selected state of the art systems in entity linking to Wikipedia.

The basis for the contemplated systems are the ones benchmarked in Cornolti et al. [9]. Addressing the changes that occurred since this publication, the new versions of DBpedia Spotlight (cf. [11]) and Illinois Wikifier (cf. [8]) have been added. Furthermore, the recently introduced systems AGDISTIS (cf. [41]) is also considered, since it uses an approach similar to the one in the present work.

Overall, the focus is placed on the features the systems use for disambiguation; thus, the system descriptions contain references to the features described in the previous section. A summary of the considered systems, along with the features they use, is given in Table 2.2. Hereby, except for the popularity and prior probability features, a short description is given on how each feature is used. For the mentioned two features the description is omitted since they are always applied in a straightforward manner.

The remainder discusses each considered entity linking system in a dedicated section.

#### DBpedia Spotlight

There are two major revisions of DBpedia Spotlight. The first major revision, which was mostly developed in 2011, solely relied on a context similarity feature (cf. Section 2.5.1) [28]. To this end, the authors built a vector space model (VSM) using Apache Lucene[25]. For comparing the context vectors, the authors developed a slight variation of the popular TF-IDF weighting scheme, by using an inverse candidate frequency (ICF) which weights the words in the context vector based on their overall frequency.

The second revision of DBpedia Spotlight uses a *generative probabilistic model* for disambiguation [11]. This model has been adapted from [16], with the central idea of combining the local features *popularity*, *prior probability* and *context similarity* in one statistical model. For the context score, the authors rely on a smoothed unigram language model, which uses the product of the maximum likelihood probabilities of all tokens in the context window.

#### Illinois Wikifier

The Illinois Wikifier has two different versions, one that has been published in 2011 (cf. [34]), and an improvement based on relational inference in 2013 (cf. [8]).

[25] http://lucene.apache.org/

| Feature Type | Local | | | Global | |
|---|---|---|---|---|---|
| Feature Name | Popularity | Prior probability | Context score | Link Overlap | Semantic Similarity |
| AGDISTIS | × | × | × | × | ✓<br>HITS on DBpedia Graph |
| AIDA | × | ✓ | ✓<br>keyphrase-based similarity | ✓<br>greedy iterative algorithm using NGD on inlinks | × |
| DBpedia Spotlight (2011) | × | × | ✓<br>TF-ICF vector space model | × | × |
| DBpedia Spotlight (2013) | ✓ | ✓ | ✓<br>smoothed unigram language model | × | × |
| Illinois Wikifier (2011) | × | ✓ | ✓<br>multiple weighted TF-IDF features | ✓<br>average/ max. PMI/ NGD on in-/ outlinks | × |
| Illinois Wikifier (2013) | × | ✓ | ✓<br>see Illinois Wikifer (2011) | ✓<br>see Illinois Wikifer (2011) | ✓<br>relation inlink and DBpedia predicate overlap |
| TAGME | × | ✓ | × | ✓<br>voting scheme using average NGD on inlinks | × |

Table 2.2: State of the Art Entity Linking Systems

In their first work from 2011, the authors experimented, aside from the common local features *popularity* and *prior probability*, with four different local *context similarity* and nine different global *link overlap* features. The context similarity features are different variants of weighted and unweighted cosine similarities between the document text, the entity article text and the entity context. For the global features, the authors use different variants of in- and outlink overlap, thereby measuring the Pointwise Mutual Information (PMI) and Normalized Google Distance (NGD)[33]. The linking algorithm, named Global Wikification (GLOW), combines all these features using a Support Vector Machine (SVM), where the coefficients $w_i$ are learned using training data from Wikipedia. The GLOW system works in a two-stage process, where at first a *ranker* obtains the best *non-null* disambiguation for each mention in the document, and then a *linker* decides whether the mention should be linked to Wikipedia, or whether instead switching the top-ranked disambiguation to *null* improves the global objective function [34].

In their follow-up work from 2013, the authors switched to an Integer Linear Programming (ILP) algorithm and incorporated linguistic features based on detecting relational inference [8]. The rules and patterns for extracting relations are proposed by the authors in a prior work [7]. In this work, the authors describe how to detect four different relational structures in text: *premodifier*, *possessive*, *preposition*, and *formulaic*. Besides the four mentioned relations, the current Wikifier version also detects *apposition* and *co-reference* relations within the text. The overall idea is that if such a relation can be detected, it is clear that the related surface forms should be disambiguated together, since they are closely connected. For such united disambiguations, the linker tries to find all direct connections between the candidates of the surface forms in consideration, using statements from DBpedia and direct page links. Since statements in DBpedia have a semantic meaning, the authors favour these explicit predicates by assigning them a higher weight than interconnections through page links.

### AIDA

The AIDA linker uses a weighted linear combination of local and global features as their overall objective function [17]. As local features, the author exploit the *prior probability* and *context similarity* between surface forms and entities, whereas the *link overlap* is applied as global feature. The authors also experiment with *semantic relatedness* features in the form of a syntax-based similarity. To this end, the overlap in YAGO types of the entities are compared. However, due to unsatisfactory results, this feature is not used in their best performing algorithm. Since the combination of all possible entity assignments is combinationally explosive, the AIDA system casts the entire optimization problem in a mention-entity graph. In this graph, the mention-entity and entity-entity edge weights are precomputed [42]. During the disambiguation phase, the greedy algorithm iteratively removes the worst candidate in the subgraph, until a final assignment is found.

### TAGME

The TAGME linker, introduced in 2010, uses a voting scheme algorithm based on *prior probability* and *link overlap* [13]. During the voting phase, each surface form assigns a combined score to each candidate entity of the other surface forms. For the ranking of of entity candidates, eventually a heuristic is adopted that selects the best link for each surface form, thereby dropping candidates whose prior probability missed a dedicated threshold. TAGME has been designed to deal with short texts; however, the evaluation by Cornolti et al. suggests that it also performs well on longer texts [9].

### AGDISTIS

AGDISTIS is the only linker surveyed in this work which also uses DBpedia graph traversal for disambiguation [41]. AGDISTIS is targeted at named entity recognition (NER); therefore, only candidates with an *rdf:type* matching a person, an organization, or a place, are considered. This leads to the following *rdf:type* restrictions:

- Person: *dbpedia-owl:Person*, *foaf:Person*
- Organization: *dbpedia-owl:Organization*, *dbpedia-owl:WrittenWork* (e.g. Journals)
- Place: *dbpedia-owl:Place*, *yago:YagoGeoEntity*

In the *candidate selection* phase (cf. Section 2.4), AGDISTIS only considers candidates with a surface form-label trigram similarity above a certain threshold $\sigma$. Here, $\sigma = 0.81$ has been found as the best threshold during evaluation. As entity label, the *rdfs:label* properties of the entities are utilized. This trigram similarity-based candidate pruning effectively removes a significant subset of the possible candidate entity set, and makes the linking of challenging cases, for example *His Airness* $\rightarrow$ *dbr:Michael_Jordan*, impossible.

For disambiguation, AGDISTIS applies a two step process. In the first step, a subgraph of the overall DBpedia graph is created by exploring the neighbor entities of all entity candidates up to a limited depth. After an experimentation phase the authors conclude that a maximum exploration depth of two results in the highest accuracy. In the second step, the *Hyperlink-Induced Topic Search (HITS)* link analysis algorithm is run on the subgraph created in the prior step. The candidates are then ranked based on their HITS score, and subsequently, the highest ranked candidate is returned.

# Chapter 3

# Methods

The methodology chapter is broken down into six sections. At first, methods are developed to construct and navigate the DBpedia dataset as a graph efficiently (Section 3.1). The following graph traversal section discusses how the subgraph for a given set of mentions and candidates is constructed (Section 3.2). Experiments in graph traversal revealed that it is necessary to restrict the set of candidate entities for a surface form. Thus, the traversal section is followed by a section dedicated to pruning the candidate set (Section 3.3). This is followed by a discussion of semantic relation weighting approaches (Section 3.4). To produce a disambiguation score for each candidate entity based on the subgraph, various graph-based connectivity measures are discussed in Section 3.5. Last, the methods for combining the graph-based linker with a traditional supervised linker are elaborated (Section 3.6).

## 3.1 Graph Construction

For constructing a navigable graph based on DBpedia, it is necessary to determine the subset of the overall DBpedia dataset that is relevant for entity linking. This step is crucial, since as much noise as possible needs to be eliminated, while still retaining all connections that are helpful for disambiguation.

For example, disambiguation pages are considered harmful for disambiguation, since they connect entities that might be otherwise unrelated. On an exemplary basis, the disambiguation page for the name Michael Jordan[1], lists, besides the famous basketball player, unrelated entities such as an Irish politician[2].

In the present work, the relevant DBpedia subset is determined based on a two level approach. On the macro level, completely irrelevant datasets are identified, which are excluded from the graph representation. On the micro level, irrelevant triples from the generally accepted datasets are filtered out. In the next two sections both levels are addressed individually.

[1] http://en.wikipedia.org/wiki/Michael_Jordan_(disambiguation)
[2] http://en.wikipedia.org/wiki/Michael_Jordan_(Irish_politician)

### 3.1.1 Selecting DBpedia Datasets

The entire DBpedia dataset is split up into multiple, individually accessible, datasets[3]. These datasets are formed into groups such as "Infobox Data" or "Classification"[4]. The different datasets are discussed in the following, including a justification for each dataset whether or not it is relevant for entity linking. Hereby, the datasets are grouped according to their Wikipedia data sources, as explained in Section 2.2.1:

- *Article Text*: Since only semantic connections between entities are considered relevant, the article text is excluded.
- *Page Links*: The page links could be exploited for graph-based entity linking to Wikipedia; however, the focus of the present work is to exploit the semantic interrelations between entities. Thus, this data source is not utilized.
- *Infobox Templates*: The datasets extracted from the infobox templates are the most interesting ones for graph-based entity linking, since they represent the key facts about an entity. Therefore, the following datasets are used, which have already been described in Section 2.2.2: *instance types*, *mapping-based properties*, and *persondata*.
- *Categorisation Information*: Generally, the category information can be considered as a helpful knowledge source in aiding the disambiguation task. Therefore, all datasets related to categories are considered: *article categories*, *skos categories*, and *topical concepts*. Hereby, the article categories dataset contains the actual statements about the categories of a resource. The *skos categories* dataset contains the interconnections among the categories. Moreover, the *topical concepts* contain, if applicable, the resource that matches each category, such as *category:Programming* and *dbr:Programming*.

Overall, the first step in determining relevant DBpedia datasets yields the set of valid datasets illustrated in Table 3.1. The *redirects* dataset is also included to maintain compatibility with DBpedia Spotlight, since the candidate entities that Spotlight produces are often redirects to actual entities.

The remaining excluded datasets are shown in Table 3.2. There are different reasons for exclusion: *Literals* means that all triples within the respective dataset contain literal objects, and thus, do not interconnect entities. *Internal irrelevant URIs* refers to the fact that the objects in the datasets are URIs in the DBpedia namespace; however, these URIs do not correspond to Wikipedia articles. *External URIs* depicts that the dataset is made up of triples with URI objects that are not in the DBpedia namespace; thus, they do no represent direct connections between entities.

[3] http://wiki.dbpedia.org/Downloads
[4] http://wiki.dbpedia.org/Datasets

| Dataset Type | Dataset Name |
|---|---|
| Infobox Data | Mapping-based Types<br>Mapping-based Properties<br>Persondata |
| Classifications | Articles Categories<br>Categories (Skos)<br>Topical Concepts |
| Other | Redirects |

Table 3.1: Included DBpedia Datasets

### 3.1.2 Triple Filtering

In the included datasets from the previous section, there are still many triples that are irrelevant for linking. Thus, during graph construction, it is necessary to inspect each individual RDF triple on an automated basis and determine whether it is relevant or not.

In this work, the applied methods for filtering out irrelevant triples in DBpedia are based on existing approaches from academia. For traversing DBpedia for topic labeling, [18] propose a blacklist of irrelevant URIs. This blacklist focusses on irrelevant categories; therefore, the authors created the list by navigating the higher levels of the category hierarchy rooted at the node *category:Contents*. As a result, the researchers define three categories of triples that are irrelevant [18]:

- *Wikipedia administrative categories*: This category consists of Wikipedia categories that solely exist for administration purposes, e.g. *Category:Pages_containing_deleted_templates*.
- *Etymology categories*: The articles belonging to etymology and its subcategories are devoted to a discussion of terminology, usually terminology related to a specific topic[5], e.g. *Category:Clothing_terminology*.
- *Generic LOD concepts*: Common concepts in the LOD cloud are considered irrelevant. In this category, [18] define five URIs: *owl:Thing*, *owl:Class*, *skos:core#Concept*, *http://www.opengis.net/gml/_Feature*, and *http://dbpedia.org/class/yago/Concept105835747*.

Overall, [18] define 852 blacklisted categories, arguing that these nodes create a range of shortcuts between concepts that do not reflect interesting relationships.

[36] further extended this blacklist for their approach of representing documents in a DBpedia graph. Since the entire DBpedia dataset has already been cut down to a smaller subset in the previous section, most filters of their work are not needed, since the respective URIs do not occur in any of the selected datasets. Overall, the only relevant filters are targeted at Wikipedia templates. These are

[5] http://en.wikipedia.org/wiki/Category:Terminology

| Dataset Type | Dataset Name | Exclude Reason |
|---|---|---|
| Basic Information | Titles | Literals |
| | Short Abstracts | Literals |
| | Extended Abstracts | Literals |
| | Images | Irrelevant internal URIs |
| | Links to Wikipedia Article | Irrelevant internal URIs |
| Classifications | Categories (Labels) | Literals |
| Infobox Data | Mapping-based Types (Heuristic) | Inferior to Mapping-based Types |
| | Mapping-based Properties (Cleaned) | Inferior to Mapping-based Types |
| | Mapping-based Properties (Specific) | Inferior to Mapping-based Types |
| | Raw Infobox Properties | Inferior to Mapping-based Types |
| | Raw Infobox Property Definitions | Inferior to Mapping-based Types |
| External Links | External Links | External URIs |
| | Homepages | External URIs |
| Geo-Coordinates | Geographic Coordinates | Literals |
| Other | Wikipedia Pagelinks | Non-semantic connections |
| | Transitive Redirects | Redundant to ”Redirects” |

Table 3.2: Excluded DBpedia Datasets

URIs starting with *http://dbpedia.org/resource/Template:*, which only occur in the *redirect* dataset as part of all selected datasets. Of the more than six million triples in the redirect dataset, there are 92,941 matching template triples. Although both blacklists are not targeted explicitly at the entity linking task, they still form a suitable basis for the domain of the present work.

After having identified appropriate datasets in the previous section and cleaning the datasets from irrelevant triples such as administrative categories through blacklist filtering, the next step is concerned with analyzing which statements are helpful indicators and which are hindering disambiguation. Therefore, additional filters in the form of blacklists and regular expressions are defined, with the goal of removing connections in the graph that are not helpful for disambiguating entities. This task has been separated according to the three types of included datasets: *properties*, *ontology*, and *category*. In the following, a detailed analysis about each of those types, along with the resulting additional filters, is given.

| Category | Frequency | Cumulative Percentage |
|---|---|---|
| Category:Living_people | 612308 | 3.69% |
| Category:Year_of_birth_missing_(living_people) | 47398 | 3.97% |
| Category:English-language_films | 26198 | 4.13% |
| Category:American_films | 18630 | 4.24% |
| Category:Main_Belt_asteroids | 17739 | 4.35% |
| Category:The_Football_League_players | 15471 | 4.44% |
| Category:Year_of_death_missing | 15236 | 4.54% |
| Category:Year_of_birth_missing | 14451 | 4.62% |
| Category:Black-and-white_films | 14261 | 4.71% |
| Category:English_footballers | 14253 | 4.79% |
| Category:Association_football_midfielders | 14011 | 4.88% |
| Category:1985_births | 11876 | 4.95% |
| Category:Association_football_defenders | 11804 | 5.02% |
| Category:Year_of_birth_unknown | 11779 | 5.09% |
| Category:1984_births | 11668 | 5.16% |
| Category:Association_football_forwards | 11643 | 5.23% |
| Category:1983_births | 11605 | 5.30% |
| Category:1986_births | 11592 | 5.37% |
| Category:1982_births | 11558 | 5.44% |
| Category:1981_births | 11400 | 5.51% |
| Category:Articles_created_via_the_Article_Wizard | 11346 | 5.58% |

Table 3.3: Top-20 Occurring Categories

### Categorisation

After removing administrative and ethimology related categories in the previous section, there are still very common categories such as *Category:Living_people*. Although these categories are no obvious noise, they are not helpful in entity linking, since they directly connect otherwise unrelated entities. Therefore, as a first step the distribution of categories needs to be analyzed.

The top 20 utilized categories are shown in Table 3.3. The distribution shows that, besides the frequently used category *Living_People*, there are many other categories that are not helpful in disambiguation, such as *Category:1985_births*. Moreover, there are still administrative categories that have not been captured by the blacklists imported from other work, such as *Category:Articles_created_via_the_Article_Wizard*. Therefore, a set of regular expressions was defined to further filter out irrelevant categories. These regular expressions, along with the number of affected categories and exemplary targeted categories, are shown in Table 3.4. All of these regular expressions are applied to the resource suffix following *Category:* in a case-insensitive manner. Due to space

| Regular Expression | Affected Categories | Top-3 Matched Categories by Frequency | |
|---|---|---|---|
| | | Category Name | Frequency |
| `^\d+.*$` | 103221 | 1985_births | 11876 |
| | | 1984_births | 11668 |
| | | 1983_births | 11605 |
| `^.*\d+$` | 34618 | Astronomical_objects_discovered_in_1998 | 1762 |
| | | Astronomical_objects_discovered_in_1999 | 1535 |
| | | Animals_described_in_1758 | 1495 |
| `^list.*` | 4991 | Lists_of_asteroids_by_number | 804 |
| | | Lists_of_American_television_series_episodes | 764 |
| | | Lists_of_drama_television_series_episodes | 676 |
| `^.*stub.*` | 919 | People_educated_at_Stubbington_House_School | 107 |
| | | Writing_system_stub_articles_needing_reassessment | 90 |
| | | Inorganic_compound_stubs | 77 |
| `^.*articles.*` | 736 | Articles_created_via_the_Article_Wizard | 11346 |
| | | Article_Feedback_5_Additional_Articles | 712 |
| | | Law_articles_needing_an_infobox | 493 |
| `^.*wiki.*` | 190 | WikiProject_Artemis_Fowl | 102 |
| | | Wikipedias_by_language | 97 |
| | | MediaWiki_websites | 93 |
| `^.*redirect.*` | 164 | Unprintworthy_redirects | 9394 |
| | | Redirects_from_songs | 4008 |
| | | Redirects_from_alternative_names | 1576 |
| `^.*pages.*` | 100 | Place_name_disambiguation_pages | 1415 |
| | | Township_name_disambiguation_pages | 1008 |
| | | Human_name_disambiguation_pages | 737 |
| `^.*disambiguation.*` | 74 | Place_name_disambiguation_pages | 1415 |
| | | Township_name_disambiguation_pages | 1008 |
| | | Human_name_disambiguation_pages | 737 |
| `^.*portal.*` | 38 | Web_portals | 132 |
| | | Sports_and_games_current_events_portals | 103 |
| | | Science_portal | 22 |
| `^.*categories.*` | 20 | Grammy_Award_categories | 114 |
| | | Motorsport_categories_in_Australia | 42 |
| | | Monoidal_categories | 32 |
| `^.*infoboxes.*` | 17 | Supreme_Court_of_Canada_case_articles_without_infoboxes | 88 |
| | | Canadian_articles_needing_infoboxes | 31 |
| | | United_States_Supreme_Court_case_articles_without_infoboxes | 6 |
| `^.*templates.*` | 13 | Musician_templates | 4 |
| | | WikiProject_United_States_Public_Policy_templates | 2 |
| | | Wikipedia_Education_Program_templates | 2 |
| `^.*navigational_boxes.*` | 12 | Telugu_film_director_navigational_boxes | 4 |
| | | Tamil_film_director_navigational_boxes | 2 |
| | | Nevada_county_navigational_boxes | 2 |

Table 3.4: Regular Expressions for filtering Wikipedia Categories

constraints, the *Category:* prefix has been omitted. Besides these regular expressions, the category *Living_people* was also blacklisted since it is too widespread and provides little additional value in disambiguation.

Due to all established filters discussed in this section, of the 753,525 categories that occur at least once, 608,412 (80.74%) are included. Of the 16,599,811 total article categorisation triples, applying the mentioned filters yields 12,772,300 (76.94%) valid. Moreover, 66,698 (45.87%) out of 145,398 topical concept triples are allowed. Among the interconnections between the categories, which are contained in the skos concepts dataset, 1,566,904 (39.05%) out of 4,012,746 triples are included into the graph. Overall, this results in 14,405,902 categorisation related triples in the concluding DBpedia graph.

### Properties

Overall, there are 1373 unique properties in the English Wikipedia as captured in the DBpedia 3.9 version[6]. [41] performed some initial experiments to determine the influence of single properties on the linking accuracy. As methodology, the authors removed single properties from the knowledge base and then investigated the delta of the overall accuracy. Due to the limited scope of the thesis, finding a good estimator for the properties influence on disambiguation is not possible. Therefore, rather generic estimators based on occurrence frequency are applied, as described in Section 3.4.

Of the overall 25,910,643 mapping-based property triples, 13,096,087 (50.54%) are imported into the final DBpedia graph. The high number of rejects is due to the fact that 12,060,592 (46.55%) triples contain literal objects, which do not interconnect entities. The remaining 2.91% rejects are due to the filters applied from mentioned prior works.

### Ontology

At the time of writing, the current DBpedia version 3.9 has 529 different ontology classes[7]. As mentioned in the dedicated section on DBpedia (cf. Section 2.2.2), the ontology classes are organized as a tree, with *owl:Thing* as root class[8]. The first level below the root class are generic classes such as *owl:Place*, *owl:Person*, *owl:Work*, *owl:Species*, and *owl:Organisation*. These generic classes provide little value to disambiguation.

To decrease the influence of those widespread classes, it would be possible to introduce weighted edges to degrade common Linked Open Data (LOD) concepts such as high-level ontology classes. However, this results in a computationally explosive graph traversal. Even if the graph is only explored with a maximum depth of two, when using undirected traversal and following a *rdf:type owl:Thing* link, in the next step all vertices within the entire graph would be traversed, since each vertex is mapped to the root ontology class. Therefore, it is essential to filter out common LOD concepts that do not help in entity linking.

Overall, there are 15,894,066 triples in the mapping-based types dataset. As a first step, the occurrence frequency of the different ontology classes needs to be analyzed.

The 20 classes with the highest occurrence frequency are displayed in Table 3.5. As expected, the distribution shows that the root ontology class *owl#Thing* has by far the highest occurrence frequency. This is followed by other high level ontology classes such as *owl:Agent*, *owl:Person*, or *owl:Place*. There are many ontology classes without the common prefix *http://dbpedia.org/ontology/*, such as *foaf:Person*. In these cases, a corresponding class with the DBpedia prefix can be

[6] http://wiki.dbpedia.org/Datasets39/DatasetStatistics
[7] http://wiki.dbpedia.org/Ontology39
[8] http://mappings.dbpedia.org/server/ontology/classes

| URI in Mapping-based Types | Frequency | Percentage | Cumulative Percentage |
|---|---|---|---|
| http://www.w3.org/2002/07/owl#Thing | 3221405 | 20.27% | 20.27% |
| http://dbpedia.org/ontology/Agent | 1041029 | 6.55% | 26.82% |
| http://xmlns.com/foaf/0.1/Person | 831558 | 5.23% | 32.05% |
| http://schema.org/Person | 831558 | 5.23% | 37.28% |
| http://dbpedia.org/ontology/Person | 831558 | 5.23% | 42.51% |
| http://schema.org/Place | 639450 | 4.02% | 46.54% |
| http://dbpedia.org/ontology/Place | 639450 | 4.02% | 50.56% |
| http://dbpedia.org/ontology/CareerStation | 577196 | 3.63% | 54.19% |
| http://dbpedia.org/ontology/PopulatedPlace | 427068 | 2.69% | 56.88% |
| http://dbpedia.org/ontology/Settlement | 395655 | 2.49% | 59.37% |
| http://schema.org/CreativeWork | 372226 | 2.34% | 61.71% |
| http://dbpedia.org/ontology/Work | 372226 | 2.34% | 64.05% |
| http://dbpedia.org/ontology/SportsTeamMember | 253901 | 1.60% | 65.65% |
| http://dbpedia.org/ontology/OrganisationMember | 253901 | 1.60% | 67.25% |
| http://dbpedia.org/ontology/Athlete | 232082 | 1.46% | 68.71% |
| http://dbpedia.org/ontology/Species | 225587 | 1.42% | 70.13% |
| http://dbpedia.org/ontology/Eukaryote | 222093 | 1.40% | 71.52% |
| http://schema.org/Organization | 209471 | 1.32% | 72.84% |
| http://dbpedia.org/ontology/Organisation | 209471 | 1.32% | 74.16% |
| http://dbpedia.org/ontology/MusicalWork | 166520 | 1.05% | 75.21% |
| http://dbpedia.org/ontology/Animal | 165476 | 1.04% | 76.25% |
| http://dbpedia.org/ontology/ArchitecturalStructure | 132479 | 0.83% | 77.08% |
| http://dbpedia.org/ontology/Village | 119860 | 0.75% | 77.84% |
| http://schema.org/MusicAlbum | 116371 | 0.73% | 78.57% |
| http://dbpedia.org/ontology/Album | 116371 | 0.73% | 79.30% |
| http://dbpedia.org/ontology/PersonFunction | 104894 | 0.66% | 79.96% |
| http://dbpedia.org/ontology/SoccerPlayer | 89078 | 0.56% | 80.52% |
| http://schema.org/Movie | 77769 | 0.49% | 81.01% |
| http://dbpedia.org/ontology/Film | 77769 | 0.49% | 81.50% |
| http://dbpedia.org/ontology/Insect | 75420 | 0.47% | 81.97% |
| http://dbpedia.org/ontology/Artist | 68237 | 0.43% | 82.40% |
| http://dbpedia.org/ontology/Building | 67287 | 0.42% | 82.83% |
| http://schema.org/MusicGroup | 66618 | 0.42% | 83.25% |
| http://dbpedia.org/ontology/Infrastructure | 63436 | 0.40% | 83.64% |
| http://dbpedia.org/ontology/TimePeriod | 62897 | 0.40% | 84.04% |
| http://dbpedia.org/ontology/NaturalPlace | 50719 | 0.32% | 84.36% |
| http://dbpedia.org/ontology/WrittenWork | 49833 | 0.31% | 84.67% |
| http://schema.org/Product | 49788 | 0.31% | 84.99% |
| http://dbpedia.org/ontology/Company | 49402 | 0.31% | 85.30% |
| http://dbpedia.org/ontology/Plant | 47011 | 0.30% | 85.59% |

Table 3.5: Top-20 Mapping-based Types by Frequency

found (e.g. *owl:Person*). Since this information is redundant, all ontology classes not in the DBpedia ontology namespace are filtered out.

The distribution also shows, that the depth of the ontology class within the ontology hierarchy is not a good indicator for the frequency of occurrence. For example, the *owl:Species* class, which is located right below the root class, occurs less often than *owl:SportsTeamMember*, which is located at depth five (*owl#Thing* $\rightarrow$ *owl:Agent* $\rightarrow$ *owl:Person* $\rightarrow$ *owl:OrganisationMember* $\rightarrow$ *owl:SportsTeamMember*). Therefore, for the task of entity linking, it is not sufficient to introduce an ontology hierarchy depth threshold and merely include all classes that are deeper than the threshold. Therefore, in the present work it has been opted for the approach of introducing a occurrence frequency threshold $t$. To this end, all ontology classes that occur less often than $t$ are included.

With the setting of $t := 100000$, as a result of the threshold and only including ontology classes in the DBpedia namespace, 2,545,913 (16.02%) out of the 15,894,066 triples are imported into the graph.

## 3.2 Graph Traversal

The graph traversal aims at finding connections between candidate entities of surface forms. This step serves as a prerequisite to the subsequent application of various graph-based connectivity measures. The traversal algorithm in this work is based on the work by [32].

Formally, the traversal algorithm is defined as follows: Given a set of surface forms $S = \{s_1, s_2, \ldots, s_n\}$ within a document and a maximum path-length of $L$. For each surface form $s_i$, let $E_i = \{e_{i1}, e_{i2}, \ldots, e_{in}\}$ be the set of candidate entities for $s_i$. The goal of our the traversal algorithm is to find for each candidate entity $e_{ij}$ all paths $p = (e_{ij}, e_1, \ldots, e_n, e_{lk})$ to entities $e_{lk}$, with $i \neq l$ and path length $l_p \leq L$. Thus, only paths from candidate entities of different surface forms are allowed. Let $P_{ij} = \bigcup_k p_{ijk}$ be the union set of all paths for a candidate entity $e_{ij}$. The set union of all candidate entities $E = \bigcup_i E_i$ combined with the vertices and edges of all paths of all entities $P = \bigcup_{i,j} P_{ij}$ results in the subgraph $G_s = E \cup P$, with $G_s \subseteq G$, of the overall DBpedia Graph $G$.

For traversal, a limited-depth depth-first-search (DFS) is applied. Limited breadth-first-search (BFS) could also be applied; however, DFS is preferable due to the lower memory usage.

There are different parameters to influence the traversal algorithm and thus the resulting subgraph:

**Traversal Depth** The maximum path length $L$ in which connections are searched highly influences the resulting subgraph. [41] use a maximum distance of two for their best-performing graph-based algorithm. In the present work, depending on the traversal direction, various experiments with $L \in [1, 6]$ have been conducted.

| Traversal Direction | Traversal Depth | Subgraph Vertices | Subgraph Edges | Traversed Nodes | Elapsed Time 3rd run [sec] |
|---|---|---|---|---|---|
| Directed | 1 | 9 | 2 | 227 | 0.09 |
| Directed | 2 | 10 | 5 | 1,835 | 1.32 |
| Directed | 3 | 14 | 14 | 15,470 | 2.31 |
| Directed | 4 | 34 | 54 | 129,917 | 4.48 |
| Directed | 5 | 87 | 163 | 1,058,106 | 8.09 |
| Directed | 6 | 184 | 414 | 8,449,595 | 17.93 |
| Directed | 7 | 481 | 1176 | 68,701,341 | 73.82 |
| Undirected | 1 | 9 | 2 | 1,865 | 0.14 |
| Undirected | 2 | 15 | 15 | 907,475 | 2.77 |
| Undirected | 3 | 277 | 1,068 | 110,006,847 | 109.83 |

Table 3.6: Subgraph comparison for sample sentence

**Traversal Direction** For finding connections, the edges as RDF triples can either be traversed in a directed or in an undirected manner. In directed traversal, the RDF triple $(subject, predicate, object)$ results in the edge $(subject, object)$, whereas in undirected traversal, the mentioned triple results in an unoriented edge $\{subject, object\}$, with $(subject, object) = (object, subject)$.

The traversal direction parameter highly influences the overall traversal process, especially regarding the ontology statements. Since DBpedia ontology classes exclusively occur as object in RDF triples, the have a high indegree in the resulting graph, but outgoing edges are restricted to other ontology classes. Therefore, when using directed traversal, there are no paths between entities with intermediate ontology vertices. Thus, when opting for directed traversal, the ontology dataset is implicitly ignored.

The opposite issue arises for undirected traversal. As mentioned in the previous section on selecting DBpedia datasets (cf. Section 3.1.1), when using undirected traversal it is critical not to traverse over very common ontology classes or categories, since this yields a vast set of connections between otherwise unrelated entities.

To illustrate the effect of graph traversal parameters on the resulting subgraph, the following exemplary sentence is used:

*Napoleon was defeated at Waterloo by Wellington.*

The surface forms in this text are *Napoleon*, *Waterloo*, and *Wellington*. The resulting subgraph metrics for different settings in traversal direction and depth are shown in Table 3.6. Hereby, the utilized DBpedia graph includes the filtered categories and ontologies with a frequency $\leq 100000$. Including the full set of candi-

dates would result in a very large subgraph, unsuited for illustration. Thus, only the top-3 candidates based on prior probability are used. The necessity of pruning the set of candidate entities is further discussed in Section 3.3. For undirected traversal with a maximum depth of two, the resulting subgraph is displayed in Figure 3.1.

As shown in the mentioned Table 3.6, even for a tiny example with three surface forms, the subgraph size exponentially increases with the traversal depth, especially for undirected traversal. Based on this initial experiment, the viable traversal depth $d$ range for directed traversal is considered to be equal $d \in [1, 6]$, and for undirected traversal $d \in [1, 2]$.

## 3.3 Candidate Pruning

Initial experiments revealed that the number of candidates for an entity can be very high. This can be illustrated by the exemplary napoleon sentence from the previous section. Table 3.7 shows the top-3 candidates ranked by prior probability. The table includes the total number of candidates for each surface form if no candidate pruning is applied.

| Surface Form | Candidate Entity | Prior Probability | Support | Number of Candidates |
|---|---|---|---|---|
| Napoleon | Napoleon | 0.9636 | 8871 | 43 |
| | Napoleon_(Animal_Farm) | 0.0095 | 87 | |
| | Napoleon,_Ohio | 0.0052 | 48 | |
| Waterloo | Waterloo,_Ontario | 0.2587 | 1043 | 88 |
| | Battle_of_Waterloo | 0.1511 | 609 | |
| | London_Waterloo_station | 0.0839 | 338 | |
| Wellington | Wellington | 0.6512 | 7811 | 116 |
| | Arthur_Wellesley,_1st_Duke_of_Wellington | 0.0472 | 566 | |
| | Wellington_Rugby_Football_Union | 0.0437 | 525 | |

Table 3.7: Top-3 candidates for sample sentence surface forms

The high number of candidates for *Wellington* and *Waterloo* are due to the fact that there are numerous villages and cities with these names; moreover, there are articles for various entities related to these cities, ranging from buildings to sports teams. For Napoleon, the candidates range from games (e.g. *dbr:Napoleon_(game)*) to a rapper called Napoleon (e.g. *dbr:Napoleon_(rapper)*).

Including the entire set of candidates has several drawbacks. Apart from the computational complexity, the large set of candidates makes the subgraphs impossible to analyze. Moreover, a vast set of candidates increases the likelihood of finding noisy paths that have not been detected in the prior step. This assumption was strengthened by initial experiments, which show an increase in overall accuracy by

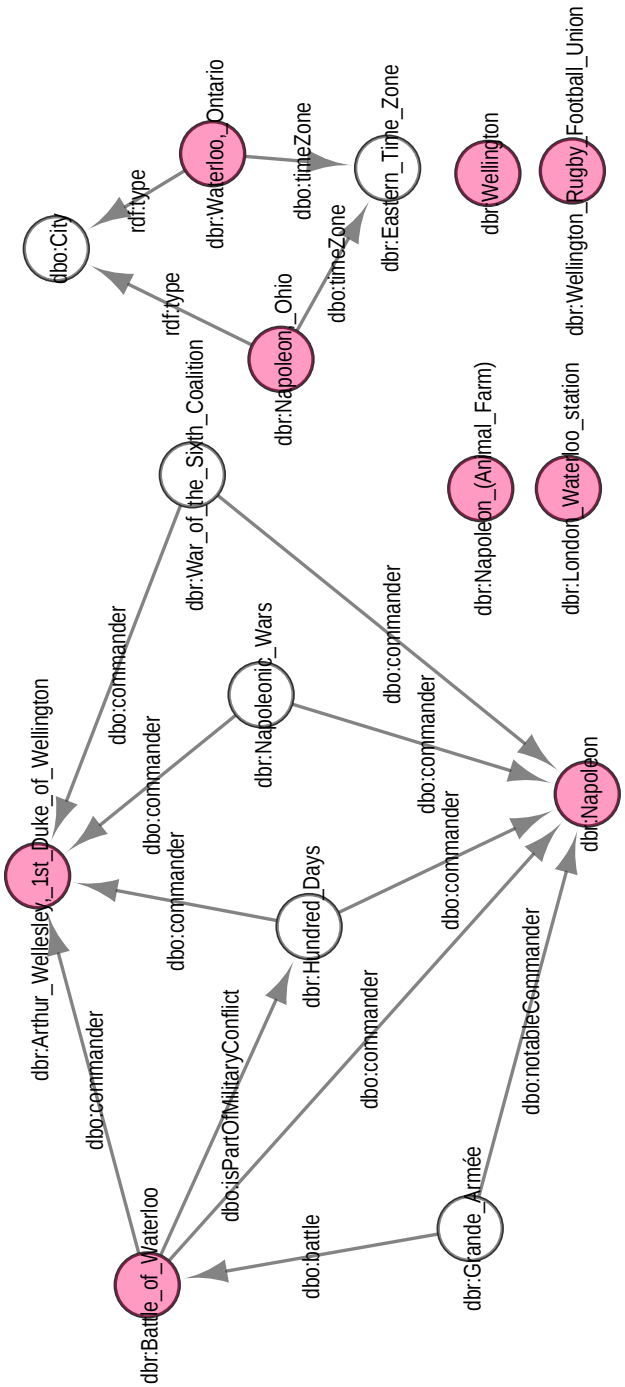


Figure 3.1: Sentence Subgraph for Undirected Traversal with Depth Two

pruning the set of candidates. Therefore, experiments with different approaches to reduce the set of candidates for a surface form are necessary. There are different measures that can be applied for ranking candidates prior to disambiguation. In the present case, the most important a priori knowledge about an entity is the *support* of an entity and its *prior probability*. For each of those two measures, experiments with two different techniques were conducted:

1. Keeping only candidate entities whose *support*/*prior probability* is higher than a minimum threshold $t$.

2. Keeping the best $k$ candidate entities based on their *support*/*prior probability* value.

Of those four approaches, removing candidates that do not fulfill the minimum support requirement performed worst. This is due to the fact that in the gold standard datasets there are some annotations where the correct entity has a very small support, especially annotations to people. For example, a text about cricket annotates the relatively unknown player *dbr:Bruce_Pairaudeau*, whose support equals four. The mentioned inflexibility is also present when removing candidates below a minimum prior probability. When candidates are removed based on a threshold, the likelihood of not having a single candidate for a surface form increases. These no-annotations always equal an incorrect annotation when the surface forms in the text are already provided during evaluation. Thus, in this evaluation setup it is crucial that a linker tries to always find an annotation.

Overall, keeping the best-$k$ candidates based on prior probability showed the best results. The exact number of how to assign $k$ highly depends on the dataset, and in the conducted experiments $k$ yields best results in the range of $[3, 10]$. Naturally, the lower the value of $k$, the higher the likelihood that the correct entity is not within the kept subset.

To quantify this likelihood, Table 3.8 shows the distribution of the gold standard entity in the overall set of candidates, ranked by prior probability. The table can be interpreted as follows for Rank $i = 2$:

- In 12.08% of all cases within the AIDA/CO-NLL dataset, the correctly annotated entity has the second highest prior probability within all candidate entities. Thereby, the set of candidate entities is generated using the candidate selection method discussed in Section 2.4.

- The respective cumulated probability value means that in 84.55% of all cases, the correctly annotated entity is among the two entities with the highest prior probability.

The highest rank where a gold standard annotation can be found equals 309. Within a smaller or equal rank, the gold standard annotation can be found in 97.29% of all cases. The missing 2.71% reflect errors of the *candidate selection*

| Prior Probability Rank $i$ | Probability of Gold Standard Annotation at Rank $i$ | Cumulated Probability of Gold Standard Annotation at Rank $\leq i$ |
|---|---|---|
| 1 | 72.47% | 72.47% |
| 2 | 12.08% | 84.55% |
| 3 | 3.31% | 87.86% |
| 4 | 1.61% | 89.47% |
| 5 | 0.76% | 90.23% |
| 6 | 0.90% | 91.13% |
| 7 | 0.83% | 91.95% |
| 8 | 0.61% | 92.57% |
| 9 | 0.46% | 93.03% |
| 10 | 0.35% | 93.37% |
| 11 | 0.20% | 93.57% |
| 12 | 0.21% | 93.78% |
| 13 | 0.18% | 93.96% |
| 14 | 0.23% | 94.19% |
| 15 | 0.10% | 94.29% |
| 16 | 0.24% | 94.53% |
| 17 | 0.06% | 94.59% |
| 18 | 0.09% | 94.68% |
| 19 | 0.13% | 94.81% |
| 20 | 0.06% | 94.87% |
| ... | ... | ... |
| 309 | 0.02% | 97.29% |

Table 3.8: Candidate Prior Probability Distribution AIDA/CO-NLL

method, which means that in these cases the gold standard entity is not part of the candidate set.

For choosing an appropriate cut-off $k$, the needed rank to receive coverage for reference values such as 85%, 90%, and 95% can be considered. To achieve 85%/90%/95% coverage, $k$ needs to be higher or equal than 3/5/21. The large gap in rank between 90% and 95% shows that 95% is not reachable while stile maintaining a relatively low noise level and acceptable computational performance. Thus, setting $k$ in the range $[3, 10]$, as determined by initial experiments, is supported by the data from the distribution.

## 3.4 Semantic Relation Weighting

The relations in DBpedia differ in their importance to disambiguation. A relation such as *<dbr:Steffi_Graf rdf:type owl:Thing>* has a lot less importance

| Predicate | Object | $joint\ IC$ | $comb\ IC$ | $IC+\ PMI$ |
|---|---|---|---|---|
| dbo:battle | dbr:Battle_of_Waterloo | 13.62 | 19.60 | 5.98 |
| dbo:commander | dbr:Arthur_Wellesley,_1st _Duke_of_Wellington | 13.76 | 19.64 | 5.87 |
| dbo:commander | dbr:Napoleon | 12.98 | 19.28 | 6.29 |
| dbo:isPartOfMilitaryConflict | dbr:Hundred_Days | 15.82 | 21.83 | 6.01 |
| dbo:notableCommander | dbr:Napoleon | 16.73 | 21.29 | 4.56 |
| dbo:timeZone | dbr:Eastern_Time_Zone | 7.08 | 11.89 | 4.81 |
| rdf:type | dbo:City | 7.51 | 10.19 | 2.68 |

Table 3.9: Napoleon sentence weighting schemes

for disambiguation than a relation such as *<dbr:Steffi_Graf dbpprop:husband dbr:Andre_Agassi>*. Therefore, a viable option is the weighting of relations based on their URIs when applying graph connectivity measures on the constructed subgraph. The goal of weighting semantic relations in the DBpedia graph constructed in Section 3.1 is to increase the relevance of especially relevant relations and degrade irrelevant relations, respectively. Thereby, statements can be weighted based on either their predicate or the combination of predicate and object.

The advantage of considering the predicate-object combinations can be illustrated with the example of *dbr:Steffi_Graf*: Here, the statement *rdf:type dbpedia-owl:TennisPlayer* about Steffi Graf is far more informative than the *rdf:type owl:Thing* statement.

Ideally, the importance of each predicate or predicate-object combination for the task of entity linking is learned using a training dataset. This would result in correlation measures in the range of $[-1, 1]$, which indicate for each relation to which extent it is helpful for disambiguation. However, due to the limited scope of the thesis, more generic indicators are applied. Therefore, in the present work the approach for weighting semantic relations by [36] is exploited. In their work, the authors apply measures from the domain of information theory for weighting semantic relations. These measures are based on the notion of Information Content (IC), which is associated with the outcome of a random variable.

The remainder of the section is structured as follows: The next section addresses the Information Content measure. This is followed by a dedicated section for each of the three different weighting schemes, which are all based on the mentioned Information Content measure. Each of those schemes takes a different assumption in terms of how the predicates and objects are interrelated. To illustrate the interplay among the mentioned concepts, a recurrent example is used based on the subgraph from Figure 3.1. This subgraph is derived from the *Napoleon* sentence example used throughout the thesis. The resulting scores for each weighting scheme are displayed in Table 3.9.

| Predicate ($Pred$) | $n_{Pred}$ | $P(\omega_{Pred})$ | $IC(\omega_{Pred})$ |
|---|---|---|---|
| dbo:battle | 83952 | 0.226768% | 6.09 |
| dbo:commander | 36652 | 0.099003% | 6.92 |
| dbo:isPartOfMilitaryConflict | 10587 | 0.028597% | 8.16 |
| dbo:notableCommander | 4867 | 0.013147% | 8.94 |
| dbo:timeZone | 287151 | 0.775642% | 4.86 |
| rdf:type | 2545913 | 6.876925% | 2.68 |

Table 3.10: Napoleon sentence predicate metrics

### 3.4.1 Information Content

The Information Content (IC) is associated with the outcome of a random variable. Underlying this principle is the assumption that the frequency of the relation over the entire dataset is a good proxy for its relevance. [36] define the Information Content measure as follows:

$$IC_{X_{Pred}}(\omega_{Pred}) = -\log(P(\omega_{Pred})), \tag{3.1}$$

where $P(\omega_{Pred})$ is the probability that the random variable $X_{Pred}$ describing the type of edge, i.e. a specific semantic relation, shows the outcome $\omega_{Pred}$.

The weighting schemes of [36] rely on IC measures of *predicate*, *object*, and *predicate-object* combinations. In the following, examples for each of those alternatives are provided. Table 3.10 depicts the IC and probability measures for the predicates within the subgraph from Figure 3.1. In this table, $n_{Pred}$ represents the frequency of the predicate URI within the entire DBpedia graph. Since the DBpedia graph in the present case has a total of $37,021,094$ edges, the probability $P(\omega_{Pred})$ is calculated by dividing $n_{Pred}$ by the total number of edges. For the predicate *dbo:battle*, $P(\omega_{\text{dbo:battle}})$ is calculated as follows:

$$P(\omega_{\text{dbo:battle}}) = \frac{83952}{37021094} \approx 0.00226768. \tag{3.2}$$

Since the Information Content measure is on a logarithmic scale, the vast differences in frequency diminish on the logarithmic scale. For example, the ratio in frequency for *dbo:notableCommander* and *rdf:type* amounts to $1 : 523$, whereas the ratio in Information Content measure is $1 : 3.34$.

Following the Napoleon subgraph example, the same measures for all URIs that appear as triple objects are displayed in Table 3.11. Respectively, Table 3.12 shows the predicate-object combinations for the exemplary case.

### 3.4.2 Joint Information Content

[36] define the Joint Information Content (jointIC) as

$$w_{jointIC}(e) = IC(\omega_{Pred}) + IC(\omega_{Obj}|\omega_{Pred}). \tag{3.3}$$

| Object ($Obj$) | $n_{Obj}$ | $P(\omega_{Obj})$ | $IC(\omega_{Obj})$ |
|---|---|---|---|
| dbo:City | 20229 | 0.054642% | 7.51 |
| dbr:Arthur_Wellesley,_1st_Duke_of_Wellington | 111 | 0.000300% | 12.72 |
| dbr:Battle_of_Waterloo | 50 | 0.000135% | 13.51 |
| dbr:Eastern_Time_Zone | 32812 | 0.088631% | 7.03 |
| dbr:Hundred_Days | 43 | 0.000116% | 13.67 |
| dbr:Napoleon | 159 | 0.000429% | 12.36 |

Table 3.11: Napoleon sentence object metrics

Here, the predicate Information Content measure is accompanied by the conditional Information Content of the object given the predicate. The core idea is to enhance the predicate specifity with the specifity of the object dependent of the occurrence of the predicate. Thus, the measure assumes full dependence between objects and predicates.

### 3.4.3 Combined Information Content

The combined Information Content (combIC) mitigates the dependence assumption of the joint Information Content by computing the weight as the sum of the Information Content of the predicate and the object:

$$w_{combIC}(e) = IC(\omega_{Pred}) + IC(\omega_{Obj}). \tag{3.4}$$

Here, infrequent objects that always occur in combination with an infrequent predicate are not penalized. Overall, this measure assumes the independence of predicates and objects.

### 3.4.4 Information Content and Pointwise Mutual Information

To find an intermediate position in the trade-off between the dependence and the independence assumption from the previous two schemes, [36] exploit the Pointwise Mutual Information (PMI) measure. The PMI between an object and an predicate is defined as follows:

$$PMI(\omega_{Pred}, \omega_{Obj}) = \log \frac{P(\omega_{Pred}, \omega_{Obj})}{P(\omega_{Pred})P(\omega_{Obj})}. \tag{3.5}$$

For the resulting IC and PMI measure (IC+PMI), [36] combine the PMI with the IC of the predicate in the same way as the other two schemes:

$$w_{IC+PMI}(e) = IC(\omega_{Pred}) + PMI(\omega_{Pred}, \omega_{Obj}). \tag{3.6}$$

| Predicate ($Pred$) | Object ($Obj$) | $n_{Pred,Obj}$ | $P(\omega_{Pred}, \omega_{Obj})$ | $IC(\omega_{Obj}\|\omega_{Pred})$ |
|---|---|---|---|---|
| dbo:battle | dbr:Battle_of_Waterloo | 45 | 0.000122% | 7.53 |
| dbo:commander | dbr:Arthur_Wellesley,_1st_Duke_of_Wellington | 39 | 0.000105% | 6.85 |
| dbo:commander | dbr:Napoleon | 85 | 0.000230% | 6.07 |
| dbo:isPartOfMilitaryConflict | dbr:Hundred_Days | 5 | 0.000014% | 7.66 |
| dbo:notableCommander | dbr:Napoleon | 2 | 0.000005% | 7.80 |
| dbo:timeZone | dbr:Eastern_Time_Zone | 31307 | 0.084565% | 2.22 |
| rdf:type | dbo:City | 20229 | 0.054642% | 4.84 |

Table 3.12: Napoleon sentence combined frequencies

## 3.5 Graph Connectivity Measures

Based on the constructed subgraph, which consists of the connections between the candidate entities of the surface forms within the document, graph-based connectivity algorithms can be applied that yield connectivity scores for each candidate entity. Therefore, this section discusses different graph-based algorithms for disambiguation in entity linking.

For the similar task of word-sense disambiguation (WSD), [32] utilize different local and global graph algorithms. Thereby, the authors derive an undirected graph using WordNet as the knowledge base. In the present work, the mentioned local algorithms are adopted for the entity linking domain, with DBpedia as a source of knowledge. Moreover, some adaptions of these algorithms are introduced. Since the edges in the DBpedia graph are directed as per RDF convention, all algorithms are evaluated using both, undirected and directed traversal on the DBpedia graph.

The global algorithms discussed by [32] are omitted, since they are not capable of producing a relevance score for a candidate entity given a surface form. Thus, although they can be used for disambiguation, they are not feasible for generating a ranked list of candidate entities, which is the focus of the present work.

The remainder of the section defines the local connectivity measure problem and discusses different approaches.

### 3.5.1 Local Measures

Local graph connectivity measures determine the relevance of a single vertex $v$ in a graph $G$. Thus, they act as a proxy for the influence of a vertex over a network. All local measures are normalized in the range $[0, 1]$. Thus, the local measure $l$ is formally defined as done by [32]:

$$l : V \rightarrow [0, 1] \tag{3.7}$$

Here, a vertex with a higher value indicates that this vertex is more important compared to a respective vertex with a lower value.

In their work, [32] experiment with five different local measures: Degree, Betweenness, Key-Player-Problem (KPP), PageRank, and Hypertext Induced Topic Selection (HITS). In the following, each of those algorithms is discussed individually. Moreover, two variants of the degree algorithm – global degree and normalized degree – are defined. To illustrate the mentioned algorithms, Table 3.13 displays the measures for the entities of the subgraph from the Napoleon example (cf. Figure 3.1).

Throughout all algorithms descriptions, the notation $G'$ refers to the subgraph $G' \subseteq G$ of the global DBpedia graph $G$. Moreover, $V$ is defined as the set of vertices in the subgraph $G'$; consequently $|V|$ is the number of vertices in $G'$. Respectively, $E$ refers to the set of edges in $G'$.

**Degree** Degree is the most straightforward way to measure the importance of a vertex. [32] define the degree as the number of edges terminating in a given

| Vertex | Degree Centrality $C_D$ | Betweenness Centrality $C_B$ | Key Player Problem $KPP$ | Page Rank $PR$ | HITS $h$ | Normalized Degree $C_{ND}$ | Global Degree $C_{GD}$ |
|---|---|---|---|---|---|---|---|
| Napoleon | .3571 | .0275 | .3929 | .2408 | .4862 | .0246 | .00001654 |
| Napoleon_(Animal_Farm) | 0 | 0 | 0 | 0 | 0 | 0 | .00000090 |
| Napoleon,_Ohio | .1429 | .0027 | .1786 | .0393 | .0425 | .0488 | .00000334 |
| Waterloo,_Ontario | .1429 | .0027 | .1786 | .0393 | .0425 | .0136 | .00001198 |
| Battle_of_Waterloo | .2857 | .0096 | .3571 | .0680 | .4577 | .0588 | .00000554 |
| London_Waterloo_station | 0 | 0 | 0 | 0 | 0 | 0 | .00000236 |
| Wellington | 0 | 0 | 0 | 0 | 0 | 0 | .00009321 |
| Arthur_Wellesley,_1st_Duke_of_Wellington | .2857 | .0137 | .3571 | .2120 | .4072 | .202 | .00001613 |
| Wellington_Rugby_Football_Union | 0 | 0 | 0 | 0 | 0 | 0 | .00000122 |

Table 3.13: Local measures for the vertices from Table 3.7

vertex:

$$deg(v) = |\{\{u, v\} \in E : u \in V\}| \tag{3.8}$$

A disconnected vertex has a degree of zero. Degree centrality is the degree of a vertex normalized by the maximum degree:

$$C_D(v) = \frac{deg(v)}{|V| - 1}, \tag{3.9}$$

where $|V|$ is the number of vertices in the subgraph $G'$. For undirected traversal, the *Napoleon* candidates from the graph in Figure 3.1 have the following degree centrality scores:

- $C_D(Napoleon) = \frac{5}{14} \approx 0.3571$,
- $C_D(Napoleon\_(Animal\_Farm)) = \frac{0}{14} = 0$,
- $C_D(Napoleon, \_Ohio) = \frac{2}{14} = 0.1429$.

**Betweenness** The betweenness of a vertex $v$ is calculated as the fraction of the shortest paths between node pairs that pass through $v$ [14]. Formally, [32] define betweenness as:

$$betweenness(v) = \sum_{s,t \in V : s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}, \tag{3.10}$$

where $\sigma_{st}$ is the number of the shortest paths from $s$ to $t$, and $\sigma_{st}(v)$ the number of the shortest paths from $s$ to $t$ that pass through vertex $v$. Since in the present thesis experiments are conducted using both directed and undirected traversal, depending on the setting the shortest paths are allowed to traverse edges opposite to their direction. [32] normalize the $betweenness(v)$ measure by dividing through the maximum number of node pairs excluding $v$:

$$C_B(v) = \frac{betweenness(v)}{(|V| - 1)(|V| - 2)} \tag{3.11}$$

The hunch behind betweenness is that a node is important if it is contained within many shortest paths between different nodes. Consequently, removing a node with a high betweenness score would increase the distances between many nodes. The betweenness of a disconnected node is zero as no path can pass through it.

In Figure 3.1, the vertex $Napoleon$ is located on the shortest paths between the following vertices:

- $\{Grande\_Armee, Napoleonic\_Wars\}$,
- $\{Grande\_Armee, War\_of\_the\_Sixth\_Coalition\}$,
- $\{Grande\_Armee, Hundred\_Days\}$,

- $\{Hundred\_Days, Napoleonic\_Wars\}$,
- $\{Hundred\_Days, War\_of\_the\_Sixth\_Coalition\}$,
- $\{Napoleonic\_Wars, War\_of\_the\_Sixth\_Coalition\}$,
- $\{Battle\_of\_Waterloo, Napoleonic\_Wars\}$,
- $\{Battle\_of\_Waterloo, War\_of\_the\_Sixth\_Coalition\}$.

Except the first two, all other vertex pairs possess two shortest paths, the second reaching either through $Battle\_of\_Waterloo$ or $Arthur\_Wellesley, \_1st\_Duke\_of\_Wellington$. Thus, the betweenness for the resource $Napoleon$ is calculated as follows:

$$betweenness(Napoleon) = 2 * \frac{1}{1} + 6 * \frac{1}{2} = 5. \tag{3.12}$$

For $Napoleon\_(Animal\_Farm)$, the betweenness score is zero, as the vertex is unconnected. $Napoleon\_(Ohio)$ is contained in one of the two shortest paths between $dbo{:}City$ and $Eastern\_Time\_Zone$; thus, $betweenness(Napoleon\_(Ohio)) = \frac{1}{2}$. The resulting betweenness centrality values for the *Napoleon* candidates are:

- $C_D(Napoleon) = \frac{5}{14*13} \approx 0.0275$,
- $C_D(Napoleon\_(Animal\_Farm)) = \frac{0}{14*13} = 0$,
- $C_D(Napoleon, \_Ohio) = \frac{0.5}{14*13} = 0.0027$.

**Key Player Problem (KPP)** The KPP considers a vertex important if it is relatively close to all other vertices [4]:

$$KPP(v) = \frac{\sum_{u \in V: u \neq v} \frac{1}{d(u,v)}}{|V| - 1}. \tag{3.13}$$

where $d(u, v)$ is the length of the shortest path between $u$ and $v$, iff such a path exists, and zero otherwise. For non-existing paths, [32] assign a distance score of $\frac{1}{|V|}$. In the present work, the zero modification is used, since unconnected vertices are not linked per default. The numerator of the KPP measure is the sum of the inverse shortest distances between $v$ and all other nodes. According to [32], KPP is similar to the better known closeness centrality measure proposed by [35], which is defined as the reciprocal of the total shortest distance from a given node to all other nodes. However, the authors consider only KPP since it outperforms closeness centrality in their experiments.

Following the *Napoleon* example in Figure 3.1, $KPP(Napoleon)$ is calculated as follows:

$$KPP(Napoleon) = \frac{5 * \frac{1}{1} + 1 * \frac{1}{2} + 8 * \frac{1}{14}}{14} = \frac{\frac{85}{14}}{14} = \frac{85}{196} \approx 0.4337. \tag{3.14}$$

Respectively, the score for all unconnected vertices equals zero, as illustrated with the example of $Napoleon\_(Animal\_Farm)$:

$$KPP(Napoleon\_(Animal\_Farm)) = \frac{14 * 0}{14} = 0. \tag{3.15}$$

**PageRank** PageRank was presented by Brin and Page in 1998 [5]. Originally it was targeted at hyperlinks on the Web for a search ranking algorithm. PageRank determines the relevance of a node $v$ using an iterative recursive approach. [5] define the PageRank algorithm as follows:

$$PR(v) = (1 - \alpha) + \alpha \sum_{\{u,v\} \in E} \frac{PR(u)}{outdegree(u)}. \tag{3.16}$$

Here, $\alpha$ is a damping factor, which [5] commonly set to $0.85$. Thus, if a node is disconnected, its PageRank value equals $1 - \alpha$. As in KPP, in the present work a zero score is used for disconnected vertices in the subgraph, to prevent them from being annotated.

In the present work, the damping factor $\alpha = 0.85$ and number of iterations $N = 52$ is used, as recommended by [5].

**Hypertext Induced Topic Selection (HITS)** HITS, like PageRank, is an iterative algorithm based on the linkage of the documents on the web. It has been proposed by [19] in 1999 and determines two values for each node $v$: the authority $a(v)$ and the hub value $h(v)$. Authority and hub values are defined in terms of one another in a mutual recursion:

$$h(v) = \sum_{u:(v,u) \in E} a(u); \quad a(v) = \sum_{u:(u,v) \in E} h(u). \tag{3.17}$$

An authority value is computed as the sum of the scaled hub values that point to that page. A hub value is the sum of the scaled authority values of the pages it points to. The hub and authority values for all vertices are commonly initialized to one: $\forall v \in V : h(v) = 1, a(v) = 1$. An entire iteration consists of multiple phases: After updating the authority values the hub values are computed. Next, the normalization takes place by dividing each hub score by the square root of the sum of the squares of all hub scores, and dividing each authority score by square root of the sum of the squares of all authority scores. The normalization is necessary so that the values eventually converge instead of further diverging.

For the present work, the number of iterations is set to 20, which is proposed in [19]. Moreover, the authority value $a(v)$ is used as the connectivity measure.

**Global Degree** As another baseline measure, an adaption of the degree algorithm by [32] is used. Hereby, instead of the subgraph $G'$, the global graph $G$ is used:

$$C_{GD}(v) = \frac{deg(v)}{|V_G| - 1}, \tag{3.18}$$

where $|V_G|$ is the number of vertices in the entire DBpedia graph. This measure is similar to the Popularity feature from Section 2.5.1. However, instead of using the Wikipedia article links, it is based on the infobox, category, and ontology relations.

Following the exemplary sentence, the global degree for $Napoleon$ is calculated as follows: Using undirected traversal, the *Napoleon* candidates from the graph in Figure 3.1 have the following global degree scores:

- $C_{GD}(Napoleon) = \frac{203}{12273625-1} \approx 0.00001654$,
- $C_{GD}(Napoleon\_(Animal\_Farm)) = \frac{11}{12273625-1} \approx 0.00000090$,
- $C_D(Napoleon, \_Ohio) = \frac{41}{12273625-1} \approx 0.00000334$.

**Normalized Degree** First experiments revealed that all graph-based measures are biased towards popular entities. Therefore, another adaption of the degree algorithm is proposed by normalizing based on the entity popularity estimated as global degree:

$$C_{ND}(v) = \frac{deg_{G'}(v)}{deg_G(v)}, \tag{3.19}$$

where $deg_{G'}(v)$ is the degree of $v$ in the subgraph $G'$, and $deg_G(v)$ is the degree of $v$ in the global graph $G$.

Following the *Napoleon* example in Figure 3.1, the normalized degree is calculated as follows:

- $C_{GD}(Napoleon) = \frac{5}{203} \approx 0.0246$,
- $C_{GD}(Napoleon\_(Animal\_Farm)) = \frac{0}{11} = 0$,
- $C_D(Napoleon, \_Ohio) = \frac{2}{41} \approx 0.0488$.

## 3.6 Federated Entity Linking

All state of the art systems evaluated in Section 2.5.2 contain multiple features for the disambiguation task. This is due to the fact that each feature has its advantages and drawbacks. For example, the graph-based approach to disambiguation works best when all surface forms within the document fit into the same narrow domain, as it is the case in the exemplary *Napoleon* sentence used throughout the thesis.

After constructing a DBpedia graph and evaluating suitable algorithms for disambiguation, it is possible to combine this graph-based approach with other features. In the present work, the DBpedia Graph Linker is combined with DBpedia

Spotlight and its contained features. Thereby, multiple combination techniques are possible, depending on the research question derived from the point of view:

- *Graph-based viewpoint*: How can the Graph Linker benefit from incorporating statistical disambiguation features?
- *Neutral viewpoint (system combination)*: How can the overall accuracy be improved by combining both disambiguation systems using a black-box approach?
- *DBpedia Spotlight viewpoint (feature combination)*: How can the multi-feature statistical system Spotlight benefit from incorporating the graph-based feature?

The remainder of the section discusses possible answers to each of the mentioned questions individually.

### 3.6.1 Enhanced Graph-based Disambiguation

For enhancing graph-based disambiguation with external features, these features need to be incorporated in a way that mitigates certain drawbacks of graph-based disambiguation.

In the present work, experiments are conducted that deal with a major flaw in graph-based disambiguation: the problem of *unrelated surface forms*. This drawback stems from the fact that in graph-based disambiguation, the candidate entities from unrelated surface forms, which do not fit into the overall domain of a document, are often unconnected in the resulting subgraph. When all candidate entities are singletons in the subgraph, a proper disambiguation is not possible. Therefore, a fallback strategy can be exploited, which uses the scores of external features in such cases. As discussed in Section 2.5.1, in the case of entity linking, the prior probability forms a strong baseline. Thus, in the present work experiments using a prior probability fallback strategy are conducted. This strategy takes action when all candidates entities of a surface form are unconnected in the subgraph, and functions by setting the similarity score for each entity of the surface form equal to its prior probability.

### 3.6.2 System Combination

For combining the DBpedia Spotlight and the DBpedia Graph linker on a system level, both ranked lists of candidate entities for each surface form, along with their respective scores, need to be merged. Hereby, DBpedia Spotlight is treated as a black-box system which produces a single similarity score for each entity.

In the present work, the most straightforward combination technique, simple linear regression, is applied:

$$P(e) = \alpha + \beta_s P_s(e) + \beta_g P_g(e), \tag{3.20}$$

where $\beta_s \in [0, 1]$ is the weight for the DBpedia Spotlight system, $\beta_g \in [0, 1]$ is the weight for the DBpedia Graph Linker, $P_s(e)$ is the function that assigns a DBpedia Spotlight score for the entity $e$, and $P_g(e)$ assigns a graph-based score. The values for $\alpha$ and $\beta_i$ are learned using a training dataset.

There are additional combination techniques such as switching, where one system is chosen based on some fix criteria. Exemplary criteria might be textual criteria such as the text length, or the scores produced by each system. However, the evaluation of these techniques is omitted due to the limited scope of the thesis.

In the present case, there are two main issues when combining the scores from these two systems, the differences in *candidate selection* and the *scaling of the scores*. These issues are discussed in the remainder of the section.

### Candidate Selection Differences

Both, DBpedia Spotlight and the Graph Linker, generate their own set of candidate entities for a surface form. Hereby, DBpedia Spotlight prunes the candidate set using a top-$k$ approach based on prior probability, with $k = 20$ in release 0.6 and $k = 10$ in release 0.7. Due to the explosive graph traversal, the DBpedia Graph Linker applies a smaller $k$, while using the same pruning strategy. Thus, the candidate set of the Graph Linker is always a subset of the candidate set of the Spotlight Linker. This unequal candidate generation strategy results in entities where only statistical similarity scores exist. There are multiple strategies on how to deal with this inequality:

1. Reduce the candidates to the set union of both sets. Effectively, this prunes the candidate set of Spotlight further than the system itself intends. Assuming that the value for $k$ that Spotlight uses maximizes its accuracy, this option has the drawback of decreasing the accuracy of the Spotlight system.

2. Use a heuristic that assumes a graph-based score $P_g(e)$ of zero for candidates without a graph-based score, and then apply the weighted linear combination. This option has the drawback that it degrades the statistical scores for entities without a graph-based score, since with $P_g(e) = 0$, the weighted linear combination degrades the statistical score $P_s(e)$ by multiplying it with its coefficient $\beta_s$. This makes it highly unlikely that a candidate entity with a graph score of zero is linked. Thus, this option is similar to the previously discussed candidate set union strategy.

3. Omit the weighted combination and simply use the statistical score for candidates without a graph-based score. This is similar to a switching approach, where either the combination of both systems or merely the statistical system is considered.

DBpedia Spotlight has a higher overall accuracy than the Graph Linker. This fact, combined with the circumstance that the first two options modify the statistical approach by effectively degrading the scores of some candidates, leads to the assump-

tion that the last option works best. However, the second option is also evaluated in the present work.

### Score Distribution Differences

DBpedia Spotlight uses a generative probabilistic model, where effectively, the score of an entity is calculated as the joint probability of all feature probabilities:

$$P_{joint}(e) = P(e)P(s|e)P(c|e) \tag{3.21}$$

To generate a normalized disambiguation score in the range $[0, 1]$, the combined scores are divided by the sum of the scores of all candidate entities for the given surface form:

$$P_{norm}(e) = \frac{P_{joint}(e)}{\sum_{i=1}^{|C|} P_{joint}(e_i)} \tag{3.22}$$

where $C = \{e_1, e_2, \ldots, e_n\}$ is the set of candidate entities for a surface form. The linearly distributed scores of the Graph Linker are also normalized by dividing through the sum of scores of all candidate entities for a surface form:

$$P_{norm}(g|e) = \frac{P(g|e)}{\sum_{i=1}^{|C|} P(g|e_i)}. \tag{3.23}$$

In the probabilistic model, the joint probability distribution is similar to a sigmoid function, which means that scores at the border of the range $[0, 1]$ occur more frequently. This differs from the linear model, where scores are equally distributed. There are different possibilities to tackle the mentioned problem:

1. Proceed without any modification, and simply apply the weighted linear combination. The drawback of this approach is that the linear regression assumes a linear distribution, and thus, learning the predictor function does not yield optimal results.

2. The DBpedia Graph Linker adopts the scaling of Spotlight by normalizing the scores using an appropriate sigmoid function. This means that a log-linear model needs to be applied, since linear regression is not feasible in this case. In terms of mathematical correctness, this is the better option. However, finding an appropriate sigmoid function is challenging.

Due to the limited scope of the thesis, the first option is used. The assumption underlying this decision is that the effect of the mathematical incorrectness in combining the linear and the probabilistic model is nearly negligible. This assumption is based on the fact that the combination favors the statistical approach, since the sigmoid distribution of the Spotlight scores pushes the scores of the higher ranked entities closer to one. As in the candidate selection issue, due to the higher overall accuracy of the Spotlight linker, favoring the statistical approach is acceptable. However, a mathematical correct solution would likely yield better results.

### 3.6.3 Feature Combination

From the view point of DBpedia Spotlight, the question is how the overall system can be improved by incorporating the graph-based feature. For the present thesis, multiple linear regression is used as an approach to model the relationship between the set of features:

$$P(e) = \beta_e P(e) + \beta_{s|e} P(s|e) + \beta_{c|e} P(c|e) + \beta_{g|e} P(g|e) + \varepsilon, \tag{3.24}$$

where the $\beta_i$ and $\varepsilon$ are learned based on a training dataset.

The issue discussed in Section 3.6.2 – candidate selection differences – is also relevant when combining all features. For the feature combination task, the difference is eliminated by using the same $k$ for both linkers, with $k = 10$. The implication of the resulting slight decrease in accuracy for the Graph Linker is not as severe as in the previous section, since four scores instead of two are combined.

The main challenge in combining the features of DBpedia Spotlight and the Graph Linker are the *log-scaled feature scores* of Spotlight. This issue is discussed in the remainder of the chapter.

#### Log-scaled Feature Scores

In the generative probabilistic model of DBpedia Spotlight, the probabilities, such as the probability of a token in the context of an entity, are negligibly small [11]. Typical values for the resulting context feature probability are e.g. $P(c|e) \approx 10^{-150}$, which does not fit into any regular 64-bit data type. To avoid underflows, Spotlight internally represents the probabilities of all features in logarithmic space (e.g., $P(c|e) \approx 10^{-150} \Leftrightarrow \log P(c|e) \approx -345$). Exploiting the properties of the logarithm function, the log-scaled feature probabilities are then combined by summing them up in an unweighted combination:

$$\begin{aligned} \log P_{joint}(e) &= \log(P(e)P(s|e)P(c|e)) \\ &= \log P(e) + \log P(s|e) + \log P(c|e). \end{aligned}$$

Table 3.14 shows the log-scaled probabilities for the four best ranked candidate entities in the Napoleon example used throughout the thesis. Here, due to the fact that the sentence has only seven words, the context probabilities are very large compared to those in a large document. Nevertheless, the example illustrates the scale of the probabilities, especially regarding the context feature.

To generate a final disambiguation score in the range $[0, 1]$, the log-scaled combined scores are normalized using a softmax function:

$$P_{norm}(e) = \frac{e^{\log P_{joint}(e)}}{\sum_{i=1}^{|C|} e^{\log P_{joint}(e_i)}} = \frac{P_{joint}(e)}{\sum_{i=1}^{|C|} P_{joint}(e_i)}. \tag{3.25}$$

Applying the softmax function transforms the log-scaled scores back to the probabilistic scaling, and at the same time normalization is achieved by dividing through the sum of scores of all candidate entities for a surface form.

| Surface Form | Candidate Entity | $P_{joint}(e)$ | $P(s\|e)$ | $P(c\|e)$ | $P(e)$ |
|---|---|---|---|---|---|
| Napoleon | Napoleon | -74.496 | -0.037 | -65.531 | -8.927 |
| | Napoleonic_Wars | -82.653 | -6.563 | -66.159 | -9.932 |
| | Napoleon_(Animal_Farm) | -84.563 | -4.662 | -65.940 | -13.962 |
| | Napoleon,_Ohio | -85.904 | -5.256 | -66.504 | -14.144 |
| Waterloo | Battle_of_Waterloo | -75.601 | -1.890 | -62.676 | -11.034 |
| | Waterloo,_Ontario | -78.616 | -1.352 | -65.959 | -11.305 |
| | Waterloo,_Belgium | -80.076 | -2.718 | -64.542 | -12.816 |
| | London_Waterloo_station | -80.525 | -2.479 | -66.596 | -11.451 |
| Wellington | Wellington | -76.246 | -0.429 | -66.363 | -9.455 |
| | Arthur_Wellesley,_1st_Duke_of_Well-ington | -77.997 | -3.054 | -64.224 | -10.720 |
| | Wellington_rugby_league-_team | -80.643 | -3.549 | -64.575 | -12.519 |
| | Wellington_Rugby_Football_Union | -81.728 | -3.129 | -66.544 | -12.055 |

Table 3.14: Spotlight top-4 log-scaled feature probabilities

The core problem is, that the log-scaled feature scores are not applicable for the multiple linear regression method used in this work. Therefore, before applying the linear combination, the features need to be transformed into a linear scale. For $P(s|e)$ and $P(e)$, the scores can be transformed by reverting the logarithm scales using the exponential function:

$$P(s|e) = e^{\log P(s|e)},\ P(e) = e^{\log P(e)}. \tag{3.26}$$

For $P(c|e)$, this reversion is not possible due to the mentioned underflow issue. As a workaround, the normalized context probability $P_{norm}(c|e)$ is used. This normalized probability is calculated in the same manner as the normalized joint probability:

$$P_{norm}(c|e) = \frac{e^{\log P(c|e)}}{\sum_{i=1}^{|C|} e^{\log P(c|e_i)}} = \frac{P(c|e)}{\sum_{i=1}^{|C|} P(c|e_i)}. \tag{3.27}$$

Obviously, this workaround is mathematically incorrect. However, due to the limited scope of the thesis, learning a log-linear model was not possible.

The final adapted weighted linear combination formula looks as follows:

$$P(e) = \beta_e P(e) + \beta_{s|e} P(s|e) + \beta_{c|e} P_{norm}(c|e) + \beta_{g|e} P(g|e) + \varepsilon. \tag{3.28}$$

# Chapter 4

# Evaluation

The purpose of the evaluation is twofold, with an internal and an external perspective. As outlined in Chapter 3, throughout each step in the setup process of the DBpedia Graph Linker, various design decisions and parameters occurred. To decide among the different options for each decision, each of the options needs to be evaluated based on actual data. Therefore, from an internal perspective, an evaluation is necessary to fine-tune each parameter within the system. From an external perspective, the evaluation is used to compare the DBpedia Graph Linker with other entity linking systems. This comparison enables deductions regarding the domains and problem scenarios in which the linkers excel, and respectively, where the shortcomings of each linker are.

Throughout evaluation, the framework for benchmarking entity-annotation systems (BAT framework) is used, which has been developed by Cornolti et al. [9]. This freely accessible framework[1] contains different gold standard datasets from domains such as news, tweets or web pages, and enables the evaluation of entity linkers that link against Wikipedia pages, which makes it suitable for the present scenario. The framework uses the standard metrics for entity linking, precision and recall, which are used to compare the performance of the linkers.

The remainder of the chapter is structured as follows: Section 4.1 discusses the different metrics which can be used to evaluate entity linkers, along with a conclusion of which metrics fit best to the present scenario. This is followed by an analysis of the datasets that the BAT framework includes (Section 4.2). Next, the baseline linker $P(s|e)$ is evaluated (Section 4.3), to enable the comparison with the evaluated DBpedia Graph Linker in Section 4.4. Hereby, apart from an evaluation of the linker's accuracy, an error analysis is conducted to investigate the shortcomings of the graph-based approach. In the same manner, the DBpedia Spotlight linker is evaluated in the subsequent Section (4.5). Next, the different approaches for the federated linker, which combines DBpedia Graph and the DBpedia Spotlight, are evaluated (Section 4.6). Finally, the core findings of the evaluation are summarized in Section 4.7.

[1] https://github.com/marcocor/bat-framework/

## 4.1 Metrics

For evaluating entity linkers to Wikipedia, various metrics can be applied. Throughout evaluation, the standard evaluation metrics from the domain of information retrieval, *precision*, *recall*, and *F-measure* are measured [22]. These metrics are based on classifying each annotation into the matrix consisting of true positives ($tp$), false positives ($fp$), true negatives ($tn$), and false negatives ($fn$). Let $g$ be the correct annotations of a text input $t$ and $s$ the set of annotations produced by a linker. Let $M$ be the binary relation which specifies the notion of a "correct match" between two annotations. Then, the measures can be defined as done by [9]:

$$\begin{aligned} tp(s,g,M) &= \{x \in s |\ \exists x' \in g : M(x',x)\} \\ fp(s,g,M) &= \{x \in s |\ \nexists x' \in g : M(x',x)\} \\ tn(s,g,M) &= \{x \notin s |\ \nexists x' \in g : M(x',x)\} \\ fn(s,g,M) &= \{x \in s |\ \nexists x' \in s : M(x',x)\} \end{aligned} \tag{4.1}$$

As the definitions emphasize, all measures depend on the matching relation $M$, which defines a correct match between an annotation from the linker and the gold standard annotation.

For the set union of the gold standard and linker annotations of all documents, the precision ($P$), recall ($R$), and $F_1$-measure ($F_1$) are defined as follows:

$$\begin{aligned} P(s,g,M) &= \frac{|tp(s,g,M)|}{|tp(s,g,M)| + |fp(s,g,M)|} \\ R(s,g,M) &= \frac{|tp(s,g,M)|}{|tp(s,g,M)| + |fn(s,g,M)|} \\ F_1(s,g,M) &= \frac{2P(s,g,M)R(s,g,M)}{P(s,g,M) + R(s,g,M)} \end{aligned} \tag{4.2}$$

Depending on the entity-annotation problem, different matching relations $M$ can be applied. The matching relations involve two possible dimensions: the "semantic" one of the entities, and the "syntactic" one of the surface forms [9]. Regarding annotations, [9] distinguish between three different annotation problems: Annotate to Wikipedia (A2W), Disambiguate to Wikipedia (D2W), and Scored-annotate to Wikipedia (Sa2W). In the following, each problem is described, including the matching relation $M$ that suits the respective problem.

**Disambiguate to Wikipedia (D2W) Problem** In the Disambiguate to Wikipedia problem, the entity linker receives a text and a set of surface forms as input. The task is to find the set of annotations, where each annotation assigns an entity to a surface form. Here, the key aspect is the definition of an entity match. This is due to the fact that the gold standard annotations often contain redirect pages in Wikipedia. For example, a gold standard annotation might link the redirect page *dbr:Michael_Jeffrey_Jordan*, whereas the linker links

to *dbr:Michael_Jordan*. To overcome this shortcoming, [9] propose a *dereference function* $d(e)$ that maps each redirect to its de-referenced concept. Based on this function, the *strong annotation match* relation $M_a$ between two annotations $a_1$ and $a_2$ exists if the following is true: their surface forms are equal and their entities $e_1$ and $e_2$ share the same de-referenced concept ($d(e_1) = d(e_2)$).

**Annotate to Wikipedia (A2W) Problem** Unlike D2W, the Annotate to Wikipedia problem requires the linker to discover the surface forms in the document. Besides the semantic aspect discussed in the D2W section, the A2W problem includes a syntactic dimension that is concerned with the textual overlap of surface forms. For example, the gold standard might discover the surface form *President Barack Obama* and link it to *Barack Obama*, whereas the entity linker only considers *Obama* as the surface form. Here, a certain fuzziness is needed to deal with the mentioned problem. [9] address this issue by introducing a *weak annotation match* measure that considers two annotations equal if they textually overlap in the input text. This measure also includes the de-referencing function from the previous section.

**Scored-annotate to Wikipedia (Sa2W) Problem** The scored annotations to Wikipedia problem asks the linker to produce a likelihood score for each annotation. In fact, the majority of the state of the art systems produce such scores. Since there are no existing datasets that include such likelihood measures in their gold standard annotations, there is no way to directly evaluate the produced likelihood values. Therefore, in order to evaluate Sa2W systems, a problem reduction approach to the A2W problem is necessary. To this end, [9] propose the calculation of an optimal threshold $t \in [0, 1]$, that is computed based on the entity linkers output regarding an entire gold standard dataset. The reduction takes place by disregarding all annotations with a likelihood score below $t$.

Since the BAT framework is targeted at the entire entity linking task, in the present work DBpedia Spotlight is used for spotting surface forms in the text and generating candidate entities.

Initial experiments revealed that with the Sa2W problem, it is difficult to assess the disambiguation performance. This is due to the confusion whether the majority of errors are due to spotting or disambiguation errors. Since the thesis is focussed on the disambiguation phase of the entity linking process, the D2W problem is more suited. However, in its initial version, the BAT framework did not support the D2W problem using DBpedia Spotlight as linker. Therefore, as part of the present thesis, the BAT framework was extended to support native D2W using DBpedia Spotlight[2].

[2]https://github.com/marcocor/bat-framework/commit/ac093efc522282e3b0973a81b567a41dc6942e74

## 4.2 Datasets

For evaluation, the datasets supported by the BAT framework are used. Overall there are five different datasets: IITB, MSNBC, AQUAINT, AIDA/CO-NLL and Meij Twitter. These datasets can be classified according to their gold standard types. Hereby, the present thesis follows the terminology by [9]:

- *Annotate to Wikipedia (A2W)*: The dataset gold standard identifies the relevant surface forms in the input text and assigns to each of them the respective entities. Thus, it contains the position and length of the surface forms in the document.
- *Concepts to Wikipedia (C2W)*: The dataset gold standard consists of a set of relevant entities that are mentioned within the document. However, there is no information regarding the surface forms matching the entities within the document.

The datasets differ a lot in terms of the types of surface forms they annotate. Therefore, in the following each dataset is described, along with its annotation characteristics. Supporting the analysis of the datasets, Table 4.1 contains basis statistics about the datasets. References to those statistics are provided within the individual dataset descriptions.

- *IITB*: The IITB dataset contains manually annotated texts drawn from popular Web pages about sport, entertainment, science and technology, and health [20]. As shown in Table 4.1, the IITB dataset has the highest annotation frequency and the largest set of different topics. Overall, it is the most detailed dataset since almost all surface forms, including those whose related concepts are not highly relevant, are annotated. Thus, it is also the most challenging dataset and not suited for linkers that only annotate named entities.
- *MSNBC*: The MSNBC dataset, which was introduced in [10], contains newswire text from the MSNBC news network. With only 279 distinct topics, the MSNBC dataset has the fewest distinct topics and also the lowest annotation frequency. This is due to the fact that only the most important entities and their referring mentions are annotated.
- *AQUAINT*: The AQUAINT dataset consists of English newswire texts from the original AQUAINT corpus [29]. Like MSBNC, the AQUAINT dataset has a very low annotation frequency. Moreover, there are few distinct topics, since only important topics are annotated. A crucial characteristic of the dataset is the fact that only the first surface form of each entity is annotated. This is supposed to reflect the linking style in Wikipedia. Therefore, it is not suited for evaluating the surface form spotting of most annotators.
- *AIDA/CO-NLL*: The AIDA/CO-NLL dataset, introduced in [17], builds on the CoNLL 2003 entity-recognition task [40]. The documents are taken

| Dataset | IITB | MSNBC | AQUAINT | AIDA/CO-NLL | Meij |
|---|---|---|---|---|---|
| Dataset Domain | Web Pages | News | News | News | Tweets |
| Gold Standard | A2W | A2W | A2W | A2W | C2W |
| Total Annotations | 11249 | 658 | 727 | 4485 | 812 |
| Total Documents | 103 | 20 | 50 | 231 | 502 |
| Average Annotations per Document | 109.21 | 32.9 | 14.54 | 19.42 | 1.62 |
| Average Document Length | 3879.369 | 3316.1 | 1415.98 | 1039.94 | 80.19 |
| Annotation Frequency (Average Annotations per Character) | *0.0282* | 0.0099 | 0.0103 | 0.0187 | 0.02017 |
| Longest Document | 9191 | 5253 | 1563 | 3992 | 143 |
| Distinct Topics | *3749* | 279 | 572 | 1537 | 566 |
| Total Documents without Annotations | 0 | 0 | 0 | 1 | 126 |

Table 4.1: Evaluation Dataset Statistics

from the Reuters Corpus V1, and thus, belong to the news domain. The dataset annotates a large set of surface forms referring to named entities, which is acknowledged by the high number of distinct topics and annotation frequency. However, common names such as *Wednesday* are not annotated. Since entities are annotated at each occurrence of a surface forms, it is suited for evaluating all steps within the entity linking process, including spotting. The entire dataset, which consists of 1393 documents, is divided into three chunks: *Training*, *Test A* and *Test B*. As done by [17], in the present work the training chunk, which contains 946 documents, is used for all training tasks. Following Cornolti et al.'s approach in [9], the *Test B* part is used for the experiments in the present work. This part consists of 231 documents. For brevity reasons, in the remainder of the chapter, AIDA/CO-NLL is listed as evaluation dataset, even though this actually refers to the *Test B* chunk of the dataset, and not the entire dataset.

- *Meij Twitter*: The Meij Twitter dataset consists of tweets annotated with a set of occurring entities [26]. Since tweets are very short, they contain about one to two annotations per tweet in average. Moreover, there are many documents without a single annotation (25%).

  The corpus is a C2W dataset, since it only contains a set of concepts for each tweet, without the respective positions in the text. For the Sa2W metric, the BAT framework applies a problem reduction approach where the linker's annotation are integrated into a set union of annotated concepts. This set is then compared with the gold standard set. However, the missing positions as text offset make an evaluation with the D2W metric impossible.

In the present thesis, the decision was to use the AIDA/CO-NLL dataset throughout evaluation, since it has none of the drawbacks mentioned in the following for the other datasets. The Meij datasets is disregarded since it does not allow D2W evaluation due to the missing positions of the surface forms in the microposts. As for the AQUAINT and MSNBC dataset, the total number of documents is very low, which makes splitting the documents into a training and testing set impractical. Moreover, the corpora are considered not representative for natural language, since the overall distinct annotated topics are very low. In the present situation, the drawback of the IITB dataset is that its annotations are not restricted to named entities, but also to rather irrelevant concepts. Other considered systems, such as AGDISTIS, only annotate named entities, therefore the IITB dataset cannot be used to compare the accuracy of those linkers with the linker developed in the thesis.

## 4.3 Baseline Linkers

In the present work, the prior probability feature $P(s|e)$ and the graph-based AGDISTIS linker are evaluated as baseline linkers.

| Dataset | Algorithm | Best Score Threshold | Precision | Recall | $F_1$ |
|---|---|---|---|---|---|
| AIDA/CO-NLL | $P(s\|e)$ | 0.398 | 0.621 | 0.508 | 0.559 |
| IITB | $P(s\|e)$ | 0.367 | 0.727 | 0.243 | 0.365 |
| MSNBC | $P(s\|e)$ | 0.75 | 0.573 | 0.428 | 0.49 |
| AQUAINT | $P(s\|e)$ | 0.828 | 0.475 | 0.424 | 0.448 |
| Meij | $P(s\|e)$ | 0.281 | 0.671 | 0.289 | 0.404 |

Table 4.2: Baseline $P(s|e)$ Sa2W Accuracy

| Dataset | Algorithm | Precision | Recall | $F_1$ |
|---|---|---|---|---|
| AIDA/CO-NLL | $P(s\|e)$ | 0.715 | 0.687 | 0.701 |
| IITB | $P(s\|e)$ | 0.739 | 0.726 | 0.732 |
| MSNBC | $P(s\|e)$ | 0.792 | 0.737 | 0.764 |
| AQUAINT | $P(s\|e)$ | 0.876 | 0.836 | 0.856 |

Table 4.3: Baseline $P(s|e)$ D2W Accuracy

Throughout literature, the prior probability feature $P(s|e)$ is commonly used as a baseline for evaluating the disambiguation task in entity linking (cf. [31, 34]). This is due to the fact that this single feature is a very reliable indicator of the correct disambiguation [34]. Using DBpedia Spotlight for spotting and candidate selection, Table 4.2 displays the baseline performance of the $P(s|e)$ feature in the Sa2W task. The baseline performance of the $P(s|e)$ feature in the D2W task is shown in Table 4.3. Hereby, DBpedia Spotlight is used for candidate selection. The Sa2W results are included to allow the comparison with the evaluation in [9]. The large differences in accuracy between Sa2W and D2W are due to spotting errors. This is especially relevant in the case of the AQUAINT dataset, where only the first mention of equal surface forms are annotated. Since the spotting in DBpedia Spotlight does not support this configuration, the Sa2W performance values for this corpus are negligible.

For comparing the DBpedia Graph Linker with the other publicly available DBpedia Graph Linker, AGDISTIS, the accuracy of this linker on the AIDA/CO-NLL dataset is required. However, in their publication, the authors evaluate AGDISTIS only against their customly created datasets (cf. [41]). Therefore, a plugin which adds support for the AGDISTIS annotator in the BAT-framework was developed as part of the present thesis[3].

The evaluation results for the AGDISTIS linker are shown in Table 4.4. Hereby, the settings recommended by its authors in [41] were applied. As the results show, the AGDISTIS linker performs worse than the prior probability linker $P(s|e)$ on all evaluated datasets. The low performance on the IITB dataset is due to the fact that AGDISTIS only annotates named entities.

[3]https://github.com/marcocor/bat-framework/commit/f1bf74b97132bd168207a3e677fd4d0f98d2107b

| Dataset | Precision | Recall | $F_1$ |
|---|---|---|---|
| AIDA/CO-NLL | 0.642 | 0.556 | 0.596 |
| IITB | 0.646 | 0.204 | 0.31 |
| MSNBC | 0.796 | 0.729 | 0.761 |
| AQUAINT | 0.777 | 0.422 | 0.547 |

Table 4.4: AGDISTIS D2W Accuracy

Overall, the D2W evaluation shows that the baseline feature achieves a high accuracy in the range of $[70.1\%, 85.6\%]$. Moreover, the accuracy of each dataset indicates how challenging each dataset is for the task of disambiguation, with AQUAINT being the easiest dataset and AIDA/CO-NLL the most challenging.

## 4.4 DBpedia Graph Linker

The evaluation of the DBpedia Graph Linker is conducted along two perspectives: a qualitative error analysis and a quantitative accuracy analysis. From a qualitative perspective, an error analysis is performed using a random sample of documents drawn from gold standard datasets. The quantitative evaluation is concerned with measuring the overall accuracy of the linker in terms of precision, recall, and $F_1$-measure. In the following, both evaluation perspectives are discussed in dedicated sections.

### 4.4.1 Error Analysis

After the initial version of the DBpedia Graph Linker was established, an error analysis was conducted to investigate the benefits and limitations of the linker. Thereby, the Graph linker annotation errors were analyzed to find common errors categories. The goal was to outline domains and text characteristics where the graph-based approach excels and where it does not.

To this end, the subgraphs, created as part of the graph-based disambiguation process, were analyzed for two random documents of the datasets AIDA/CO-NLL, IITB, MSNBC and AQUAINT. Hereby, a directed graph traversal with a maximum path distance of 3 was applied. Moreover, the target DBpedia graph contained the filtered ontology and category information. The resulting subgraphs possess more than 250 vertices and 500 edges on average.

The analysis resulted in different error categories, that are further discussed in the following sections.

#### Popular Entity Bias

The investigation revealed that in general, graph-based entity linkers are biased towards more popular entities. Popular entities (e.g. countries such as *Germany*) are strongly interconnected with many other entities, therefore the resulting subgraph

contains a lot more paths that start or end in more popular entities compared to less popular entities. The following text excerpt about Berlin serves as an example:

> "After World War II, the city became divided into *East Berlin*, [...], and West Berlin, [...]."

For the surface form *East Berlin*, the set of candidate entities includes *dbr:Berlin* and *dbr:East_Berlin*. Figure 4.1 displays an excerpt of the subgraph resulting from this document. The graph shows that, due to the higher popularity, *dbr:Berlin* is far more interconnected. Therefore, all graph connectivity measures annotate the surface form *East Berlin* with the entity *dbr:Berlin*. This bias could be alleviated by normalizing a local graph connectivity measure by the entity's overall degree; however, the experiments in later sections show that this drastically reduces accuracy, since popularity is also an indicator of how often the entity is annotated.

#### Unrelated Domain Surface Forms

Overall, the graph-based linker performs well if all surface forms within the entire text belong to a single domain. In the case of unrelated surface forms, where the surface form does not fit to the main topic of the document, all entities within the candidate set are often singletons in the subgraph. In the graph context, a singleton is a vertex without any in- or outgoing edges.

The following text excerpt from a document from the error analysis illustrates this problem. The text deals with the breakup of Justin Timberlake and Cameron Diaz:

> "According to Star, Diaz, 34, spent *Christmas* with her family in Vail, Colo., while Timberlake, 25, was with his family near Memphis."

Here, the surface form *Christmas* is unrelated to the other surface forms in the remainder of the text, such as *Cameron Diaz*. The resulting subgraph in Figure 4.2 illustrates that the correct candidate *dbr:Christmas* is a singleton in the subgraph. Only considering the graph connectivity, the surface form *Christmas* would be linked to the relatively unknown music album *dbr:Christmas_(Plus_One_Album)*, due to the fact that its band shares its hometown with Justin Timberlake's.

#### Multiple Equal Surface Forms

Another drawback of the Graph Linker is the incapability of treating multiple equal surface forms differently. When a text has multiple equal surface forms, the graph-based linker assigns the same annotation to all equal surface forms, since they share the same candidates and consequently the same graph-based score measures. This differs from a linker that uses a context similarity, where the tokens surrounding the surface forms are taken into consideration.

The following text excerpt from a document about cricket from the error analysis illustrates this problem:

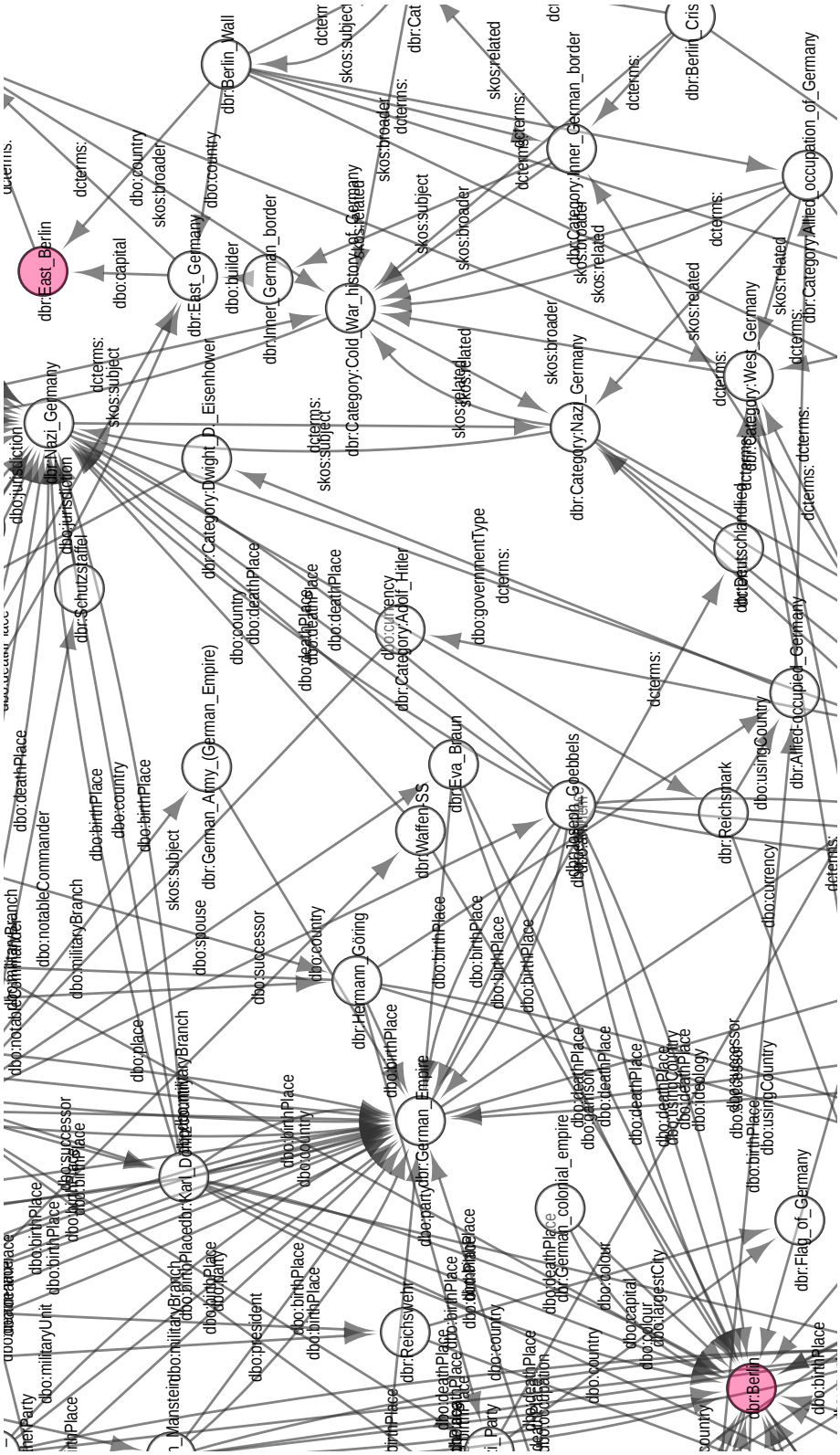


Figure 4.1: Berlin Popular Entity Bias Example

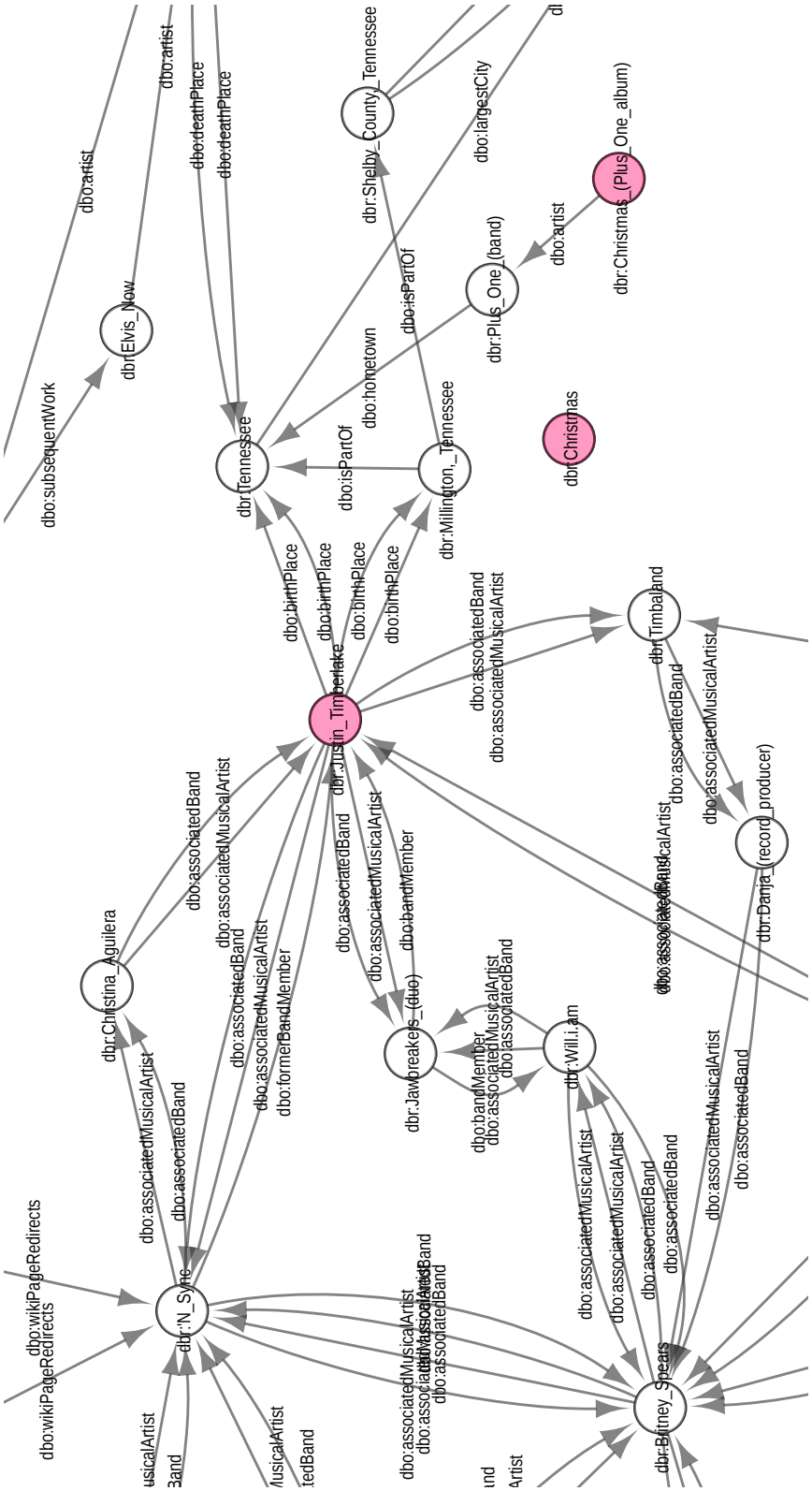


Figure 4.2: Unrelated Domain Surface Forms Example

> "Cricket player Brian Lara suffered a loss on his when playing against *Australia*. The game was held in Canberra, *Australia*."

Here, the gold standard annotates the first occurrence of *Australia* to *dbr:Australia_national_cricket_team* and the second to *dbr:Australia*. The graph-based linker, however, treats both surface forms equally and assigns them to *dbr:Australia*.

#### Short texts

Unlike statistical local features, the graph-based linker requires a certain amount of surface forms and candidates to be able to interrelate them within the subgraph. In the case of short texts, the graph-based linker performs poorly. For example, microposts from microblogging services such as Twitter are limited to 140 characters. The Meij Twitter dataset, that has been discussed in Section 4.2, contains 1.6 annotations per tweet on average. Especially in the case of a single annotation, the subgraph does not contain any edges and thus the graph-based linker cannot derive annotations.

### 4.4.2 Performance

As outlined in the beginning of the chapter, apart from a pure performance evaluation of the Graph Linker, the purpose is to find the best properties for the linker among various design decisions parameters.

On the AIDA/CO-NLL dataset, the best performance is achieved by the DBpedia Graph Linker with the following properties:

- *Graph Construction*: Partial inclusion of the category datasets (using regular expression filters) and the ontology datasets (using $t = 100,000$ as the ontology class frequency threshold)
- *Graph Traversal*: Undirected with a maximum depth of two
- *Candidate Pruning*: Best-7 candidate entities based on prior probability
- *Semantic Relation Weighting*: No weighting
- *Graph Connectivity Measure*: Degree

Using these properties, the resulting D2W performance on all datasets is displayed in Table 4.5. To evaluate the full entity linking process and allow the comparison to the accuracies of other linkers discussed in [9], Table 4.6 shows the Sa2W performance on all datasets. Hereby, DBpedia Spotlight is used for spotting and candidate selection.

In the following, the conducted evaluation for each of the mentioned core aspects of the DBpedia Graph Linker is discussed in dedicated sections. For each considered aspect of the Graph Linker, the performance of different options was

| Dataset | Precision | Recall | $F_1$ |
|---|---|---|---|
| AIDA/CO-NLL | 0.734 | 0.763 | 0.706 |
| IITB | 0.511 | 0.533 | 0.490 |
| MSNBC | 0.764 | 0.811 | 0.722 |
| AQUAINT | 0.627 | 0.708 | 0.563 |

Table 4.5: DBpedia Graph Linker D2W Accuracy

| Dataset | Best Score Threshold | Precision | Recall | $F_1$ |
|---|---|---|---|---|
| AIDA/CO-NLL | 0.391 | 0.677 | 0.524 | 0.591 |
| IITB | 0.195 | 0.587 | 0.186 | 0.283 |
| MSNBC | 0.445 | 0.590 | 0.468 | 0.522 |
| AQUAINT | 0.375 | 0.368 | 0.378 | 0.373 |

Table 4.6: DBpedia Graph Linker Sa2W Accuracy

evaluated. Hereby, for all other settings, the best performing setup mentioned above was used. Therefore, these settings are not included in the evaluation tables that follow.

### Graph Construction

During the graph construction step discussed in Section 3.1, various design decisions were involved.

Regarding the *Selecting DBpedia Datasets* part (cf. Section 3.1.1), evaluating all possible combinations of datasets is not feasible. For most datasets, such as image information, it should be clear that including these datasets is not helpful for disambiguating entities. The aspects considered wortwhile for evaluation are the selection of the infobox properties dataset, and the decision to which extend the category and ontology information should be included. Concerning the selection of the infobox dataset, in the present work the choice was to use the dataset with the highest overall quality, the *Mapping-based Properties*. Another viable option might be to use the *Mapping-based Properties (Cleaned)*, which has a higher coverage but lower overall accuracy. Due to the fact that other evaluation steps revealed that the subgraphs for the documents are already very large and dense, the likelihood that a higher coverage will result in a higher accuracy is considered low. Therefore, due to the limited scope of the thesis, the explicit evaluation of the selection of the mapping-based dataset is omitted.

For the *Triple Filtering* part (cf. 3.1.2), the generic regular expressions to filter out noise such as administrative information are considered to be certainly not helpful. Thus, they are not evaluated on gold standard datasets. The decision whether to exclude, partially include, or fully include the ontology classes and categorical information is more relevant. Initial experiments with undirected traversal and the full inclusion of both datasets lead to computationally explosive traversal. Here,

| Category Inclusion | Ontology Inclusion | Precision | Recall | $F_1$ |
|---|---|---|---|---|
| RegExp Filter | $t = 100000$ Filter | 0.763 | 0.706 | 0.734 |
| RegExp Filter | — | 0.764 | 0.704 | 0.733 |
| — | $t = 100000$ Filter | 0.756 | 0.686 | 0.719 |
| — | — | 0.752 | 0.673 | 0.710 |

Table 4.7: Graph Construction D2W Accuracy AIDA/CO-NLL

the resulting subgraphs were equal to the entire DBpedia graph, since all entities share the class *owl:Thing*. Therefore, using the Graph Linker with the otherwise best performing setting, experiments were conducted with the exclusion or partial inclusion of category and ontology class information. Overall, this results in four experiments, one with the partial inclusion of both, category and ontology information, two with the partial inclusion of one of the two, and one which excludes both.

Table 4.7 displays the resulting accuracy of the mentioned settings. The results show that including the filtered category triples increases the $F_1$ accuracy by 2.3%. Moreover, the inclusion of the partial ontology result in 0.9% increase in $F_1$-measure. Including both, ontology and category triples partially, results in diminishing effects, meaning that instead of an increase of 3.4%, the overall accuracy increases by 2.4%. The fact that the marginal increase of including the category data is a lot higher compared to the ontology data could be due to the fact that the filtering of ontology was performed in a very aggressive manner, effectively excluding 83.8% of the ontology triples. Therefore, further experiments could be conducted with a higher ontology class frequency threshold $t$, e.g. $t = 500000$ instead of the applied $t = 100000$. However, these experiments were omitted due to the limited scope of the thesis.

Overall, the experiments reveal that partially including the ontology class and category information has a positive effect on the overall accuracy.

### Graph Traversal

For graph traversal, the optimal traversal direction and depth is evaluated. Table 4.8 shows the D2W experiments using the viable traversal depth ranges for both directions estimated in Section 3.2. The results show that undirected traversal with $d = 2$ outperforms all directed traversal methods with $d \in [1, 6]$ by more than 3% on the AIDA/CO-NLL dataset. This leads to the conclusion that undirected traversal outperforms directed traversal.

### Candidate Pruning

The necessity of candidate pruning for entity linking is discussed in Section 3.3. In the mentioned chapter, it is concluded that a best-$k$ approach based on prior probability is the best option, with $k \in [3, 10]$.

| Traversal Direction | Traversal Depth | Precision | Recall | $F_1$ |
|---|---|---|---|---|
| Undirected | 2 | 0.763 | 0.706 | 0.734 |
| Directed | 6 | 0.723 | 0.678 | 0.700 |
| Directed | 5 | 0.718 | 0.665 | 0.691 |
| Directed | 4 | 0.721 | 0.649 | 0.683 |
| Directed | 3 | 0.717 | 0.624 | 0.667 |
| Directed | 2 | 0.734 | 0.596 | 0.658 |
| Undirected | 1 | 0.710 | 0.477 | 0.570 |
| Directed | 1 | 0.706 | 0.465 | 0.561 |

Table 4.8: Graph Traversal D2W Accuracy AIDA/CO-NLL

| Candidate Pruning | Precision | Recall | $F_1$ |
|---|---|---|---|
| best 7 (prior) | 0.763 | 0.706 | 0.734 |
| best 6 (prior) | 0.762 | 0.705 | 0.733 |
| best 8 (prior) | 0.760 | 0.705 | 0.732 |
| best 10 (prior) | 0.759 | 0.704 | 0.731 |
| best 3 (prior) | 0.764 | 0.697 | 0.729 |
| best 5 (prior) | 0.759 | 0.699 | 0.728 |
| best 20 (prior) | 0.743 | 0.692 | 0.717 |

Table 4.9: Candidate Pruning D2W Accuracy AIDA/CO-NLL

Therefore, experiments with different values for $k$ were conducted. The results in Table 4.9 show that the best results are achieved with $k = 7$. At this point, the optimum between the trade-off of precision versus accuracy is found. From $k = 7$, the more the value for $k$ decreases, the lower the recall becomes, since the likelihood increases that the correct entity is not contained in the best-$k$ list. In the same fashion, a further increase in $k$ results in a decrease in precision, since the graph becomes more noisy.

The accuracy for $k = 20$, the default value that Spotlight uses, is added in Table 4.9 as a baseline. Here, the difference of 1.7% in $F_1$ compared to $k = 7$ shows the improvements received from applying restrictive candidate pruning.

### Semantic Relation Weighting

As part of the evaluation process, the relation weighting metrics discussed in Section 3.4 are evaluated.

The results in Table 4.10 show that, for each algorithm $JointIC$, $CombIC$, and $IC\_PMI$, the overall precision and recall slightly decrease. Since all of the evaluated measures are based on information content, which favors infrequent relations, this leads to the conclusion that specificity is not a good proxy for relevance in the present context.

This conclusion can be illustrated by the example of *dbr:Michael_Jordan*.

| Semantic Relation Weighting | Precision | Recall | $F_1$ |
|---|---|---|---|
| $JointIC$ | 0.761 | 0.705 | 0.732 |
| $CombIC$ | 0.761 | 0.704 | 0.732 |
| $IC\_PMI$ | 0.760 | 0.702 | 0.730 |

Table 4.10: Semantic Relation Weighting D2W Accuracy AIDA/CO-NLL

| Graph Connectivity Measure | Precision | Recall | $F_1$ |
|---|---|---|---|
| *Degree* | 0.763 | 0.706 | 0.734 |
| *Global Degree* | 0.748 | 0.719 | 0.733 |
| *PageRank* (52 iterations, $\alpha = 0.85$) | 0.756 | 0.699 | 0.727 |
| *KPP* | 0.692 | 0.641 | 0.665 |
| *HITS* (20 iterations, $\alpha = 0$) | 0.684 | 0.633 | 0.658 |
| *Normalized Degree* | 0.445 | 0.412 | 0.428 |

Table 4.11: Graph Connectivity Measure D2W Accuracy AIDA/CO-NLL

Michael Jordan is most famous for being a basketpall player who played for the Chicago Bulls team. When looking at his connections in the DBpedia graph, there are edges that depict the mentioned characteristics:

*dbr:Michael_Jordan rdf:type dbo:BasketballPlayer.*
*dbr:Michael_Jordan dcterms:subject category:Chicago_Bulls_players.*

However, the semantic relation weighting scores for both statements, especially the fact that Michael Jordan is a basketball player, are rather low. For example, the $jointIC$ measure for both statements equals $8.65$ and $11.68$, respectively.

In contrast, the $jointIC$ measures for the following statement is very high ($13.67$):

*dbr:Michael_Jordan dcterms:subject category:People_from_Highland_Pa–rk,_Illinois.*

For disambiguation, the fact that Michael Jordan origins from the Highland Park area in Illinois is far less important than the fact that he played for Chicago Bulls. Overall, the example illustrates that the applied semantic relation weighting measures do not resemble the human understanding of important facts about a topic. Thus, they are hardly suited to emphasize the key facts related to an entity.

### Graph Connectivity Measure

The graph-based connectivity measures, which are applied on the subgraph connecting candidate entities, are described in Section 3.5.

Table 4.11 shows the results for evaluating the different algorithms in the otherwise best setup. As illustrated, the *Degree Centrality* measure performs best

among all evaluated algorithms. This finding is shared with the evaluation conducted in [32], where the authors report in their experiments that Degree performs best among all their evaluated graph connectivity measures.

Apart from the algorithms in [32], the performance of the adaptions of the Degree Centrality measure differ greatly. The *Global Degree* measure, which serves as another baseline, performs almost as good as the Degree algorithm, and exceeds its recall by more than 1%. The high recall is due to the fact that this algorithm does not suffer from the *Unrelated Domains Surface Forms* issue discussed in Section 4.4.1. In contrast, the *Normalized Degree* algorithm, which tries to mitigate the discussed *Popularity Bias* issue, performs poorly. Here, the assumption is that due to the high accuracy of the popularity measure, moving away from this measure results in decreasing accuracy.

## 4.5 DBpedia Spotlight Linker

Following the methodology of Section 4.4, an error analysis and performance evaluation is also conducted for DBpedia Spotlight. In the following, both evaluation perspectives are discussed in dedicated sections.

### 4.5.1 Error Analysis

The error analysis of DBpedia Spotlight was performed to outline typical errors done by a state of the art linker. The goal was to find hard cases where supervised approaches do not perform well, and consequently deduct patterns where the graph-based approach might fit in.

As methodology, 20 random documents were sampled from the Test-B chunk of the AIDA/CO-NLL dataset. This resulted in the following set of documents, all represented by their document id:

$$\{1174, 1176, 1182, 1183, 1184, 1193, 1206, 1212, 1231, 1242,$$
$$1262, 1274, 1300, 1316, 1317, 1320, 1335, 1363, 1364, 1368\}$$

Using this set, a D2W evaluation was conducted using the BAT framework. Then, the resulting Spotlight and Gold Standard annotations were compared. In the case of incorrect or missing annotations, the entity candidates produced by Spotlight were determined for each surface form. Thereby, the score of each Spotlight feature was analyzed to find out which feature had the highest contribution in assigning the incorrect or NIL annotation.

Table 4.12 shows the distribution of errors in the considered document set. Hereby, *Process Step* indicates in which step of the linking process the error occurred. Since the D2W task is applied, there are no errors in spotting. Of the 41 errors, 23 (56.1%) amount to candidate selection errors, 15 (36.6%) are disambiguation errors and 3 (7.3%) are due to coreference resolution errors. Due to the small sample size, these values are not significant. However, the disambiguation

| Process Step | Error Reason | Count |
|---|---|---|
| Candidate Selection | Empty candidate set | 8 |
| | GS entity not in best-20 (by prior) | 4 |
| | GS entity not in candidate set | 11 |
| Disambiguation | Misleading Prior score | 3 |
| | Misleading Context score | 10 |
| | Misleading Popularity score | 2 |
| Coreference resolution | Propagation of incorrect entity | 3 |

Table 4.12: DBpedia Spotlight Analysis Error Distribution

errors suggest that the context score might be too dominant. This suggestion is supported by the fact that in the current version of DBpedia Spotlight, the feature probabilities are multiplied in an unweighted manner. Therefore, Spotlight might benefit from training feature weights and generating the final score using an appropriate model.

With regards to the document characteristics, the finding of the error analysis was that mistakes occurred mostly within articles in the Sports domain. In this domain there are many challenging cases for a linker, such as the distinguishing between a country and a national team of a country. To illustrate this case, the following text excerpt has been taken from a document about Badminton results:

> "Chen Gang( *China*) beat Martin Londgaard Hansen( *Denmark*) 15-12 15-6."

Here, the gold standard links the surface forms *China* and *Denmark* to their respective country articles, whereas Spotlight annotates China as *China_national_badminton_team* and Denmark as *Denmark_open_(badminton)*. This is due to the fact that the context similarity outvotes the popularity and prior probability features, which would correctly annotate the surface forms to their countries. However, the excerpt also illustrates that such cases are very specific, and often the Spotlight annotation could also be considered correct, as it is the case with the *China_national_badminton_team*.

Overall, the error analysis did not reveal general drawbacks of the statistical approach.

### 4.5.2 Performance

Regarding the linking performance of DBpedia Spotlight, Table 4.13 shows the D2W and Table 4.14 the Sa2W accuracies. Hereby, the statistical backend of version 0.6 was used, with default spotting and disambiguation settings. Since the default $k$ for pruning the candidate set on a best-$k$ basis was changed from $k = 20$ to $k = 10$ from version 0.6 to version 0.7, experiments with both settings were conducted. The Sa2W results are included to allow the comparison with the evaluation in [9].

| Dataset | Candidate Pruning | Precision | Recall | $F_1$ |
|---|---|---|---|---|
| AIDA/CO-NLL | best-20 | 0.827 | 0.792 | 0.809 |
| | best-10 | 0.833 | 0.796 | 0.814 |
| IITB | best-20 | 0.614 | 0.603 | 0.608 |
| | best-10 | 0.631 | 0.620 | 0.625 |
| MSNBC | best-20 | 0.825 | 0.766 | 0.794 |
| | best-10 | 0.830 | 0.771 | 0.799 |
| AQUAINT | best-20 | 0.863 | 0.825 | 0.844 |
| | best-10 | 0.865 | 0.827 | 0.845 |

Table 4.13: DBpedia Spotlight D2W Accuracy

| Dataset | Candidate Pruning | Best Score Threshold | Precision | Recall | $F_1$ |
|---|---|---|---|---|---|
| AIDA/CO-NLL | best-20 | 0.531 | 0.692 | 0.576 | 0.629 |
| | best-10 | 0.531 | 0.702 | 0.585 | 0.638 |
| IITB | best-20 | 0.383 | 0.626 | 0.215 | 0.320 |
| | best-10 | 0.383 | 0.636 | 0.218 | 0.325 |
| MSNBC | best-20 | 0.828 | 0.597 | 0.483 | 0.534 |
| | best-10 | 0.828 | 0.594 | 0.486 | 0.535 |
| AQUAINT | best-20 | 0.523 | 0.411 | 0.477 | 0.442 |
| | best-10 | 0.523 | 0.410 | 0.477 | 0.441 |

Table 4.14: DBpedia Spotlight Sa2W Accuracy

The results show that DBpedia Spotlight outperforms the baseline linker $P(s|e)$ (cf. Section 4.3) on the AIDA/CO-NLL dataset by more than 10%. Moreover, on the MSNBC dataset, Spotlight outperforms the baseline by 3%, whereas on IITB and AQUAINT, Spotlight performs worse than the baseline by 12.4% and 1.2%, respectively. The rather unaccurate result on the IITB dataset might be due to the fact that the context-based feature, which is trained on Wikipedia, does not work well on web pages.

## 4.6 Federated Linker

For the evaluation of the federated linkers, the chapter is structured according to the respective methodology Section 3.6. Thus, the *Enhanced-graph based Linker*, the *System Combination Linker*, and the *Feature Combination Linker* are addressed individually.

### 4.6.1 Enhanced Graph-based Disambiguation

As discussed in Section 3.6.1, the problem of unrelated surface forms is addressed by introducing a fallback strategy for cases where all candidate entities are single-

| Graph Connectivity Measure | Candidate Pruning | Precision | Recall | $F_1$ |
|---|---|---|---|---|
| Degree with Prior Fallback | best 7 (prior) | 0.758 | 0.728 | 0.743 |
| Degree with Prior Fallback | best 6 (prior) | 0.756 | 0.727 | 0.741 |
| Degree with Prior Fallback | best 3 (prior) | 0.756 | 0.726 | 0.741 |
| Degree with Prior Fallback | best 10 (prior) | 0.755 | 0.726 | 0.74 |
| Degree with Prior Fallback | best 5 (prior) | 0.753 | 0.724 | 0.738 |

Table 4.15: Enhanced DBpedia Graph Linker D2W Accuracy AIDA/CO-NLL

tons in the subgraph. In these cases, the prior probability scores are assigned as similarity measures to each candidate entity.

Table 4.15 shows the results for the fallback strategy. Thereby, different candidate pruning measures were evaluated, to see if this setting affects the accuracy in a different way than it does in the regular DBpedia Graph Linker. Compared to the regular DBpedia Graph Linker evaluation results from Section 4.4, the accuracy of the best performing setting improves by nearly 1%, even though precision drops by 0.5%. The 2.2% increase in recall is due to the fact that assigning no annotation for only singletons is equal to an incorrect annotation for the D2W task. Therefore, using a prior as fallback helps to mitigate the unrelated surface form issue. Regarding candidate pruning, the best result is also achieved in a best-7 setting.

Overall, the enhanced Graph Linker does not reach the accuracy of DBpedia Spotlight. This is due to the fact that not all drawbacks discussed in Section 4.4.1 could be eliminated.

### 4.6.2 System Combination

The methodology on how to combine DBpedia Spotlight and the DBpedia Graph Linker on a system level is discussed in Section 3.6.2. For learning the linear regression coefficients $\beta_i$ and the error variable $\alpha$, the training subset of the AIDA corpus, discussed in Section 4.2, was used.

Table 4.16 shows the evaluation results for various settings, including initial experiments with manually selected weights. In this table, the "Candidate without $P_g(e)$ Strategy" column refers to the strategy applied to deal with the candidate selection differences discussed in Section 3.6.2. Moreover, different values for the candidate pruning measure $k$ were evaluated. As can be seen, the accuracy among the different strategies used in combination with the trained weights only differ very slightly, by up to 0.01%. This is assumed to be due to the fact that with the switch from $k = 20$ to $k = 10$ from DBpedia Spotlight version 0.6 to 0.7, the $\Delta_k := k_{stat} - k_{graph}$ got very low, with $\Delta_k = 3$ for version 0.6 compared to $\Delta_k = 13$ in version 0.7. With these similar settings for $k$, the set of candidates is almost equal among both linkers.

Overall, the system combination linker with the trained weights outperforms the DBpedia Spotlight linker by 0.7% in precision, recall, and $F_1$-measure. Although the improvement is not substantial, it shows that the statistical implemen-

| $\alpha$ | $\beta_s$ | $\beta_g$ | Spotlight Candidate Pruning | Candidate without $P_g(e)$ Strategy | Candidate Pruning | Precision | Recall | $F_1$ |
|---|---|---|---|---|---|---|---|---|
| 0 | 0.445 | 0.388 | best10 | Omit weighted combination | best 10 (prior) | 0.840 | 0.804 | 0.822 |
| 0 | 0.445 | 0.388 | best10 | $P_g(e) = 0$ heuristic | best 10 (prior) | 0.840 | 0.804 | 0.822 |
| 0 | 0.445 | 0.388 | best10 | Omit weighted combination | best 7 (prior) | 0.840 | 0.803 | 0.821 |
| 0 | 0.445 | 0.388 | best10 | $P_g(e) = 0$ heuristic | best 7 (prior) | 0.840 | 0.803 | 0.821 |
| 0 | 0.65 | 0.35 | best10 | Omit weighted combination | best 10 (prior) | 0.837 | 0.801 | 0.819 |
| 0 | 0.60 | 0.40 | best20 | Omit weighted combination | best 3 (prior) | 0.831 | 0.796 | 0.813 |
| 0 | 0.60 | 0.40 | best20 | Omit weighted combination | best 7 (prior) | 0.83 | 0.795 | 0.812 |
| 0 | 0.60 | 0.40 | best20 | Omit weighted combination | best 4 (prior) | 0.83 | 0.795 | 0.812 |
| 0 | 0.60 | 0.40 | best20 | Omit weighted combination | best 2 (prior) | 0.829 | 0.794 | 0.811 |
| 0 | 0.65 | 0.35 | best10 | $P_g(e) = 0$ heuristic | best 10 (prior) | 0.829 | 0.793 | 0.811 |

Table 4.16: System Combination Linker D2W Accuracy

| $\alpha$ | $\beta_{P(e)}$ | $\beta_{P_{norm}(c\|e)}$ | $\beta_{P(s\|e)}$ | $\beta_{P(g\|e)}$ | Precision | Recall | $F_1$ |
|---|---|---|---|---|---|---|---|
| 0.0870 | 186.8943 | 0.3146 | 0.1556 | 0.1841 | 0.824 | 0.788 | 0.805 |

Table 4.17: Feature Combination Linker D2W Accuracy

| Linker | Precision | Recall | $F_1$ |
|---|---|---|---|
| *System Combination* | 0.840 | 0.804 | 0.822 |
| *DBpedia Spotlight* | 0.833 | 0.796 | 0.814 |
| *Feature Combination* | 0.824 | 0.788 | 0.805 |
| *Enhanced DBpedia Graph Linker* | 0.758 | 0.728 | 0.743 |
| *DBpedia Graph Linker* | 0.763 | 0.706 | 0.734 |
| $P(s\|e)$ *Baseline Linker* | 0.715 | 0.687 | 0.701 |
| *AGDISTIS Linker* | 0.642 | 0.556 | 0.596 |

Table 4.18: Evaluation Linkers D2W Accuracy AIDA/CO-NLL

tation benefits from a graph-based addition.

### 4.6.3 Feature Combination

For combining the features of both linkers, a multiple linear regression method was applied (cf. Section 3.6.3).

Table 4.17 shows the learned weights using the training chunk of the AIDA/CO-NLL dataset. Here, the $\beta_{P(e)}$ coefficient exceeds the other coefficients by several orders of magnitude. This is due to the fact that the popularity score of an entity is very low, since the indegree is normalized by the number of all links within Wikipedia.

The displayed performance measures in the mentioned Table 4.17 show that the feature combination performs worse than the best system combination linker, with 1.6% less precision and recall. This leads to the conclusion that multiple linear regression is not an appropriate model for combining the feature scores.

## 4.7 Findings

This section summarizes the findings of the overall evaluation. Hereby, a section is dedicated to each of the following linkers: the *DBpedia Graph Linker*, the *Baseline Linkers* including DBpedia Spotlight, and the *Federated Linkers*. Table 4.18 shows the accuracy of all evaluated linkers.

### 4.7.1 DBpedia Graph Linker

For the linker that has been developed as part of the present thesis, all aspects regarding graph-based disambiguation have been evaluated individually on the AIDA/CO-NLL dataset:

- Regarding *Graph Construction*, the basis was formed by the infobox properties data. On top of that, the best results were achieved by partial inclusion of the category datasets (using regular expression filters) and the ontology datasets (using $t = 100,000$ as the ontology class frequency threshold).
- For *Graph Traversal*, using undirected traversal with a maximum depth of two yielded the best results.
- An important finding of initial experiments revealed the necessity of pruning the set of candidate entities for a surface form (cf. Section 3.3). Hereby, the best *Candidate Pruning* setting was found to be a best-7 filter based on the candidate entities prior probability.
- Regarding *Semantic Relation Weighting*, the evaluated measures adapted from [36], which are based on the concept of Information Content, resulted in a decrease in accuracy.
- For applying the *Graph Connectivity Measures* on the subgraph resulting from the graph traversal step, the most straightforward algorithm, *Degree Centrality*, showed the best results.

Using the mentioned properties, the DBpedia Graph Linker achieves 73.4% in $F_1$-measure, as displayed in Table 4.18.

### 4.7.2 Baseline Linkers

To compare the Graph Linker to other linkers, three different baselines have been evaluated. As shown in Table 4.18, the Graph Linker developed in the present work outperforms the *AGDISTIS* linker – to the authors knowledge the only other linker that follows a similar approach to the Graph Linker – by more than 12% regarding precision and recall. Moreover, the Graph Linker outperforms the $P(s|e)$ baseline linker by 3.3% with regards to $F_1$-measure. Compared to a state of the art statistical linker, the Graph Linker's precision and recall is lower than DBpedia Spotlight's by at least 7%.

### 4.7.3 Federated Linkers

In Section 4.6, three different federated linker approaches have been evaluated. The results in Table 4.18 show that the *Enhanced Graph-based Linker* performs worse than DBpedia Spotlight by more than 7% in precision and recall. This is due to the fact that no method was found which eliminates all drawbacks discussed in Section 4.4.1.

Overall, the *System Combination* method is the only linker that outperforms DBpedia Spotlight. As discussed in Section 4.6.3, the unsatisfactory results of the *Feature Combination* linker is due to the fact that linear regression is unsuited for a probabilistic model. Here, an appropriate model, capable of dealing with the log-scaled feature scores of Spotlight, would likely produce better results.

# Chapter 5

# Conclusion

In the remainder of the chapter, the present thesis is concluded by first providing an overview of the main contributions, followed by an outline of promising future work.

## 5.1 Overview

In the present thesis, a graph-based entity linker exploiting DBpedia has been developed. Hereby, all steps relevant in designing a graph-based linker have been analyzed. For constructing a suitable graph, the selection of appropriate DBpedia datasets and the filtering of irrelevant triples have been evaluated. With regards to finding connections between entities and forming a subgraph of the resulting paths, experiments were conducted to find the optimal traversal direction and limit. To cope with the vast number of potential entities for a surface form, different candidate set pruning techniques have been analyzed. Moreover, approaches for weighting semantic relations in the graph have been adapted from existing research. To come up with a ranked list of entities, various graph-based connectivity measures have been evaluated.

Overall, the extensive set of experiments lead to a linker that clearly outperforms the commonly applied prior probability baseline. However, a conducted error analysis revealed general drawbacks of graph-based methods for entity linking, such as the incapability of dealing with surface forms from different domains. Therefore, the graph-based linker does not perform as well as state of the art statistical approaches. Nevertheless, the graph-based linker excels in situations where many surface forms fit into a single overall domain, such as news about a football team.

Among the analyzed state of the art systems, DBpedia Spotlight is the only system that does not exploit a global feature for disambiguation. The Graph Linker uses an approach similar to the one of a global feature, where the disambiguation of one surface form depends on the candidate entities of other surface forms. Therefore, different approaches for combining the Graph Linker with DBpedia Spotlight

have been evaluated. Using a straightforward weighted linear combination that combines the ranked list of candidate entities of both systems, an overall increase in precision and recall was measured. This proof of concept shows that supervised methods for entity linking can benefit from incorporating graph-based approaches.

## 5.2 Future Work

In future work, the conducted research could be extended in many directions.

Regarding the DBpedia Graph Linker, further research could be targeted at improving each step in the development process of the linker. In particular, the application of less generic weighting schemes, which are learned based on the disambiguation task, seems promising. Considering the graph-based connectivity measures, more algorithms could be evaluated, to find measures that outperform the Degree Centrality algorithm.

The focus of the present work was to construct a graph-based linker with the highest accuracy possible. Therefore, computational performance aspects were neglected. Consequently, DBpedia Spotlight, which has been designed as a fast performing linker, is orders of magnitude faster than the graph-based approach. To reach a performance of the Graph Linker applicable for a real-time system, future research could be targeted at improving the traversal algorithm adapted from [32], so that edges are not explored multiple times. Other performance improvements could be gained by shrinking the DBpedia graph, while stile maintaining a high disambiguation accuracy. To this end, automated methods need to be developed which cut off all edges in the graph that do not significantly aid in disambiguation. This task is closely connected to the mentioned development of more sophisticated weighting schemes, since it requires finding an estimate for the relevance of each edge in the graph for the disambiguation task.

For further improvements in enhancing DBpedia Spotlight with a graph-based feature, a deeper understanding of all features in question would be beneficial. To this end, an analysis could be conducted to determine how the features are correlated with each other, and how their scores are distributed over an entire corpora. Based on the results of this analysis, an appropriate model for combining all features in a weighted manner could be determined.

# Bibliography


[1] Domingo De Abreu, Alejandro Flores, Guillermo Palma, Valeria Pestana, Jonathan Queipo, and Maria-esther Vidal. Choosing Between Graph Databases and RDF Engines for Consuming and Mining Linked Data. 2013.

[2] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web SE - 52*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735. Springer Berlin Heidelberg, 2007.

[3] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. DBpedia - A crystallization point for the Web of Data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154–165, 2009.

[4] Stephen P. Borgatti. Identifying sets of key players in a social network. *Computational & Mathematical Organization Theory*, 12(1):21–34, 2006.

[5] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[6] Razvan C. Bunescu and Marius Pasca. Using Encyclopedic Knowledge for Named entity Disambiguation. *EACL*, pages 9–16, 2006.

[7] Yee Seng Chan and Dan Roth. Exploiting Syntactico-semantic Structures for Relation Extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 551–560, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[8] Xiao Cheng and Dan Roth. Relational Inference for Wikification. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2013.

[9] Marco Cornolti, Paolo Ferragina, and Massimiliano Ciaramita. A Framework for Benchmarking Entity-annotation Systems. In *Proceedings of the 22Nd International Conference on World Wide Web*, WWW '13, pages 249–260, Republic and Canton of Geneva, Switzerland, 2013. International World Wide Web Conferences Steering Committee.

[10] Silviu Cucerzan. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *EMNLP-CoNLL*, number June, pages 708–716, 2007.

[11] Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N Mendes. Improving Efficiency and Accuracy in Multilingual Entity Extraction. In *Proceedings of the 9th International Conference on Semantic Systems*, I-SEMANTICS '13, pages 121–124, New York, NY, USA, 2013. ACM.

[12] P Ferragina and U Scaiella. Fast and Accurate Annotation of Short Texts with Wikipedia Pages. *Software, IEEE*, 29(1):70–75, 2012.

[13] Paolo Ferragina and Ugo Scaiella. TAGME: On-the-fly Annotation of Short Text Fragments (by Wikipedia Entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1625–1628, New York, NY, USA, 2010. ACM.

[14] Linton C Freeman. Centrality in social networks conceptual clarification. *Social Networks*, 1(3):215–239.

[15] Evgeniy Gabrilovich and Shaul Markovitch. Overcoming the brittleness bottleneck using Wikipedia: Enhancing text categorization with encyclopedic knowledge. *AAAI*, 6:1301–1306, 2006.

[16] Xianpei Han and Le Sun. A Generative Entity-mention Model for Linking Entities with Knowledge Base. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 945–954, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[17] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. Robust Disambiguation of Named Entities in Text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 782–792, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[18] Ioana Hulpus, Conor Hayes, Marcel Karnstedt, and Derek Greene. Unsupervised Graph-Based Topic Labelling using DBpedia. In *Proceedings of the the Sixth ACM Conference on Web Search and Data Mining, WSDM 2013*, 2013.

[19] Jon M Kleinberg. Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM*, 46(5):604–632, September 1999.

[20] Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. Collective Annotation of Wikipedia Entities in Web Text. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 457–466, New York, NY, USA, 2009. ACM.

[21] Claudia Leacock and Martin Chodorow. Combining Local Context and WordNet Similarity for Word Sense Identification. *WordNet: An Electronic Lexical Database*, pages 265–283, 1998.

[22] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*. Number c. Cambridge University Press, 2008.

[23] Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[24] Frank Manola, Eric Miller, and Brian McBride. RDF 1.1 Primer, 2014.

[25] Olena Medelyan, IH Witten, and David Milne. Topic indexing with Wikipedia. *Proceedings of the AAAI WikiAI workshop*, pages 19–24, 2008.

[26] Edgar Meij, Wouter Weerkamp, and Maarten de Rijke. Adding Semantics to Microblog Posts. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, WSDM '12, pages 563–572, New York, NY, USA, 2012. ACM.

[27] Pablo Mendes, Joachim Daiber, Rohana Rajapakse, Felix Sasaki, and Christian Bizer. Evaluating the Impact of Phrase Recognition on Concept Tagging. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA).

[28] Pablo N Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. DBpedia Spotlight: Shedding Light on the Web of Documents. In *Proceedings of the 7th International Conference on Semantic Systems (I-Semantics)*, I-Semantics '11, pages 1–8, New York, NY, USA, 2011. ACM.

[29] Rada Mihalcea and Andras Csomai. Wikify!: Linking Documents to Encyclopedic Knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 233–242, New York, NY, USA, 2007. ACM.

[30] David Milne and Ian H Witten. An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links. In *In Proceedings of AAAI 2008*, 2008.

[31] David Milne and Ian H Witten. Learning to Link with Wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 509–518, New York, NY, USA, 2008. ACM.

[32] Roberto Navigli and Mirella Lapata. An experimental study of graph connectivity for unsupervised word sense disambiguation. *IEEE transactions on pattern analysis and machine intelligence*, 32(4):678–92, April 2010.

[33] Lev Ratinov and Dan Roth. Design Challenges and Misconceptions in Named Entity Recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, pages 147–155, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[34] Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. Local and Global Algorithms for Disambiguation to Wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1375–1384, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[35] Gert Sabidussi. The centrality index of a graph. *Psychometrika*, 31(4):581–603, 1966.

[36] Michael Schuhmacher and Simone Paolo Ponzetto. Knowledge-based Graph Document Modeling. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM '14, pages 543–552, New York, NY, USA, 2014. ACM.

[37] Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. LINDEN: Linking Named Entities with Knowledge Base via Semantic Knowledge. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 449–458, New York, NY, USA, 2012. ACM.

[38] Michael Strube and Simone Paolo Ponzetto. WikiRelate! Computing semantic relatedness using Wikipedia. *AAAI*, 6:1419–1424, 2006.

[39] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. YAGO : A Core of Semantic Knowledge Unifying WordNet and Wikipedia. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA, 2007. ACM.

[40] Erik F Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-independent Named Entity Recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 142–147, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[41] Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Sören Auer, Daniel Gerber, and Andreas Both. AGDISTIS - Agnostic Disambiguation of Named Entities Using Linked Open Data. In *Submitted to 14th Conference of the 2, Gothenburg, Sweden, 26-30 April 2014*, page TBA, 2013.

[42] Mohamed Amir Yosef, Johannes Hoffart, Ilaria Bordino, Marc Spaniol, and Gerhard Weikum. AIDA : An Online Tool for Accurate Disambiguation of Named Entities in Text and Tables. In *VLDB*, pages 1450–1453, 2011.

# Appendix A

# Implementation Details

The source code for the DBpedia Graph Linker developed in this work is publicly available in a Github repository[1]. It is implemented as a Java project that runs graph-based linking algorithms. The Graph Linker takes a set of surface forms as input, along with a set of candidate entities for each surface form. As output, the best-k entities are returned, each with an assigned score in the range $[0, 1]$.

To support the entire entity linking process, the DBpedia Spotlight source code has been forked and extended[2]. In the resulting linker, Spotlight performs spotting and candidate selection itself and the DBpedia Graph project is used for disambiguation. Detailed instructions on how to use both projects are documented on the README page[3] of the DBpedia Graph project.

In the remainder of the chapter, a section is dedicated to the utilized graph database, the core technology in the overall project. Moreover, the BAT framework is shortly discussed, since it was used throughout the evaluation in Chapter 4.

## A.1 Graph Database

For constructing and navigating the DBpedia graph in this work, a graph database was utilized. Among the different graph computing technologies, such as in-memory graph toolkits and graph libraries, this technology is best suited for a developing a real-time system based on a large graph.

To remain independent from a concrete graph database vendor, throughout the project the Blueprints project[4] is used. Blueprints is a Java library which provides a common set of interfaces wrapping the different implementations of graph databases. This allows to plug-and-play Blueprints-supported graph databases, such as Neo4j, OrientDB, DEX, or Titan, using the same code base[5]. The Blueprints

[1] https://github.com/bernhardschaefer/dbpedia-graph
[2] https://github.com/bernhardschaefer/dbpedia-spotlight
[3] https://github.com/bernhardschaefer/dbpedia-graph/blob/master/README.md
[4] http://blueprints.tinkerpop.com/
[5] https://github.com/tinkerpop/blueprints

project refers to itself as ”the JDBC for graph databases”[6]. Throughout the evaluation, Blueprints was configured to run with the embedded version of Neo4j[7], the most prominent graph database. This decision is supported by the evaluation of different graph databases and RDF engines in [1], where Neo4j shows the highest performance for BFS and DFS traversal.

## A.2 BAT Framework

As mentioned in Chapter 4, the BAT framework was used for all experiments conducted in this work (cf. [9]). Thereby, in accordance with its author, the BAT framework was extended in the following areas:

- Including the possibility to customize host, port, and disambiguator settings for the Spotlight annotator[8][9].
- Add native D2W support for the Spotlight annotator[10].
- Add an annotator plugin to enable support for the AGDISTIS linker[11].

For the conducted experiments, a branch titled “experiments” was created in the fork of the BAT framework[12]. This branch contains the classes that were used for running the D2W ($SpotlightBatchD2W$) and Sa2W ($SpotlightBatchSa2W$) experiments.

[6] https://github.com/tinkerpop/blueprints/wiki
[7] http://www.neo4j.org/
[8] https://github.com/marcocor/bat-framework/commit/a2cf16bd6a66cf2d2c5fa65332798c744ab74146
[9] https://github.com/marcocor/bat-framework/commit/ac093efc522282e3b0973a81b567a41dc6942e74
[10] https://github.com/marcocor/bat-framework/commit/ac093efc522282e3b0973a81b567a41dc6942e74
[11] https://github.com/marcocor/bat-framework/commit/f1bf74b97132bd168207a3e677fd4d0f98d2107b
[12] https://github.com/bernhardschaefer/bat-framework/tree/experiments

## Ehrenwörtliche Erklärung

Ich versichere, dass ich die beiliegende Masterarbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Mannheim, den 28.05.2014 Unterschrift