Bauhaus-Universität Weimar
Faculty of Media
Degree Programme Media Informatics (Medieninformatik)

Entity-Based Query Interpretation

Bachelor's Thesis

Marcel Gohsen

1. Referee: Prof. Dr. Benno Stein
2. Referee: Prof. Dr. Matthias Hagen

Submission date: June 25, 2018

Declaration

Unless otherwise indicated in the text or references, this thesis is entirely the product of my own scholarly work.

Weimar, June 25, 2018

. . . . . .
Marcel Gohsen

Abstract

The goal of this thesis is to define precise and distinct problem statements for traditional definitions of entity linking in queries and query interpretation. Therefore, we introduce the problems of explicit and implicit entity recognition which are the basis for the task of entity-based query interpretation. For the redefined problems we create a corpus containing altogether 2068 queries. These queries were gathered from commonly used entity linking datasets. Our corpus provides manually annotated explicit and implicit entities as well as query interpretations.

Moreover, we develop algorithmic approaches for automatically solving the problems of explicit and implicit entity recognition with a focus on recall and efficiency. These algorithms are meant for developing a foundation for a query interpretation system that could be built on top of them.

Contents

Chapter 1

Introduction

Understanding a query is fundamental for a web search engine when trying to satisfy a user's information need. According to Guo et al., 70% of the queries of a search engine today contain named entities [Guo et al., 2009] which are defined as objects from the real world that can be referred to by a proper name (i.e., a unique identifier for its representative). Thus, identifying and disambiguating entities can help to find possible interpretations of a user's intent for a majority of queries.

Most entities can be represented by their related entries in a knowledge base like Wikipedia. Linking an entity to the most likely candidate in some knowledge base is commonly referred to as *entity linking* [Hasibi et al., 2015]. Entity linking in queries is usually realized following the guidelines of the ERD'14 Challenge [Carmel et al., 2014]. This challenge defines, that a segment of a query (i.e., a sequence of consecutive query terms) has to be a proper noun to be considered as an candidate for an entity mention. While common nouns (e.g., `planet`) usually refer to a group of entities, proper nouns (e.g., `Jupiter`) primarily refer to a unique entity. Except that condition, named entities require to fit in a set of predefined classes. Typical classes of named entities are persons (e.g., `Barack Obama`), geographic locations (e.g., `Minnesota`), or organizations (e.g., `Google`). For example, the named entity recognizer of the Stanford Natural Language Processing Toolkit [Manning et al., 2014] classifies named entities as persons, locations, organizations, miscellaneous, and optionally numeric expressions (e.g., `13.37%`). However, "miscellaneous" is rather broad and therefore contains instances that are too general to be considered as named entities in queries. We will base our entity linking approach on a taxonomy derived from the "extended named entity hierarchy" [Sekine et al., 2002]. This taxonomy consists of 8 main classes (e.g., *Organization*) and 108 specialized subclasses (e.g., *Sports League*) (cf. Section 3.1 for a detailed description).

Current definitions of entity linking are somewhat imprecise and mix dif-

ferent subtasks. As our first contribution, we formalize entity linking as two distinct computational problems wherein we differentiate between *explicit* and *implicit entities*. For instance, an explicit entity in the query `barack obama mother` is `Barack Obama` whereas `Ann Dunham` (the mother of Barack Obama) is an implicit entity. The most important usage of these entities for our work is for the task of *entity-based query interpretation*, which is a problem precisely defined further in this work as a major task.

Today's commonly used datasets for evaluation of entity linking were developed for settings incompatible with our definition of this problem. Further issue of these datasets is that some of them contain insufficient numbers of queries for a meaningful evaluation of entity linking frameworks. For example, the current available dataset for the short text entity linking task of the ERD'14 Challenge [Carmel et al., 2014] only contains 91 queries, of which about 50% not even contain entities. Remember that Guo et al. found that about 70% of the queries contain entities [Guo et al., 2009]. Hasibi et al. [Hasibi et al., 2017b] used crowdsourcing to gather annotations for their larger corpus for entity linking but its quality suffers from inconsistencies and questionable ratings. For example, for the query `john lennon parents` the entity `Julia Lennon` (his mother) is rated as highly relevant, while `Alfred Lennon` (his father) is only rated as relevant although both parents were requested. Furthermore, of the observed datasets only the smallest from the ERD'14 Challenge [Carmel et al., 2014] provides both linked entities and interpretations. Therefore, a large corpus for both entity linking and query interpretation is still missing. For that purpose we create an aggregation of the most frequently used entity linking corpora and manually annotate explicit and implicit entities as well as entity-based query interpretations for 2068 queries.

As existing entity linking frameworks like Nordlys [Hasibi et al., 2017a] or Smaph [Cornolti et al., 2016] focus on the traditional entity linking tasks, we also develop algorithmic solutions for our new problem settings of explicit and implicit entity recognition with a focus on efficiency and recall. Existing approaches to entity linking usually result in rather complex systems that take a lot of time identifying entities not fast enough for many practical applications with desired response times below some hundred milliseconds. The efficiency-effectiveness trade-off is an interesting aspect that we take into consideration for the development of our algorithms (i.e., response time vs. entity/interpretation quality).

This thesis is organized as follows. In Chapter 2 we review related work and the issues of existing problem statements and corpora. Chapter 3 explains the refined definitions of entity linking and query interpretation and the associated concepts. Based on these redefined problems we manually annotate a compar-

atively large corpus that we present in Chapter 4. We also provide algorithmic approaches to the refined problems described in Chapter 5. The conducted experiments and the evaluation results of our algorithms and existing entity linking systems are presented in Chapter 6. Chapter 7 briefly summarizes what we have learned through this work and the achieved results. In Chapter 8 we discuss which further steps and improvements can be considered based on this thesis.

Chapter 2

Related Work

In this chapter, we review the related work with a focus on the concept of named entities (Section 2.1), existing entity linking approaches and the respective problem definitions (Section 2.2), existing entity-based query interpretation methods (Section 2.3), and an overview of existing datasets (Section 2.4).

2.1 Named Entities

The concept of *named entities* was first introduced at the Message Understanding Conference and was defined as "identifying the names of all the people, organizations, and geographic locations in a text" [Grishman and Sundheim, 1996]. Further, the specification of named entities was extended to include numeric expressions like percentages, prices and time expressions (e.g., `31th May 1995`). The Stanford Natural Language Processing Toolkit [Manning et al., 2014] implements approaches to detect locations, persons and organizations and also offers to find numeric expressions or other miscellaneous entities. Our taxonomy is based on the extended named entity hierarchy [Sekine et al., 2002]. Although we use most of the classes from this taxonomy, a few subclasses are excluded not containing proper nouns. This means that the excluded subclasses contain instances referencing rather a group of entities than a unique one. We also do not consider numeric expressions.

Since Cornolti et al. define entities as instances of Wikipedia pages and their underlying concepts, Smaph [Cornolti et al., 2016] as well as TagMe [Ferragina and Scaiella, 2010] retrieve general concepts as part of entity linking. General concepts are defined as nouns that are no named entities (e.g., `city, peace`) and thus, for the purpose of our work, they are not covered by our refined definitions of entity linking.

2.2 Entity Linking in Queries

Since the ERD'14 Challenge took place in 2014 [Carmel et al., 2014], entity linking in queries gained remarkable attention. The goal of the ERD'14 Challenge was to utilize entity linking in order to generate possible interpretations of a query. The problem setting is to provide a set of valid entity linking interpretations using all available context for a given search query [Carmel et al., 2014]. Due to the lack of context, single queries may contain ambiguous entity mentions. The ERD'14 Challenge required its participants to generate different interpretations for each entity sharing the same mention (e.g., `new york` can be a mention of `New York City` as well as `New York (state)`). Also, part of their task was to resolve aliases. For example, "the governator" is an alias of the entity `Arnold Schwarzenegger` but also a mention of the eponymous TV series. Thus, the ERD organizers included different subtasks into the entity linking challenges which lead to rather imprecise interpretations.

Hasibi et al. distinguish two distinct definitions related to entity linking in queries, namely entity linking and semantic mapping [Hasibi et al., 2015]. Entity linking as defined by Hasibi et al. is identifying named entities in queries and link them to the corresponding entry in a knowledge base [Hasibi et al., 2017a]. Entity linking restricts annotations to be non-overlapping what means that each segment in a query can only be linked to a single entity. This limits ambiguous queries (i.e., queries with ambiguous entity mentions) to be interpreted having only one meaning. Semantic mapping, instead, includes all possible disambiguations to a mention and also handles aliases since semantic mapping is defined to return all entities which have a semantic relation to a query. These entities are not even required to be mentioned in the query as long they are related to the context. The goal of semantic mapping is to "support users in their search and browsing activities by returning entities that can help them to acquire contextual information or valuable navigational suggestions" [Hasibi et al., 2015].

An approach in the Nordlys system [Hasibi et al., 2017a] is entity retrieval which is similar to semantic mapping. Entity retrieval provides a ranked set of related entities to a given query rather than mention-entity pairs and therefore overlap is not restricted. Although interpreting queries is not the goal of entity retrieval solutions, Nordlys is potentially interesting for evaluating entity linking since ambiguous entities in a query are not prohibited.

One of the most common frameworks for annotating entities in short texts is TagMe [Ferragina and Scaiella, 2010]. TagMe focuses on on-the-fly annotations and therefore aims for efficiency while always keeping an eye on effectiveness. TagMe could yield interesting evaluation results especially in terms of performance vs. speed. However, TagMe also does not allow overlapping en-

tity mentions. Since Smaph [Cornolti et al., 2016] uses TagMe as a subroutine, the Smaph interpretation of the entity linking problem is similar. Overlap is not allowed and disambiguation is solved by selecting the most likely entity for ambiguous mentions.

To summarize, we can say that most of the literature define entity linking as non-overlapping which may result in too few candidate entities for query interpretation. Approaches allowing overlap typically are designed for a different purpose such as entity retrieval or semantic mapping and therefore may extract non-related entities with respect to interpretations of a query. Thus, we refine the traditional entity linking to enable high quality and more precise query interpretations.

2.3 Query Interpretation

In the ERD'14 Challenge [Carmel et al., 2014], interpretations of queries consisted of semantically compatible non-overlapping entities and segmentations of the remainder. Also Hasibi et al. treat the problem this way with potential multiple interpretations, who also agrees to the problem statement. We consider refined entity linking as a requirement for creating candidate entities for query interpretations.

2.4 Datasets

Since most definitions of entity linking are based on the task descriptions of the ERD'14 Challenge [Carmel et al., 2014], the corresponding dataset is one of the most frequently used datasets for evaluation. Back then, the short text task was hosted as an evaluation service equipped with a training and test set with 500 queries each. Unfortunately, these datasets were kept private. Only 91 queries and the associated Freebase knowledge base containing about 2.35 million entities [Google, 2013] were made publicly available for training purposes. These 91 queries were sampled from a query log of a commercial search engine but are rather too few for a meaningful evaluation. Additionally, only 50% of the 91 queries contain entities which is considerably lower than the 70% of commercial queries containing entities reported by Guo et al. [Guo et al., 2009].

Yahoo published a dataset for entity linking, the "Yahoo Search Query Log to Entities" [Yahoo], with a comparatively large set of 2635 queries. However, the Yahoo dataset does not contain query interpretations. Instead, annotated entities were marked if they are part of a query's intent but not for which of the multiple potential query interpretations.

In 2013, a dataset was introduced for "Entity Search in DBpedia" (ES-DBpedia) [Balog and Neumayer, 2013] with 485 queries. The queries were gathered from common benchmarking evaluation campaigns in the domain of search and entities such as the TREC 2009 Entity Track [Balog et al., 2009] or the INEX 2009 Entity Ranking Track [Demartini et al., 2010]. All queries of the dataset were annotated with entities assigned with a binary relevance judgment gathered via crowdsourcing. As a result, some of the queries were annotated with unrelated entities. This dataset does not include interpretations either. The entities of this corpus are entries from the DBpedia v3.7 [DBpedia, 2011] which is an outdated version of DBpedia.

Because of that, Hasibi et al. introduced a revised version of the ES-DBpedia collection following the same guidelines for relevance assessments and even acquired the same group of crowdsourcing workers. The dataset "Entity Search in DBpedia v2" [Hasibi et al., 2017b] is based on the DBpedia dump DBpedia 2015-10 [DBpedia, 2016], which is a more recent version. Unfortunately, the first version nor the second of the ES-DBpedia corpus contain interpretations. Moreover, the relevance judgments for the entities often do not take the entity's relevance to the query into account. Having irrelevant entities is not expedient for query interpretation. Since crowdsourcing was used to collect the relevance judgments, some entities also got some questionable relevance (e.g., the entity `Amsterdam Guitar Trio` was annotated as relevant to the query `guitar classical bach`).

Since the existing corpora do not really fit our problems and come with some issues, we aim to create a consistent corpus with a sufficient number of queries providing explicit and implicit entities as well as query interpretations.

Chapter 3

Problem Statement

In this chapter, we define our version of the problems related to detecting entities in queries (Section 3.1 - Section 3.4) and introduce our view on entity-based interpretation (Section 3.5).

3.1 Entities in Queries

A query q is sequence $t_1, t_2, ..., t_n$ of terms. A segment s of a query is a consecutive subsequence $t_i, ..., t_j$ and the set of all possible segments of q is denoted by S. For example, the query `new york` has the segments $S = \{(\texttt{new}), (\texttt{new york}), (\texttt{york})\}$. A segment s will be called mention if it represents an entity e.

An entity e is an instance of a class of a named entity taxonomy. We employ the extended named entity hierarchy from Sekine et al. [Sekine et al., 2002] but exclude a few very general classes for natural objects (e.g., `oxygen`), diseases (e.g., `heart failure`), colors (e.g., `red`), time and numeric expressions (e.g., `3 o'clock`, `100 yen`). From the remaining 8 main classes shown in Figure 3.1, we exclude the following subclasses: postal addresses, phone numbers, emails or URLs, facility parts like room numbers, low specificity classes for materials (e.g., `adobe`), clothing types (e.g., `kimono`, `clog`), weapon types (e.g., `shotgun`), crimes (e.g., `murder`) and identification numbers (IDs, class specifications or service numbers), religions (e.g., `christianity`), academic fields (e.g., `computer science`), sports (e.g., `baseball`) and arts (e.g., `impressionism`) as well as general titles of persons (e.g., `Mr.`, `Dr.`) and units (e.g., `kg`). The resulting reduced taxonomy includes 108 classes from the originally 150 classes of named entities by Sekine et al. [Sekine et al., 2002]. Please note the full taxonomy in Appendix A.

Entities are not restricted to a knowledge base. If an instance of a class of our taxonomy is mentioned in a query and there is no associated resource in

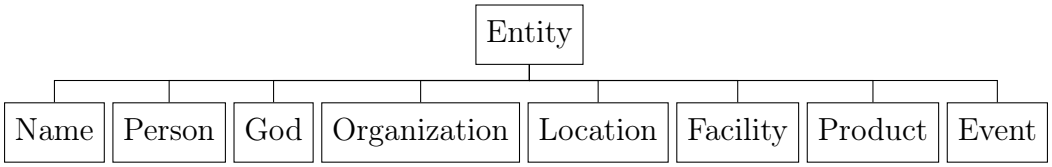

Figure 3.1: Main entity classes.

the specified knowledge base it still will be treated as entity.

3.2 Entity Recognition

Given: A query q

Task: Entity recognition is the task of identifying all entities in q. The entities can either be mentioned explicitly (Section 3.3) or implicitly (Section 3.4). In either case, an identified entity needs to be mapped to a mentioning segment s including a numeric relevance score r. Thus, solving entity recognition results in a set of unique triples $ER = \{(s_1, e_1, r_1), (s_2, e_2, r_2), ...\}$. This set may be empty if q does not contain any entity.

The relevance of an entity depends on the context of a query. A mentioned entity has a relevance of zero if it is semantically incompatible with the context of a query. For example, the city of `Fence, Wisconsin` should get a 0-relevance in the query `how to build a fence` since it is not likely for a user to request a tutorial for "building" the city.

3.3 Explicit Entity Recognition

Given: A query q

Task: The problem of explicit entity recognition is to identify all entities in a query that are explicitly mentioned. A query contains an explicit mention if the underlying segment is an entity's name or another associated surface form of it.

A surface form is a representative of an entity used in text. For example, `new york` is a surface form of `New York (state)` and of `New York City` as well. As shown in the example, many entities share surface forms which can yield ambiguity for certain query segments. This may also leads to ambiguous interpretations of a query since explicit entity recognition will be one of the

two base problems for finding entity-based query interpretations. Note that we allow overlapping entity mentions in the base problem to not miss any candidate for interpretation.

3.4 Implicit Entity Recognition

Given: A query q

Task: The identification of all entities in a query that are implicitly referenced is called implicit entity recognition. An implicit entity is rather mentioned by a description than by a name or surface form.

For example, `president of the usa` today is an implicit mention of `Donald Trump` and potentially also the previous 44 other presidents. Identifying implicit entities is quite important for query interpretation since entities that are intended to be mentioned are not covered by explicit entity recognition. Note that implicit entities and explicit entities are disjoint such that the combination of the respective detection problems forms the entity recognition problem. We are the first to really clarify this distinction in order to not mix subtasks like traditional entity linking.

3.5 Entity-based Query Interpretation

In order to find interpretations of a query, it is mandatory to detect the explicitly and implicitly mentioned entities. For some queries a specific entity can be both, explicit and implicit. This happens if an entity is addressed by an implicit mention and the query contains one of the surface forms of the same entity as well. A short example is the query `dulles airport location`. The area of `Dulles, Virginia` is mentioned explicitly as `dulles` and also implicitly requesting the location of the `Washington Dulles International Airport` which includes `Dulles, Virginia`.

Given: A query q and the in q identified entities

Task: Entity-based query interpretation is the task of segmenting a query and replace mentioning segments with the associated explicit or implicit entities.

An entity-based query interpretation is a semantically compatible segmentation in which mentions are replaced by their related explicit and implicit entities. Therefore, distinguishing conceptual segments from entity mentions is

an important aspect for query interpretation. Each interpretation comes with a numeric relevance score which depends on the relevance of the entities the interpretation consists of.

Although overlapping entities are not prohibited in entity recognition, they are not allowed in query interpretation. That means, that explicit entities sharing the same mention or overlap with each other cannot be part of the same interpretation. Implicit entities are treated more freely. The containment of explicit entity mentions in implicit ones and vice versa is allowed. For example, a potential interpretation of the query `obama mother` includes the explicit entity `Barack Obama` and the implicit entity `Ann Dunham`. Here, the explicit entity mention `obama` is contained in the implicit entity mention `obama mother`.

Chapter 4

Query Interpretation Corpus

In this chapter we explain the process of creating our new query interpretation corpus. We describe the query selection (Section 4.1) and how these queries were manually assessed with a difficulty index (cf. Section 4.2) and classified in certain classes (cf. Section 4.3) and the annotated entities have relevance scores (cf. Section 4.4). The specification of the underlying knowledge base can be found in Section 4.5. The actual annotation process is detailed in Section 4.6. In Section 4.7, we give an analysis of the resulting corpus.

4.1 Query Aggregation

The queries for our corpus were gathered from the existing corpora, namely the ERD'14 dataset [Carmel et al., 2014], the "Yahoo Search Query Log to Entities" dataset [Yahoo] and the "Entity Search in DBpedia v2" dataset [Hasibi et al., 2017b]. In total, 3193 queries from these datasets were included. We normalized the queries by lowercasing, spell correction and we removed unnecessary special characters and whitespaces. Since several of the queries appeared twice or more after normalization we removed duplicates which result in 2697 remaining queries for the second annotation phase.

4.2 Difficulty Assignment

For the annotation of a query's difficulty we created 5 levels of difficulty represented by numerical values. The difficulty is determined based on the complexity of the mentioned entities and the query's grade of ambiguity. Table 4.1 shows the difficulty criteria and examples for the defined difficulty classes. The index of difficulty 4 was assigned if a query's intent was unclear. A difficulty of 20 represents that the number of explicit or implicit entities exceeds our

Index	Criteria	Example
1 (easy)	low ambiguity few/no explicit entities no implicit entities	`connie britton`
2 (moderate)	moderate ambiguity various explicit entities easy implicit entities	`member of u2`
3 (difficult)	high ambiguity complex implicit entities	`university of north dakota`
4 (unclear)	unclear intent	`best house plans`
20 (overflow)	entities exceed limit of 20	`free online games`

Table 4.1: Criteria for the difficulty index of a query.

threshold of 20 entities. Both, 4 and 20 are disqualifier for a query for the second annotation phase. The removal of queries concerning difficulty resulted in 2068 remaining queries with a difficulty of 1.33 on average. In Table 4.2 you can see the distribution of difficulty across all queries.

Difficulty	#Queries
1	1425
2	613
3	30
4	74
20	555

Table 4.2: Distribution of difficulty

4.3 Query Classification

Our annotations also introduced classes based on a query's intent and common behaviour which can be common patterns of explicit and implicit query mentions.

- **Categorical Query** (CatQ)
  Queries referring to a group of implicit entities are called categorical queries. Entity mentions in plural are a good indicator for a categor-

ical query. The implicit entities a categorical query references are often related to a contained explicit entity. The query `members of u2` is an example of such a categorical query since the implicit entities `Bono`, `The Edge`, `Adam Clayton` and `Larry Mullen Jr.` are related to the explicit entity `U2` and the query contains the plural entity mentions. Usually, one of the interpretations of a categorical query contains all the addressed implicit entities since the intent of a categorical query may be to receive an overview of members of the "category".

- **Conceptual Query** (ConQ)
  Basically, there are two kinds of queries that aim to find information about some concept. A concept is a noun phrase that is not mentioning an explicit or implicit entity (e.g., `kiwi fruit`). Queries that do not contain any entities often request information about general concepts (e.g., `black powder ammunition`). But it can also happen that concepts related to an explicit entity in the query are requested. For example, the query `churchill downs horse racing schedule` aims for a schedule of horse races taking place at `Churchill Downs` (racetrack). The schedule addressed here is a "concept" related to the explicit entity `Churchill Downs` such that the query is tagged as conceptual.

- **Literal Entity Query** (LQ)
  In order to be classified as literal, a query is required to be an entity's name or surface form itself (e.g., `barack obama`). A mention of an annotated explicit entity for a literal entity query includes the whole span of the query.

- **Resource Locator Query** (RLQ)
  Some of the queries in the corpus are plain URLs and are classified as instances of resource locator queries. Since URLs are not a part of our entity taxonomy, these queries do not contain any entities.

- **Relational Query** (RQ)
  Relational queries are similar to the conceptual queries. The difference is, that the requested information regarding a contained explicit entity is an implicit entity itself (e.g., `niagara falls origin lake` where `Niagara Falls` is the explicit and `Lake Erie` is the respective implicit entity for `origin lake`). All relational queries share the same pattern: a mention of an explicit entity followed by a mention of a related implicit one.

- **Question Query** (QQ)
  Question queries are queries formulated as questions (e.g.,

`how do sunspots effect us`). Even if the answer to such question is an entity, they are not considered as implicit entities because question answering is no subtask of implicit entity recognition.

The classes are not necessarily distinct. There are common combinations, like ConQ and LQ, as well as combinations impossible by definition (e.g., LQ cannot also be RLQ). Table 4.3 shows the distribution of queries belonging to certain classes or combinations of classes.

Categories	#Queries
Categorical Queries	21
Conceptual Queries	1287
Literal Entity Queries	468
Resource Locator Queries	39
Relational Queries	15
Question Queries	55
ConQ, LQ	174
ConQ, RQ	3
CatQ, ConQ	5
CatQ, LQ	1

Table 4.3: Distribution of categories in our corpus.

4.4 Entity Relevance

The relevance of an entity is a numerical assessment how likely an entity is part of a query's intent. For our corpus we annotate three levels of relevance an entity can have: relevant (2), plausible (1) and irrelevant (0). An entity is relevant to a query if it is most likely a part of the intent of the query. A plausible entity also has full semantic compatibility to the query context but another entity or concept seems to be more related. Irrelevant entities are semantically incompatible or not mentioned entities and are not included in our corpus. If an entity does not appear in our corpus as explicit or implicit entity, it is automatically classified as irrelevant. To give an example: the query `aol` contains the relevant entity `AOL`, the plausible entity `Alert on LAN` (since its the corresponding meaning of the abbreviation) and, among others, `Microsoft` is an irrelevant entity to that query.

4.5 Knowledge Base

In general, we view the web as the source of entities. In practice, there would be even more possible entities (e.g., every person's name is an entity by definition) but the web as a knowledge base is a good starting point to extend traditional approaches that only consider Wikipedia or Freebase as knowledge base. In our work, we prioritize different domains for different classes of entities. Basically, we prefer Wikipedia if an associated resource is available. If a local venue or organization is mentioned, we typically use Yelp to specify this entity. For persons who do not appear as an instance of an Wikipedia article, we choose the associated resource from Linkedin. Some small number of entities are either specified by their own websites or, if no resource is available at all, a clarifying description is chosen as the identifier for such an entity.

4.6 Annotation Procedure

In this section, we present a detailed overview of the annotation process consisting of annotation of entities (cf. Section 4.6.1) and interpretations (cf. Section 4.6.2).

4.6.1 Annotation of Entities

The annotation phase was one of the most time consuming operations of this academic work. Due to the quality issues of the existing datasets discussed in Section 2.4, we decided to start from scratch with a manual annotation procedure. In order to improve the consistency, all annotations come from a single expert annotator, who discussed difficult cases with a second annotator. On average about 50 queries were annotated within one hour. The semantic compatibility of entities fitting the context of a query requires at least a basic understanding of what an entity is about, which results in a workload of about 40 to 50 hours over 3 months to complete the annotations of entities (not including the annotation procedure of query interpretation).

During the process of annotating entities, we followed a guideline adapted from the annotation guidelines from Hasibi et al. [Hasibi et al., 2015]:

($E1$) An entity has to be an instance of a class of the named entity taxonomy presented in Section 3.1. This implies that general concepts are not annotated.

($E2$) Overlapping entities are allowed in the sets of explicit and implicit entities, since it enables ambiguity of queries. The query `promoting`

`diversity` mentions `Diversity (album)` as well as `Diversity (dance troupe)` in the segment `diversity`.

($E3$) Take the longest possible span for a mention of an entity. The whole span of the query `asheville north carolina` mentions `Asheville, North Carolina` but not the segment `asheville` alone. Note that `North Carolina` should still be annotated as well due to ($E2$).

($E4$) An entity has to be semantically and grammatically compatible with the context of a query in order to be annotated. The entity `India (Vega album)` is semantically not compatible to the query `living in india` but the country `India` is. So, the album will not be annotated as an (explicit) entity here.

Following these guidelines, we obtained 1597 queries with a difficulty between 1 and 3 (which is about 77% of all queries) having at least one annotated explicit or implicit entity. The fact that our guidelines are less restrictive and the additional implicit entities included in our corpus result in the somewhat higher percentage of entity-containing queries compared to the 70% found earlier [Guo et al., 2009].

4.6.2 Annotation of Interpretations

In a second annotation round, the interpretations were formed using the annotated entities from the first phase. Since we focused on query interpretations being entity-based, we decided to provide interpretations for the 1597 queries that contain at least one entity. An interpretation of such a query is represented by a segmentation wherein all mentioning segments are mapped to the associated entities. For building the interpretations we stated a set of guidelines we followed during the annotation phase.

($I1$) Each individual interpretation contains only non-overlapping explicit entities. However, overlapping entities are allowed in different interpretations of the same query. If two overlapping explicit entities both form a relevant interpretation then separate interpretations are generated for each. For example, the query `my yahoo mail` has the two interpretations (`My Yahoo`, `mail`) and (`my`, `Yahoo Mail`) but not (`My Yahoo`, `Yahoo Mail`).

($I2$) The boundaries of a mention of an implicit entity are treated more freely in interpretations than explicit entity mentions: the containment of explicit entities in implicit ones is allowed and vice versa. For example, the query `obama mother` contains `Barack Obama` as explicit and

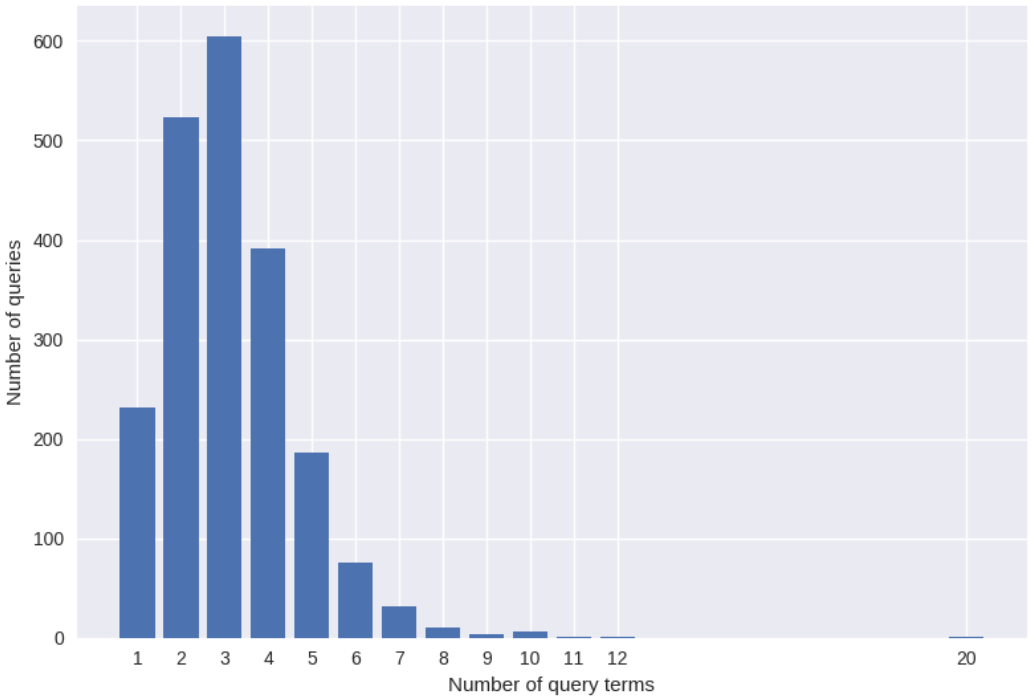

Figure 4.1: Distribution of query lengths of our query interpretation corpus.

`Ann Dunham` as implicit entity. The mention of the entity `Ann Dunham` is `obama mother`, which contains the mention of `Barack Obama` (`obama`). Still (`Barack Obama`, `Ann Dunham`) is a valid interpretation.

($I3$) The entities of an interpretation have to be semantically compatible. The set of entities a query interpretation is based on may contain entities that are compatible in the context of the query or a specific entity but may not be meaningful in a different setting of entities. For example, the query `nyc tourism` may (rather far-fetched) be interpreted as where to buy `Tourism (Roxette album)` in `New York City` but not as (`NYC (band)`, `Tourism (Roxette album)`) since there is no semantic connection between these two entities.

4.7 Corpus Analysis

The average query in our corpus has 3.14 terms (cf. Figure 4.1 for a histogram of the length distribution). Most of the queries have between 1-5 terms but one query also has 20 terms. This longest query is

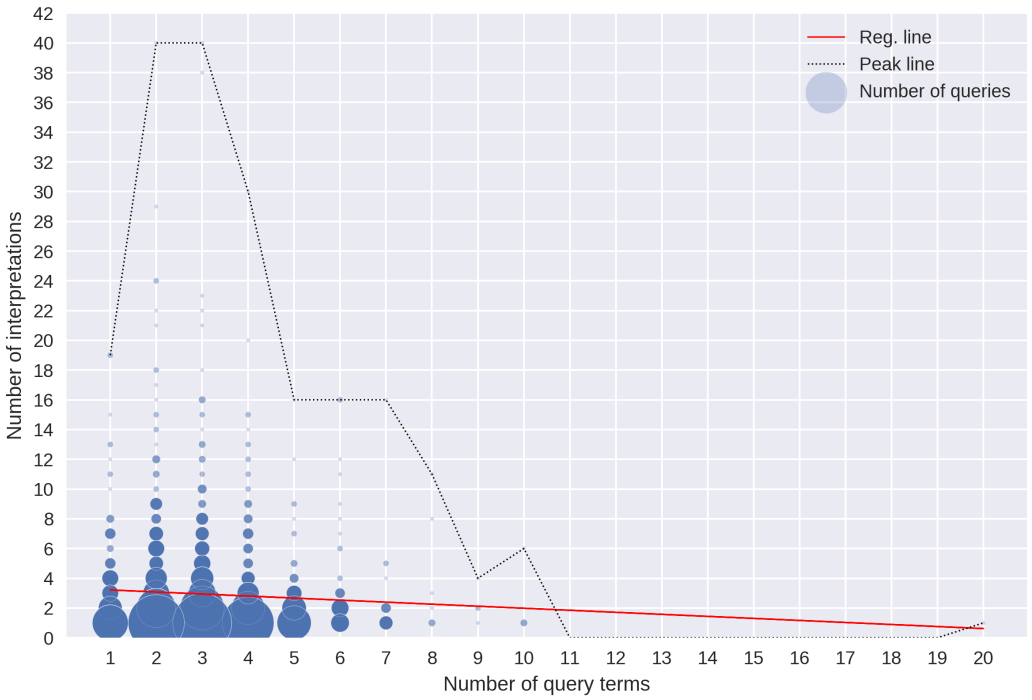

Figure 4.2: Correlation of query length and number of interpretations in our query interpretation corpus.

```
what are the effects of the ongoing eurozone crisis on
national economic indicators such as unemployment gdp
public debt etc
```

Since interpretations also depend on the context, long queries basically have fewer potential semantically compatible interpretations. Nevertheless, rather short queries do not necessarily result in many interpretations since the number of mentioned entities has a big influence on the number of meaningful interpretations. Figure 4.2 shows the number of interpretations as a function of the number of query terms. The area of the scattered query markers is mapped to the number of queries fulfilling the same condition. According to that diagram, we can see that on average the number of interpretations is somehow distributed equally. At first, it appeared that there is no relation between the length of a query and the number of interpretations since the regression line also shows only a slight trend that short queries have more interpretations than long ones. However, the visualization of the peak number of interpretations shows that short queries allow a larger quantity of potential interpretations than long queries.

Table 4.4 presents the five most ambiguous queries (i.e., queries with the

most interpretations) of the corpus with difficulty between 1 and 3. Their high numbers of interpretations is based on combinations of ambiguous, semantically compatible mentions of entities. For example, `hoboken nightlife` contains four explicit entities for `hoboken` and nine for `nightlife`. Since all of these and the conceptual segment `hoboken` are semantically compatible, the query can be interpreted in 40 different ways ($4 \cdot (9+1)$). `hoboken` can be a mention for four different locations. In fact, an ambiguous mention of a location in a query is a reoccurring pattern that often leads to many interpretations. The segment `jersey` from the query in Table 4.4 mentions six different locations, `port arthur` mentions six, `costa rica` mentions three and `goodwin` may be a mention for four different locations.

Query	Interpretations
`hoboken nightlife`	40
`the current jersey`	40
`port arthur massacre`	38
`four seasons costa rica`	30
`goodwin games`	29

Table 4.4: Ranking of the most ambiguous queries in the query interpretation corpus.

The corpus contains 1578 queries with explicit and 131 with implicit entities. An average query contains 2.14 explicit and 0.18 implicit entities. The average for explicit entities is rather high since 471 queries do not contain entities at all. One reason for the rather high number of explicit entities is the ambiguity of mentions of explicit entities (i.e., number of explicit entities sharing the same mention). Table 4.5 shows the five most ambiguous mentions in our corpus and the number of associated entities.

Mention	Entities
`arsenal`	19
`apple`	18
`breakdown`	18
`carnival`	17
`ben franklin`	15

Table 4.5: Ranking of the most ambiguous mentions

The term `arsenal` is the most ambiguous mention in the corpus and also has a great variety of associated entities. It refers to three locations, a comic series,

two films, two bands, eight sport clubs, a former car, a Bulgarian company and a former series of French aircraft.

Chapter 5

Algorithmic Approach

This chapter introduces our approaches to solving entity recognition (Section 5.1). Our approaches are based on retrieving candidates from large text corpora. An overview of the used tools and libraries is given in Section 5.1.1. The necessary indexing procedure for implicit and explicit entity recognition is described in Section 5.1.2. Finally, Section 5.1.3 and Section 5.1.4 explain the concepts of our algorithmic solutions for recognition of explicit and implicit entities, respectively.

5.1 Entity Recognition

Algorithms for entity recognition basically consist of three steps: (i) candidate generation, (ii) scoring, and (iii) selection. We mainly focus on the candidate generation and scoring because these are the important tuning knobs to receive a high recall. Query interpretation is based on entity recognition as a candidate generation phase and so it is important to not miss any candidate(s) in the first phase.

5.1.1 Implementation Details

Our approaches to entity recognition (explicit and implicit) are implemented with the Java SE 8 SDK [Oracle, 2014]. For some specific problems, we use a series of third-party libraries. Parsing and processing CSV files is done via Apache Commons CSV [Apache, 2014], for additional string processing (e.g., capitalization) Apache Commons Lang [Apache, 2011a] is used. Some statistical computations are done with Apache Commons Math [Apache, 2011b] and Apache Lucene [Apache, 2017] is used to tokenize queries into terms and segments. Since we need fast and efficient access to huge amounts of data, we use Multimap IO [Martin Trenkmann] for 1:n

and RocksDB [Facebook Open Source, 2016] for 1:1 persistent key-value storage of documents. The serialization of such documents is handled with FST [Moeller, 2015]. HTML is parsed with jsoup [Jonathan Hedley, 2017] and JSON with Jackson Core [FasterXML, 2016]. Since the provided dumps of Wikipedia are wrapped in markup we parse these dumps with WikiXMLJ [Delip Rao].

5.1.2 Indexing

In order to automatically decide whether a segment represents an entity, we developed a digital representation of our entity taxonomy with the help of DBpedia Ontology [DBpedia, 2017], from which we manually picked the "best" matching classes. The DBpedia class *Agent* covers *Person*, *God* and *Organization* of our taxonomy. DBpedia's class *Place* represents our *Location* class. *Work* in combination with *Game* forms our *Product* class. *Event* from DBpedia is used to cover our same-titled class. *Agent* and *Mean of Transportation* together form our *Facility* class. All the titles of documents belonging to those listed DBpedia classes form the dictionary for the entity detection. This dictionary contains around 2.8 million Wikipedia titles we considered as entities.

For an efficient search on Wikipedia, we created an inverted index where an entity is mapped to a list of Wikipedia articles containing it. We use the Wikimedia dump of February 20/2018. For each document of this dump, we check if it is one of the 2.8 million entity articles. If so, we check for each segment of the content of these articles, if it represents some entity and counted the occurrences. These detected entities are indexed and the articles referring to the entities are sorted by their respective entity frequency. Since most of the mentioned entities in a text are rather conjugated or represented as surface forms, we also include redirects that refer to these surface forms. Wikipedia provides redirects to improve a user's search by eliminating misspellings and surface forms to lead a user to the desired resource. More than 3.2 million redirects leading to an entity page were collected from the Wikimedia dump and included in the index.

5.1.3 Explicit Entity Recognition

Algorithm 1 gives the pseudocode of our approach to explicit entity recognition. Further, we explain the typical three steps.

(i) Candidate Generation

For each segment of a given query, we derive a list of search terms. These search terms are obtained by resolving redirects and disambiguations. Disam-

biguation pages in Wikipedia connect ambiguous terms to all articles having the same name or surface form as their title. Thus, resolving disambiguations and redirects of a segment can lead to a promising list of representations of this segment. However, if the segment does not redirect to an entity page and does not share a name or surface form with some entity, the list of search terms only contains the segment itself. For each search term the index returns a list of candidate entities mentioned by the current segment. We only consider the top 100 results by frequency of occurrence because usually this list is rather long and applying the following scoring procedure to all of these candidates would slow down the system.

(ii) Scoring

For computing the relevance score of an entity we choose the term-based *Jaccard Index* since Cornolti et al. presented this method as a promising measure for relatedness [Cornolti et al., 2016]. The Jaccard Index of two sets $T1$ and $T2$ of terms is defined as follows:

$$Jaccard(T_1, T_2) = \frac{|T_1 \cap T_2|}{|T_1 \cup T_2|} \tag{5.1}$$

The Jaccard Index yields values in the range $[0, 1]$. For the computation of relevance of an entity, we are interested in the Jaccard Index of a segment and the terms of an entity's name. Additionally, redirects are 1:1 mappings between surface forms and entities and therefore always receives a score of 1. To prefer longer spans (cf. $(E3)$ of Section 4.6.1), we factor in a normalization term.

$$norm = \frac{|s|}{|q|} \tag{5.2}$$

By multiplying the Jaccard-based relevance with $norm$, entities with a greater span in a query can benefit.

(iii) Selection

We suggest two different approaches for the selection process. The precision-oriented approach is to select every result with a relevance above a fixed threshold. The recall-oriented way of selection is to choose the n best entities by score. Since it is important to supply as many entities as possible in the candidate phase of query interpretation, the recall-oriented approach supports that scenario better.

Algorithm 1 Explicit Entity Recognition

```
1: procedure RECOGNIZE-EXPLICIT(query)
2:     results ← ∅
3:     for all segment ∈ query do
4:         redirect ← redirect of segment
5:         disambiguations ← disambiguations titled as segment as a list
6:
7:         documents ← documents containing segment,
                           redirect or disambiguations
8:         candidates ← select k documents
                           with highest occurrence frequency
9:
10:        for all entity ∈ candidates do
11:            score_segment ← JACCARD(entity, segment)
12:            score_redirect ← JACCARD(entity, redirect)
13:            if score_redirect < 1 then
14:                score ← score_segment
15:            else
16:                score ← 1
17:            end if
18:            norm ← |segment|/|query|
19:            score ← score · norm
20:
21:            if score > 0 then
22:                annotation ← (segment, entity, score)
23:                results ← results ∪ annotation
24:            end if
25:        end for
26:    end for
27:
28:    return n best results by score
29: end procedure
```

5.1.4 Implicit Entity Recognition

Algorithm 2 gives the pseudocode of our approach to implicit entity recognition. Further, we explain the typical three steps.

(i) Candidate Generation

In our corpus implicit entities often relate to an explicit entity. Therefore, we generate candidates in a similar to the explicit entity approach way. The important difference is that those candidates that were requested for a certain segment are used as candidates for all other not overlapping segments of the query. Hereby, we assume that a segment is a potential mention for explicit entities and all other non-overlapping segments are potential specifiers for an implicit entity.

(ii) Scoring

Let *segment* be a potential mention of an explicit entity and *segment*$'$ a non-overlapping segment to *segment* in the same query. The relevance score of an entity is computed by the harmonic mean between the frequencies of *segment* and *segment*$'$ in the content of an entity's underlying document. Both frequencies have to be greater than zero to result in a score greater than 0.

(iii) Selection

The selection process of implicit entities is completely identical with the selection of explicit entities. The recall-oriented approach which takes the top n results by score is chosen for implicit entities as well.

Algorithm 2 Implicit Entity Recognition

```
1: procedure RECOGNIZE-IMPLICIT(query)
2:     results ← ∅
3:     for all segment ∈ query do
4:         segments′ ← segments in query with no overlap with segment
5:
6:         redirect ← redirect of segment
7:         documents ← documents containing segment or redirect
8:         candidates ← extract k best documents by frequency
9:
10:        for all document ∈ candidates do
11:            freq_seg ← frequency of segment in content of document
12:            for all segment′ ∈ segments′ do
13:                freq_seg′ ← frequency of segment′ in content of document
14:
15:                if freq_seg′ > 0 then
16:                    score ← HARMONIC_MEAN(freq_seg, freq_seg′)
17:                    annotation ← (segment′, document, score)
18:                    results ← results ∪ annotation
19:                end if
20:            end for
21:        end for
22:    end for
23:    return n best results by score
24: end procedure
```

Chapter 6

Experiments

In this chapter we report about the experiments conducted to evaluate our algorithmic approaches. The used evaluation metrics are presented in Section 6.1. The general experimental setup is described in Section 6.2. The concept of the baseline method used in the evaluation is given in Section 6.3. Finally, the evaluation results achieved by our and other systems for explicit and implicit entity recognition are presented and discussed in Section 6.4.

6.1 Evaluation Metrics

To assess the quality of a set E of found entities in a query compared to a set E' of desired entities from the ground truth, the precision P of E is defined as

$$P = \begin{cases} \frac{|E \cap E'|}{|E|}, & \text{if } |E| > 0 \\ 1, & \text{if } |E| = 0, |E'| = 0 \\ 0, & \text{if } |E| = 0, |E'| > 0 \end{cases} \tag{6.1}$$

Note that cases where a query contains no entities or the system does not provide entities are treated differently. Recall R of E is defined analogously as

$$R = \begin{cases} \frac{|E \cap E'|}{|E'|}, & \text{if } |E'| > 0 \\ 1, & \text{if } |E| = 0, |E'| = 0 \\ 0, & \text{if } |E| > 0, |E'| = 0 \end{cases} \tag{6.2}$$

As usual, the F_1 is the harmonic mean of P and R

$$F_1 = \frac{2 \cdot P \cdot R}{P + R} \tag{6.3}$$

Since every entity is assessed with a relevance score, we further can weight the recall and the F_1 using the weight w depending on the relevance of the entities in the ground truth.

$$w = \frac{\sum_{e \in E \cap E'} rel(e)}{\sum_{e' \in E'} rel(e')} \tag{6.4}$$

Where $rel()$ represents assigned relevance to an entity in the ground truth weighted recall R^* then is defined as

$$R^* = w \cdot R \tag{6.5}$$

The weighted $F1^*$ is computed accordingly as

$$F_1^* = \frac{2 \cdot P \cdot R^*}{P + R^*} \tag{6.6}$$

Since we want to observe the efficiency-effectiveness trade-off, we measured the response time RT. All metrics are macro-averaged over all queries.

6.2 Setup

We evaluate our algorithmic approaches for explicit and implicit entity recognition against TagMe [Ferragina and Scaiella, 2010], Nordlys [Hasibi et al., 2017a] and Smaph [Cornolti et al., 2016]. TagMe is used via accessing the provided API. The results of TagMe were gathered in June 2018 with a default parametrization in the English language.

Nordlys as deployed via its Python library locally. We are interested in two different functionalities of Nordlys, namely *entity linking in queries* and *entity retrieval*. The applied method for entity linking from Nordlys is LTR-greedy, since it is the recommended method in terms of efficiency and effectiveness [Hasibi et al., 2017a]. The default parametrization is used for entity retrieval except that the value of the variable *num_docs* of *first_pass* is decreased from 1000 (default) to 100 because our parameter for the number of scored documents is also set to 100.

Smaph was deployed as a local server on our system. *Smaph-3 greedy* was used for evaluation with its default parametrization.

The corpus used for the evaluation is our query interpretation corpus presented in Chapter 4. All of the evaluated systems collected their entities from

Wikipedia or DBpedia and therefore, for this experiments, we ignored entities from other domains.

Regarding the response times of our and other systems, for comprehensible results we would like to mention the system's specifications used for the conducted experiments. Important here is the CPU which is an Intel(R) Core(TM) i5-6400 @ 2.7 GHz. The available RAM was 32GB whereof 8GB were assigned to the Java Virtual Machine. Since several API calls are made for TagMe and Smaph the Internet speed is measured resulting in a download rate of about 900 Mb/s and an upload speed of 750 Mb/s.

6.3 Baseline

A baseline method was developed that assumes that none of the queries of the corpus contains entities. The baseline method never returns any annotation to a given query. With this approach we find out a potential benefit from not returning entities when none are present in the ground truth.

6.4 Evaluation

6.4.1 Explicit Entity Recognition

Nordlys' entity linking method performes best identifying explicit entities regarding precision, recall and F_1 score. Table 6.1 shows the evaluation results on all 2068 queries of our corpus.

Algorithm	P	R	F_1	R^*	F_1^*
Nordlys Entity Linking	.6911	.5521	.5802	.5013	.5218
Explicit Entity Recognizer	.1568	.4041	.1776	.3505	.1596
Smaph	.4510	.3756	.3704	.3223	.3140
TagMe	.3885	.3657	.3324	.3108	.2780
Nordlys Entity Retrieval	.0453	.3282	.0683	.2885	.0637
Baseline	.2606	.2606	.2606	.2606	.2606

Table 6.1: Evaluation results of our and other entity linking frameworks for identifying explicit entities in queries from our query interpretation corpus.

One reason Nordlys' entity linking method achieved rather good results may be due to the strict evaluation metrics for precision and recall for queries that do not contain entities. As shown by the baseline method, systems can benefit from not returning any entities if a query of the ground truth does

not contain any either. To be precise, 0.2606 of recall and precision could be gained for not return entities if none are desired. Since our algorithmic approach to explicit entity recognition aims to maximize the recall, our system tends to return entities even if a query does not contain any. Thus, our system achieved rather bad precision and F_1 scores compared to Nordlys. Nordlys' entity retrieval method returns a fixed number of entities independent of being mentioned in the query. Therefore, entity retrieval achieved the worst results.

For analysis of the influence of the strict evaluation metrics on a system's performance we also evaluate on the subset of queries which contain at least one explicit entity. Table 6.2 presents the evaluation results on queries containing entities. The strict behaviour of the evaluation metrics has a huge impact on the evaluation results. Regarding the recall, TagMe achieved the best result although the results are rather close. Our system reached similar numbers for recall as Smaph but with comparatively bad precision. However, the response times of Smaph are remarkably higher than the average response time of our system. Smaph needs about 2 minutes per query to identify mentioned entities. Response times differ in a huge range. TagMe achieved the best response times of 40ms and Smaph's response times is the longest with about 117s. In fact, only TagMe and our system are able to identify entities in under a second. Our algorithmic approach is capable to identify explicit entities in 270ms. The exact response times are presented in Table 6.3. To conclude, the best efficiency-effectiveness trade-off achieved TagMe and our approaches.

Algorithm	P	R	F_1	R^*	F_1^*
TagMe	.5202	.4894	.4443	.4151	.3708
Smaph	.5780	.4759	.4689	.4038	.3927
Explicit Entity Recognizer	.1394	.4739	.1673	.4016	.1433
Nordlys Entity Linking	.6391	.4511	.4891	.3824	.4102
Nordlys Entity Retrieval	.0443	.4269	.0753	.3731	.0691

Table 6.2: Evaluation results of our and other entity linking frameworks for identifying explicit entities in the queries containing entities from our query interpretation corpus.

These results prove that explicit entity recognition is a considerably harder problem than traditional entity linking. Semantic compatibility is hard to assess and because of high ambiguity it is also difficult to receive all desired overlapping entities.

Algorithm	RT [ms]
TagMe	40
Implicit Entity Recognizer	226
Explicit Entity Recognizer	270
Nordlys Entity Retrieval	1884
Nordlys Entity Linking	4448
Smaph	117076

Table 6.3: Average response times per query of the evaluated entity linking frameworks identifying explicit and implicit entities.

6.4.2 Implicit Entity Recognition

The results of identifying implicit entities in all queries of our query interpretation corpus can be found in Table 6.4. Due to the rare occurrence of implicit entities in queries (about 6%), the baseline method was very effective. Also, systems which tend to rather not return entities, like Nordlys' entity linking system, benefit a lot from this and the strict evaluation metrics. Thus, we also evaluate on the subset of queries containing at least one implicit entity.

Algorithm	P	R	F_1	R^*	F_1^*
Baseline	.9468	.9468	.9468	.9468	.9468
Nordlys Entity Linking	.3776	.3771	.3771	.3769	.3768
Implicit Entity Recognizer	.2163	.2213	.2168	.2199	.2166
Smaph	.0538	.0531	.0532	.0528	.0529
Nordlys Entity Retrieval	.0180	.0337	.0195	.0307	.0188
TagMe	.0081	.0089	.0078	.0088	.0076

Table 6.4: Evaluation results of our and other entity linking frameworks for identifying implicit entities in all queries from our query interpretation corpus.

Table 6.5 shows the evaluation results on the subset of queries containing implicit entities. Detecting implicit entities Nordlys' entity retrieval method performed best regarding recall. Since all systems except Nordlys entity retrieval and our system were designed to only return entities explicitly mentioned in a query, they achieved comparatively low results.

The results reveal that solving the problem of implicit entity recognition is a quite complex tasks and even harder than identifying explicit entities. Our algorithmic is able to generate quality candidates but also needs improvement of the scoring method since its an experimental approach.

Algorithm	P	R	F_1	R^*	F_1^*
Nordlys Entity Retrieval	.0477	.3430	.0749	.2869	.0634
Implicit Entity Recognizer	.0127	.1051	.0212	.0798	.0168
TagMe	.0335	.0492	.0282	.0466	.0241
Nordlys Entity Linking	.0345	.0265	.0264	.0216	.0206
Smaph	.0303	.0159	.0184	.0113	.0128

Table 6.5: Evaluation results of our and other entity linking frameworks for identifying implicit entities in the queries containing entities from our query interpretation corpus.

Chapter 7

Conclusion

One of the main goals of this thesis was to refine the rather imprecise traditional definition for entity linking and query interpretation. We distinguished the problem of entity linking into the two distinct subtasks - explicit and implicit entity recognition - which created a reasonable base for generating entity-based interpretations. Further, we adapted the entity-based query interpretation task, making it suitable for our task by allowing the presence of ambiguous entities in queries. With this adjustment we were able to obtain more diverse, precise and reasonable interpretations.

Based on the redefined problems we were able to create a large corpus which eliminates most of the issues of the existing datasets. Since each query was manually annotated by a single expert annotator, the number of conceptual errors was minimized and the consistency of the data was monitored constantly. Further, a large number of queries in our corpus contain entities, which allows the corpus to deliver diverse, meaningful and interesting interpretations for the majority of queries. An advantage of our corpus is that it contains queries with low and high difficulty level, which makes the corpus a valuable base for evaluation of entity recognition frameworks.

We managed to develop algorithmic approaches in order to identify explicit and implicit entities in queries automatically. These approaches focus on providing valuable entities for entity-based query interpretation in a reasonable amount of time. The experiments on these algorithmic approaches showed that a rather simple system like ours can also create comparable results in terms of recall to some extent. In the process of developing our algorithmic solutions, we discovered that the explicit and implicit entity recognition are complex problems. This is due to their dependence on semantic compatibility, which is one of the main keys for good quality entity identification resulting in meaningful interpretations.

Chapter 8

Future Work

The query interpretation corpus we created for this thesis is a large and up-to-date corpus developed for evaluation of entity recognition and query interpretation frameworks. In this chapter, we would like to give some suggestions for potential improvements of the algorithmic approaches and extensions of the corpus.

Although our corpus is rather large in comparison to existing datasets, we suggest this corpus to be extended by additional queries. Since the average difficulty of a query of our corpus is below 2, more queries with a higher difficulty can improve the diversity of queries and increase the significance of potential evaluations on the corpus.

Entity-based query interpretations are only provided for queries of our corpus containing entities. We defined entity-based query interpretation as a semantic compatible segmentation replacing entity-mentioning segments of a query with their corresponding entities. Additionally, query interpretation could also be applied to queries not containing entities as semantic query segmentation without replacements. As a result, all queries of this corpus would be annotated with at least one interpretation. This would be methodically correct since every regular query can be interpreted in at least one meaningful manner. This would also allow a more stable and reasonable evaluation for entity-based query interpretation systems.

One of our future goals is to develop an entity-based query interpretation algorithm based on the algorithmic approaches for explicit and implicit entity recognition presented in this thesis. In order to achieve this, the candidate generation method for this algorithm should be based on more stable and well performing entity recognition approaches. Therefore, our explicit entity recognition tool and especially our implicit entity recognition approach need further improvements. First, the index can be improved by developing an entity specific retrieval model instead of sorting documents by their frequency of

occurrence of a certain entity. Both algorithmic approaches would benefit from this optimization, since such index would be able to return more related entity candidates. Additionally, further methods for scoring the implicit entities should be tested in order to potentially improve the existing one.

Note that all suggested improvements aim to increase the precision and recall of the algorithmic approaches while retaining the high efficiency of the entity recognition in a query.

Appendix A

Entity Taxonomy

- Entity
  - **Name** (e.g., `Barbaro, Bubbles, Max, Maggie`)
  - **Person** (e.g., `Michael Jackson, Elizabeth II`)
  - **God** (e.g., `Zeus, Indra, Danu, Ra`)
  - **Organization**
    - International (e.g., `UN, League of Nations, SEATO`)
    - Show (e.g., `The Cleveland Orchestra, The Beatles`)
    - Family (e.g., `The House of Hamilton, Clan Henderson`)
    - Ethnic Group
      - Nationality (e.g., `Japanese, Israeli`)
    - Sports
      - Pro Sports (e.g., `New York Yankees, Manchester United`)
      - Sports League (e.g., `NFL, Atlantic Coast Conference`)
      - Other (e.g., `Shinagawa Jogging Club`)
    - Corporation
      - Company (e.g., `Toyota, SONY, Microsoft`)
      - Company Group (e.g., `Tata Group, JR`)
      - Other (e.g., `National Rifle Association, BBC`)
    - Political
      - Government (e.g., `Ministry of Finance, US Senate`)
      - Political Party (e.g., `Bharatiya Janata Party, LDP`)
      - Cabinet (e.g., `Thatcher's Cabinet, Major's Cabinet`)
      - Military (e.g., `US Air Force, Royal Navy`)
      - Other (e.g., `Clinton Regime`)

- Entity
  - **Location**
    - Spa (e.g., `Hakone Spa, Fukuchi Spa`)
    - GPE
      - City (e.g., `New York City, Brooklyn`)
      - County (e.g., `Madison County, Orange County`)
      - Province (e.g., `Kansas, Nova Scotia`)
      - Country (e.g., `Japan, UK`)
      - Other (e.g., `Hong Kong, Puerto Rico`)
    - Region
      - Continental (e.g., `North America, Asia`)
      - Domestic (e.g., `New England, East Coast`)
    - Geological
      - Mountain (e.g., `Mount Everest, K2`)
      - Island (e.g., `Florida Keys, Key West`)
      - River (e.g., `Mississippi River, Hudson River`)
      - Lake (e.g., `Lake Michigan, Lake Baikal`)
      - Sea (e.g., `Pacific Ocean, Sea of Japan`)
      - Bay (e.g., `Bay of Bengal, Delaware Bay`)
      - Other (e.g., `Grand Canyon, Altamira Cave`)
    - Astral Body
      - Star (e.g., `Antares, Sirius`)
      - Planet (e.g., `Earth, Moon`)
      - Constellation (e.g., `Taurus, Cassiopeia`)
      - Other (e.g., `Andromeda Galaxy, Callisto`)
    - Other (e.g., `Times Square, Ground Zero`)

- Entity
  - **Facility**
    - Archaelogical
      - Tumulus (e.g., `Daisen-Kofun, Zhaoling`)
      - Other (e.g., `Archaeological Ruins at Moenjodaro`)
    - GOE
      - Public Institution (e.g., `New York Public Library`)
      - School (e.g., `Harvard University`)
      - Research Institute (e.g., `Royal Greenwich Observatory`)
      - Market (e.g., `Tokyo Stock Exchange`)
      - Park (e.g., `Central Park, Banff National Park`)
      - Sports (e.g., `Yankee Stadium, Rose Bowl`)
      - Museum (e.g., `British Museum, Louvre`)
      - Zoo (e.g., `Bronx Zoo, Brooklyn Botanic Garden`)
      - Amusement Park (e.g., `Disneyland`)
      - Theater (e.g., `Palais Garnier, Carnegie Hall`)
      - Worship Place (e.g., `Cathedral of Saint John the Divine`)
      - Car Stop (e.g., `Port Authority Bus Terminal`)
      - Station (e.g., `Grand Central Terminal`)
      - Airport (e.g., `John F. Kennedy International Airport`)
      - Port (e.g., `Port of Hong Kong`)
      - Other (e.g., `White House, Yokota Base`)
    - Line
      - Railroad (e.g., `Lexington Avenue Line`)
      - Road (e.g., `Broadway, U.S. Route 66`)
      - Canal (e.g., `Suez Canal, Panama Canal`)
      - Water Route (e.g., `Empire Route, Seikan Route`)
      - Tunnel (e.g., `Lincoln Tunnel, North Cape Tunnel`)
      - Bridge (e.g., `George Washington Bridge`)
    - Other (e.g., `Empire State Building, Eiffel Tower`)

- Entity
  - **Product**
    - Clothing (e.g., `NIKE`)
    - Money Form (e.g., `Denarius, Quarter Coin`)
    - Drug (e.g., `Aspirin, Aspirin`)
    - Weapon (e.g., `Walther P38`)
    - Stock (e.g., `ATT Stock, MSFT`)
    - Award (e.g., `Nobel Prize, Academy Award`)
    - Decoration (e.g., `Order of the Garter, Order of Lenin`)
    - Character (e.g., `Mickey Mouse, Popeye`)
    - Vehicle
      - Car (e.g., `Toyota Camry, Mercedes-Benz`)
      - Train (e.g., `Pendolino, TGV`)
      - Aircraft (e.g., `Concorde, Mitsubishi Ki-15`)
      - Spaceship (e.g., `H-IIA, Apollo 13`)
      - Ship (e.g., `Titanic, Mayflower`)
      - Other (e.g., `Harley-Davidson, Silver Wing`)
    - Food
      - Other (e.g., `Coca Cola, Guinness Beer`)
    - Art
      - Picture (e.g., `Mona Lisa, Guernica`)
      - Broadcast (e.g., `Lost, American Idol`)
      - Movie (e.g., `Matrix, Titanic`)
      - Show (e.g., `West Side Story, The Phantom of the Opera`)
      - Book (e.g., `Les Miserables, Old Testament`)
      - Other (e.g., `David, Gurdian Deity`)
    - Printing
      - Newspaper (e.g., `The New York Times, The Boston Globe`)
      - Magazine (e.g., `Foreign Affairs, Tel Quel`)
      - Other (e.g., `The Declaration of Independence`)
    - Doctrine
      - Movement (e.g., `American Civil Rights Movement`)
      - Theory (e.g., `Theory of Relativity, Superstring Theory`)
      - Planet (e.g., `Marshall Plan, Appolo Program`)
    - Rule
      - Treaty (e.g., `Treaty of Versailles, Convention of Kanagawa`)
      - Legislation (e.g., `Qubec Act, Tydings-McDuffie Act`)
      - Other (e.g., `Resolution 1441`)
    - Language
      - National (e.g., `English, Japanese`)
    - Other (e.g., `Wii, GDP, Dow Jones Industrial Average`)

- Entity
  - **Event**
    - Occasion
      - Religous (e.g., `Christmas, Easter`)
      - Game (e.g., `2004 Summer Olympics`)
      - Conference (e.g., `Yalta Conference, Taba Summit`)
      - Other (e.g., `Expo '70, Cannes Film Festival`)
    - Incident
      - War (e.g., `World War II, American Revolutionary War`)
      - Other (e.g., `Chernobyl disaster, 9/11`)
    - Natural
      - Disaster (e.g., `Hurricane Katrina, Typhoon Xangsane`)
      - Earthquake (e.g., `Great Alaska Earthquake`)
    - Other (e.g., `Cuban Missile Crisis`)

Bibliography

Apache. Apache Commons Lang 3.1. https://commons.apache.org/proper/commons-lang/, 2011a. 5.1.1

Apache. Apache Commons Math 2.2. http://commons.apache.org/proper/commons-math/, 2011b. 5.1.1

Apache. Apache Commons CSV 1.1. https://commons.apache.org/proper/commons-csv/, 2014. 5.1.1

Apache. Apache Lucene 6.4.2. https://lucene.apache.org/, 2017. 5.1.1

Krisztian Balog and Robert Neumayer. A test collection for entity search in DBpedia. In Gareth J. F. Jones, Paraic Sheridan, Diane Kelly, Maarten de Rijke, and Tetsuya Sakai, editors, *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 737–740, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2034-4. doi: 10.1145/2484028.2484165. URL http://doi.acm.org/10.1145/2484028.2484165. 2.4

Krisztian Balog, Arjen P. de Vries, Pavel Serdyukov, Paul Thomas, and Thijs Westerveld. Overview of the TREC 2009 Entity Track. In *Proceedings of The Eighteenth Text REtrieval Conference, TREC 2009, Gaithersburg, Maryland, USA, November 17-20, 2009*, 2009. 2.4

David Carmel, Ming-Wei Chang, Evgeniy Gabrilovich, Bo-June (Paul) Hsu, and Kuansan Wang. ERD'14: Entity recognition and disambiguation challenge. *SIGIR Forum*, 48(2):63–77, December 2014. ISSN 0163-5840. doi: 10.1145/2701583.2701591. URL http://doi.acm.org/10.1145/2701583.2701591. 1, 2.2, 2.3, 2.4, 4.1

Marco Cornolti, Paolo Ferragina, Massimiliano Ciaramita, Stefan Rüd, and Hinrich Schütze. A piggyback system for joint entity mention detection and

linking in web queries. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, pages 567–578, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee. ISBN 978-1-4503-4143-1. doi: 10.1145/2872427.2883061. URL https://doi.org/10.1145/2872427.2883061. 1, 2.1, 2.2, 5.1.3, 6.2

DBpedia. DBpedia Dataset v3.7. http://wiki.dbpedia.org/data-set-37, 2011. 2.4

DBpedia. DBpedia Dataset 2015-10. http://wiki.dbpedia.org/Downloads2015-10, 2016. 2.4

DBpedia. DBpedia Ontology 2016-10. https://wiki.dbpedia.org/services-resources/ontology, 2017. 5.1.2

Delip Rao. WikiXMLJ. https://github.com/delip/wikixmlj. 5.1.1

Gianluca Demartini, Tereza Iofciu, and Arjen P. De Vries. Overview of the INEX 2009 Entity Ranking Track. In *Proceedings of the Focused Retrieval and Evaluation, and 8th International Conference on Initiative for the Evaluation of XML Retrieval*, INEX'09, pages 254–264, Berlin, Heidelberg, 2010. Springer-Verlag. ISBN 3-642-14555-8, 978-3-642-14555-1. URL http://dl.acm.org/citation.cfm?id=1881065.1881096. 2.4

Facebook Open Source. RocksDB: A persistent key-value store for fast storage environments 4.9.0. https://rocksdb.org/, 2016. 5.1.1

FasterXML. Jackson Core 2.8.1. https://github.com/FasterXML/jackson-core, 2016. 5.1.1

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. ISBN 978-1-4503-0099-5. doi: 10.1145/1871437.1871689. URL http://doi.acm.org/10.1145/1871437.1871689. 605100. 2.1, 2.2, 6.2

Google. Freebase data dumps. https://developers.google.com/freebase/data, 2013. 2.4

Ralph Grishman and Beth Sundheim. Message understanding conference-6: A brief history. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 1*, COLING '96, pages 466–471, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics. doi:

10.3115/992628.992709. URL https://doi.org/10.3115/992628.992709. 2.1

Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. Named entity recognition in query. In *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 267–274, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-483-6. doi: 10.1145/1571941.1571989. URL http://doi.acm.org/10.1145/1571941.1571989. 1, 2.4, 4.6.1

Faegheh Hasibi, Krisztian Balog, and Svein Erik Bratsberg. Entity linking in queries: Tasks and evaluation. In James Allan, W. Bruce Croft, Arjen P. de Vries, and Chengxiang Zhai, editors, *Proceedings of the 2015 International Conference on The Theory of Information Retrieval, ICTIR 2015, Northampton, Massachusetts, USA, September 27-30, 2015*, pages 171–180. ACM, 2015. ISBN 978-1-4503-3833-2. doi: 10.1145/2808194.2809473. URL http://doi.acm.org/10.1145/2808194.2809473. 1, 2.2, 2.3, 4.6.1

Faegheh Hasibi, Krisztian Balog, Darío Garigliotti, and Shuo Zhang. Nordlys: A toolkit for entity-oriented and semantic search. In Kando et al. [2017], pages 1289–1292. ISBN 978-1-4503-5022-8. doi: 10.1145/3077136.3084149. URL http://doi.acm.org/10.1145/3077136.3084149. 1, 2.2, 6.2

Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan. DBpedia-Entity v2: A test collection for entity search. In Kando et al. [2017], pages 1265–1268. ISBN 978-1-4503-5022-8. doi: 10.1145/3077136.3080751. URL http://doi.acm.org/10.1145/3077136.3080751. 1, 2.4, 4.1

Jonathan Hedley. jsoup: Java HTML Parser 1.11.2. https://jsoup.org/, 2017. 5.1.1

Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White, editors. *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, 2017. ACM. ISBN 978-1-4503-5022-8. doi: 10.1145/3077136. URL http://doi.acm.org/10.1145/3077136. (document)

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014. 1, 2.1

Martin Trenkmann. Multimap IO: A 1:n key-value store. 5.1.1

Ruediger Moeller. Fast-Serialization 2.43. `https://github.com/RuedigerMoeller/fast-serialization`, 2015. 5.1.1

Oracle. Java SE Development Kit 8. `http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html`, 2014. 5.1.1

Satoshi Sekine, Kiyoshi Sudo, and Chikashi Nobata. Extended named entity hierarchy. In *LREC*, 2002. 1, 2.1, 3.1

Yahoo. L24 - Yahoo Search Query Log To Entities v1.0. `https://webscope.sandbox.yahoo.com/`. 2.4, 4.1