Chapter 5
Entity Linking

Machine-understanding of text is an extremely challenging problem. The importance of named entities in this regard has been acknowledged early on in natural language processing research; being able to identify entities in a document is a key step towards understanding what the document is about. Like words, entity names can be ambiguous and the same entity may be referred to by many different names. Human readers can use their prior knowledge in combination with the context of a particular *entity mention* (i.e., a text span referring to an entity) to make a decision between the possible choices; for machines, the automatic disambiguation of entity mentions presents many difficulties and challenges. A key enabling component in this process is the availability of large-scale knowledge repositories (such as Wikipedia and various knowledge bases). Having a reference catalog of entities, which are equipped with unique identifiers, the ambiguity of the recognized entity mentions can be resolved by assigning ("linking") them to the corresponding entries in the entity catalog. For instance, there are at least three different Freebase IDs that may be assigned to the mention "Ferrari," depending on whether it refers to the Italian sports car manufacturer (`/m/02_kt`), their racing division that competes in Formula One (`/m/0179v6`), or the founding father Enzo Ferrari (`/m/0gc0s`). The topic of this chapter, *entity linking*, is the task of annotating an input text with entity identifiers from a reference knowledge repository (KR). The output of this annotation process is illustrated in Fig. 5.1.

Linking entities in unstructured text to a structured knowledge repository can greatly empower users in their information consumption activities. For instance, readers of a document can acquire contextual or background information with a single click or can gain easy access to related entities. Entity annotations can also be used in downstream processing to improve retrieval performance or to facilitate better user interaction with search results. We shall look at some of these usages in detail in Part III. Finally, semantic enrichment of documents with entities can prove useful in a number of other text processing tasks as well, including summarization,

K. Balog, *Entity-Oriented Search*, The Information Retrieval Series 39,
https://doi.org/10.1007/978-3-319-93935-3_5

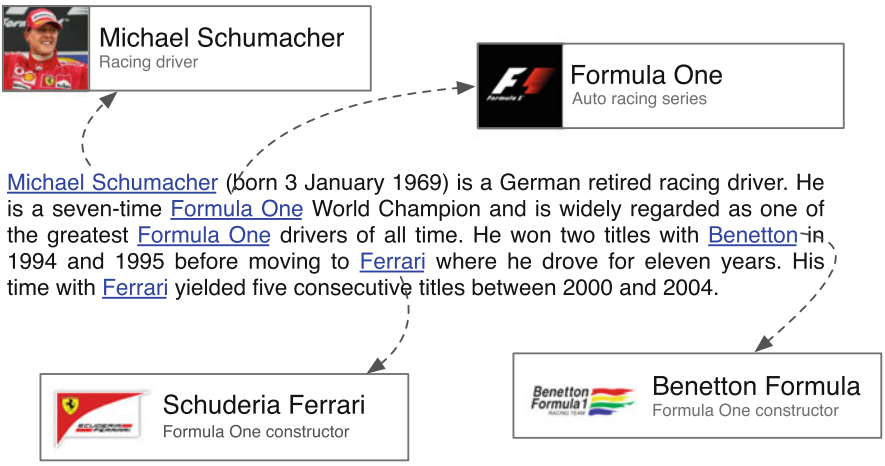

Fig. 5.1 Example of text annotated with entities from Wikipedia. Blue underlined text indicates the linked entity mentions. Text is taken from https://en.wikipedia.org/wiki/Michael_Schumacher

text categorization, topic detection and tracking, knowledge base population, and question answering.

In this chapter our focus is on long text, where it is implicitly assumed that there is in principle always enough context to resolve all entity mentions unambiguously. We discuss the case of short text, such as tweets and search queries, that can possibly have multiple interpretations, in Chap. 7.

The remainder of this chapter is organized as follows. We begin by situating entity linking in the broader context of entity annotation problems in Sect. 5.1. Next, Sect. 5.2 presents an overview of the entity linking task, which is commonly approached as a pipeline of three components. The following sections elaborate on these components: mention detection (Sect. 5.4), candidate selection (Sect. 5.5), and disambiguation (Sect. 5.6). Section 5.7 provides a selection of prominent, publicly available entity linking systems. Evaluation measures and test collections are introduced in Sect. 5.8. We list some useful large-scale resources in Sect. 5.9.

5.1 From Named Entity Recognition Toward Entity Linking

The importance of named entities has long been recognized in natural language processing [64]. Before discussing various approaches to solving the entity linking task, this introductory section gives a brief overview of a range of related entity annotation tasks that have been studied in the past. Table 5.1 provides an overview.

Table 5.1 Overview of named entity recognition and disambiguation tasks

Task	Recognition	Assignment
Named entity recognition	Entities	Entity type
Named entity disambiguation	Entities	Entity identifier/NIL
Wikification	Entities and concepts	Entity identifier/NIL
Entity linking	Entities	Entity identifier

```
<LOC>Silicon Valley</LOC> venture capitalist <PER>Michael Moritz</PER>
said that today's billion-dollar "unicorn" startups can learn from
<ORG>Apple</ORG> founder <PER>Steve Jobs</PER>.
```

Listing 5.1 Text annotated with ENAMEX entity types

5.1.1 Named Entity Recognition

The task of *named entity recognition* (NER) (also known as *entity identification*, *entity extraction*, and *entity chunking*) is concerned with detecting mentions of entities in text and labeling them each with one of the possible entity types. Listing 5.1 shows an example.

Traditionally, NER has focused on three specific types of proper names: person (PER), organization (ORG), and location (LOC). These are collectively known as ENAMEX types [78]. Proper names falling outside the standard ENAMEX types are sometimes considered under an additional fourth type, miscellaneous (MISC). From an information extraction point of view, temporal expressions (TIMEX) and certain types of numerical expressions (NUMEX) (such as currency and percentages) may also be considered as named entities [78] (primarily because the techniques used to recognize them can be similar). The ENAMEX types only allow for a coarse distinction, whereas for certain applications a more fine-grained classification of entities may be desired. Question answering, in particular, has been a driving problem for the development of type taxonomies. Sekine et al. [71] developed an extended named entity hierarchy, with 150 entity types organized in a tree structure. In follow-up work, they extended their hierarchy to 200 types [70] and defined popular attributes for each category to make their type taxonomy an ontology [69]. This approach, however, "relies heavily on an encyclopedia and manual labor" [69]. That is, why in recent years, (existing) type systems of large-scale knowledge bases have been leveraged for NER. For instance, Ling and Weld [53] introduced a (flat) set of 112 types manually curated from Freebase types, while Yosef et al. [83] derived a fine-grained taxonomy with 505 types, organized in a 9 levels deep hierarchy, from YAGO.

NER is approached as a sequence labeling problem, where a categorical label (entity type or not-an-entity) is to be assigned to each term. The dominant technique is to train a machine-learned model on a large collection of annotated documents. Widely used sequence labeling models are *hidden Markov models* [86] and

conditional random fields [23]. Commonly used features include word-level features (word case, punctuation, special characters, word suffixes and prefixes, etc.), character-level n-grams, part-of-speech tags, dictionary lookup features (whether the term is present in a dictionary, often referred to as gazetteer list), and document and corpus features (other entities, position in document, meta information, corpus frequency, etc.). We refer to Nadeau and Sekine [64] for a detailed overview of traditional NER techniques. Recently, neural networks have been shown to achieve state-of-the-art performance without resorting to hand-engineered features [50, 54].

Named entity recognition is a basic functionality in most NLP toolkits (e.g., GATE,[1] Stanford CoreNLP,[2] NLTK,[3] or Apache OpenNLP[4]). NER techniques have been evaluated at the MUC, IREX, CoNLL, and ACE conferences.

5.1.2 Named Entity Disambiguation

Named entity disambiguation (NED), also called *named entity normalization* or *named entity resolution*, is the task of disambiguating entity mentions by assigning entity identifiers to them from some catalog. It is usually assumed that entity mentions have already been detected in the input text (i.e., it has been processed by a NER system). NER is closely related to *word sense disambiguation* (WSD), which is one of the earliest problems in natural language processing. WSD is the process of identifying in what sense (meaning) a word is being used in the given context, when the word has multiple meanings [65]. The possible senses are assigned from some dictionary or thesaurus (typically, WordNet [59]). This way, one can decide, e.g., "whether the word 'church' refers to a building or an institution in a given context" [12]. WSD evaluations exclude proper noun disambiguation (that is addressed separately in NED). It is easy to see that NED and WSD share similarities: they attempt to resolve language ambiguity by mapping words or phrases to unique identifiers. However, there are at least two key differences. First, the input in WSD is a single token (e.g., "church"), while in NED it may be a sequence of tokens (e.g., "Church of England") or an abbreviation (e.g., "CofE"). Second, WSD assumes that each possible word sense has an entry in the dictionary and candidate senses are provided directly; since "named entity mentions vary more than lexical mentions in WSD" [32], candidate entity generation (i.e., identifying the set of entities that the mention possibly refers to) is a critical step in NED. Furthermore, entity mentions without a corresponding catalog entry need are annotated with a special NIL identifier. Nevertheless, the two tasks may be seen as analogous, and early NED approaches were indeed inspired by WSD research [58].

[1] https://gate.ac.uk/.

[2] http://stanfordnlp.github.io/CoreNLP/.

[3] http://www.nltk.org/.

[4] https://opennlp.apache.org/.

The advent of Wikipedia has facilitated large-scale entity recognition and disambiguation by providing a comprehensive catalog of entities along with other invaluable resources (specifically, hyperlinks, categories, and redirection and disambiguation pages; cf. Sect. 2.2). The work by Bunescu and Paşca [2] was the first to perform named entity disambiguation using Wikipedia and was soon followed by others [12, 58].

Within the general problem area of named entity disambiguation, a number of more specific tasks can be distinguished, cf. Table 5.1. Mihalcea and Csomai [58] define *wikification* as "the task of automatically extracting the most important words and phrases in the document, and identifying for each such keyword the appropriate link to a Wikipedia article." The *entity linking* task is to assign mentions of entities in a document to entity identifiers in a reference knowledge repository. We make a conscious distinction between wikification and entity linking, emphasizing that the latter considers only proper names, while the former includes concepts too. Nevertheless, the techniques for the two are essentially the same. We also wish to point out that named entity disambiguation and entity linking are often considered to be synonymous in the NLP community; we make a distinction between the two because of the following important differences:

- Most NED datasets mark up entity mentions explicitly and supply these as part of the input; entity linking is also concerned with the detection of these mentions in the input text.
- Recognizing out-of-KR entities and marking them as NIL is an important subproblem within NED; in entity linking a "closed world" assumption is typically made, i.e., all "possible meanings of a name are known upfront" [37].

We present evaluation methodology and resources in Sect. 5.8.

5.1.3 Entity Coreference Resolution

Another task related to but different from named entity disambiguation is *entity coreference resolution*. Here, entity mentions are to be clustered "such that two mentions belong to the same cluster if and only if they refer to the same entity" [75]. In this task, "there is no explicit mapping onto entities in a knowledge base" [36]. The task is addressed in two flavors: within-document and cross-document coreference resolution. Coreference resolution has been evaluated at the MUC and ACE conferences. We refer to Ng [66] for a survey of approaches.

5.2 The Entity Linking Task

Definition 5.1 *Entity linking* is the task of recognizing entity mentions in text and linking them to the corresponding entries in a knowledge repository.

For simplicity, we will refer to the input text as a *document*. Consider, e.g., the mention "Ferrari" that can refer to any of the entities FERRARI (the Italian sports car manufacturer), SCUDERIA FERRARI (the racing division), FERRARI F2007 (a particular model with which Ferrari competed during the 2007 Formula One season), or ENZO FERRARI (the founder), among others. Based on the context in which the mention occurs (i.e., the document's content), a single one of these candidate entities is selected and linked to the corresponding entry in a knowledge repository. Our current task, therefore, is limited to recognizing entities for which a target entry exists in the reference knowledge repository. Further, it is assumed that the document provides sufficient context for disambiguating entities.

Formally, given an input document d, the task is to generate entity annotations for the document, denoted by $\mathcal{A}_d$, where each annotation $a \in \mathcal{A}_d$ is given as a triple $a = (e, m_i, m_t)$: e is an entity (reference to an entry in the knowledge repository), and m_i and m_t denote the initial and terminal character offsets of the entity's mention in d, respectively. The linked entity mentions in $\mathcal{A}_d$ must not overlap.

Unless pointed out explicitly, the techniques presented below rely on a rather broad definition of a knowledge repository: It provides a catalog of entities, each with one or more names (surface forms), links to other entities, and, optionally, a textual description. The attentive reader might have noticed that we are here using the term knowledge repository as opposed to knowledge base. This is on purpose. The reference knowledge repository that is most commonly used for entity linking is Wikipedia, which is not a knowledge base (cf. Sect. 2.3). General-purpose knowledge bases—DBpedia, Freebase, and YAGO—are also frequently used, since these provide sufficient coverage for most tasks and applications. Also, mapping between their entries and Wikipedia is straightforward. Alternatively, domain-specific resources may also be used, such as the Medical Subject Headings (MeSH) controlled vocabulary.[5]

We refer to Table 5.2 for the notation used throughout this chapter.

5.3 The Anatomy of an Entity Linking System

Over the years, a canonical approach to entity linking has emerged that consists of a pipeline of three components [4, 32], as shown in Fig. 5.2.

[5] https://www.nlm.nih.gov/mesh/.

Table 5.2 Notation used in this chapter

Symbol	Meaning
a	Annotation ($a = (e, m_i, m_t) \in \mathcal{A}_d$)
$\mathcal{A}_d$	Entity annotations for document d
d	Document
d_e	Textual representation (entity description) of entity e
e	Entity ($e \in \mathcal{E}$)
$\mathcal{E}$	Entity catalog (set of all entities)
$\mathcal{E}_d$	Set of all candidate entities in the document d
$\mathcal{E}_m$	Set of candidate entities for mention m
$\mathcal{E}_s$	Set of entities denoted by the surface form s
$\mathcal{L}_e$	Set of links of an entity e
m	Mention (text span) ($m \in \mathcal{M}_d$)
$\mathcal{M}_d$	Set of mentions for document d
$n(m, e)$	Number of times e is a link target of m
s	Surface form ($s \in \mathcal{S}$)
$\mathcal{S}$	Surface form dictionary

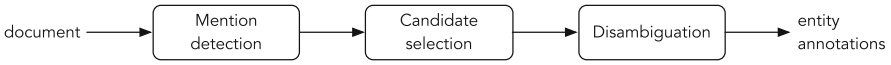

Fig. 5.2 Entity linking pipeline

Mention detection The first component, also known as *extractor* or "*spotter*," is responsible for the identification of text snippets that can potentially be linked to entities. Commonly, mention detection is based on an extensive dictionary of entity names and variations thereof, which we will refer to as (entity) *surface forms*. Mention detection is closely related to the problem of named entity recognition (cf. Sect. 5.1.1) and can indeed be performed with the help of NER techniques. Since only mentions detected by the extractor are considered for subsequent processing in the pipeline, the emphasis here is on achieving high recall.

Candidate selection Next, a set (or ranked list) of candidate entities is generated for each mention. This component is sometimes referred to as the *searcher*. Given that the next step (disambiguation) is typically the computationally most expensive one of all, "an ideal searcher should balance precision and recall to capture the correct entity [for each mention] while maintaining a small set of candidates" [32].

Disambiguation Finally, in the disambiguation step, a single best entity (or none) is selected for each mention, based on the context. This task can be framed as a ranking problem: Given a mention along with the set of candidate entities for that mention, rank candidates based on their likelihood of being the correct referent for the mention. The assigned score can be interpreted as the confidence in the linking, and the annotation (mention-entity pair) may be rejected if its score falls

below a certain (user-defined or machine-learned) threshold. This threshold may also be used to balance the trade-off between precision and recall. Alternatively, disambiguation may be approached as an inference problem, with the objective of optimizing the coherence among all entity linking decisions in the document.

In the following three sections, we look at each of the processes corresponding to the components in Fig. 5.2 in detail.

Before we continue, we note that the organization of the entity linking task along these steps is the most commonly used, but certainly not the only possibility. One particular alternative is where only two stages are distinguished: entity detection and disambiguation [2]. With this approach, mention detection and candidate selection are essentially performed jointly in a single step—a reasonable choice when detection is performed using dictionary-based methods.

5.4 Mention Detection

The first component in the entity linking pipeline is responsible for the detection of entity mentions in the document.

Definition 5.2 A *mention* is a text span (contiguous sequence of terms) in the document that refers to a particular entity. The referred entity may or may not exist in the reference knowledge repository.

Formally, for an input document d, the set of mentions $\mathcal{M}_d$ is to be identified, where each mention $m \in \mathcal{M}_d$ is defined by its initial and terminal character offsets. Bear in mind that the scope of this task is restricted to entities that are contained in the knowledge repository. For that reason, virtually all modern entity linking systems rely on a dictionary of known surface forms to detect mentions; see, e.g., [2, 12, 22, 37, 49, 57, 68]. In a sense, we work under a controlled vocabulary setting; if the text span under consideration does not match any entry in the dictionary then it will not be recognized as a mention, and, consequently, will not be linked to any entity. Therefore, it is vitally important for the dictionary of surface forms to be extensive, including common variations, nicknames, abbreviations, etc. We detail the construction of the surface form dictionary in Sect. 5.4.1.

Assuming that this surface form dictionary $\mathcal{S}$ has been constructed, mention detection works as follows. The input document is parsed and all possible text spans are checked if they are present in $\mathcal{S}$. Text spans are typically token n-grams at length $\leq n$, with n set between 6 and 8. Figure 5.3 illustrates the process. This kind of lexicon-based string matching can be performed efficiently using, e.g., the Aho–Corasick algorithm [1]. To reduce the number of unnecessary dictionary lookups, and thereby increase the efficiency and throughput of mention detection, certain snippets may be disregarded. For example, a system might be instructed not to annotate common words or text spans that are only composed of verbs, adjectives,

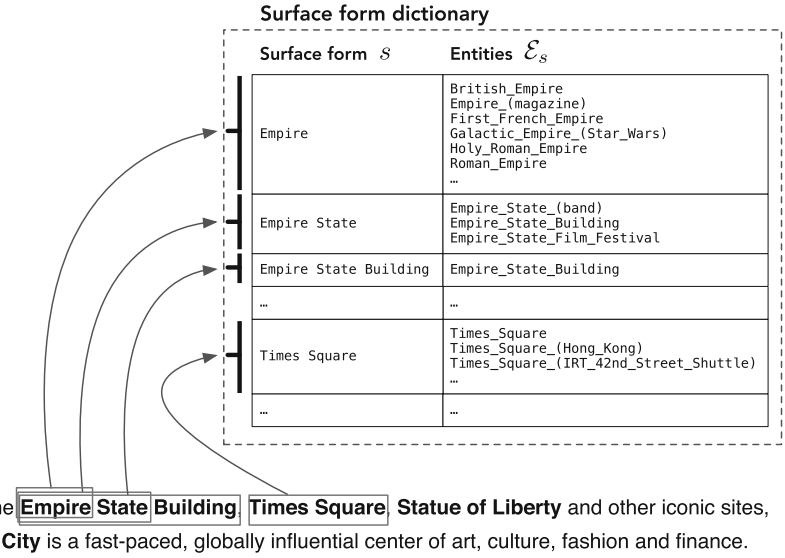

Fig. 5.3 Illustration of dictionary-based mention detection. Detected mentions are boldfaced. The boxes show some of the mentions being looked up (indicated by arrows) in the surface form dictionary. Note the overlaps between mentions

adverbs, and prepositions [57]. Moreover, one might employ simple heuristics, for instance, restrict detection to words that have at least one capitalized letter [16].

Another approach to mention detection is to use NER techniques from natural language processing (cf. Sect. 5.1.1) to identify text spans (typically noun phrases) that refer to named entities; see, e.g., [9, 37, 68]. In this case, an additional string comparison step is involved, where the detected mentions are to be matched against known entity surface forms using some string similarity measure, e.g., edit distance [85], character Dice score, skip bigram Dice score, or Hamming distance [14]. Mentions that do not match any of the dictionary entries, even under a relaxed matching criteria, are likely to denote new, out-of-KR entities.

In practice, it is often desirable that mention detection works directly on the raw text, before any of the standard pre-processing steps, such as tokenization, stopword removal, case-folding, etc., would take place. Sentence boundaries and capitalization can provide cues for recognizing named entities.

5.4.1 Surface Form Dictionary Construction

Dictionary-based mention detection relies on known *surface forms* of entities. These surface forms, also known as *name variants* or *aliases*, are organized in a dictionary

structure (map), $\mathcal{S} : s \rightarrow \mathcal{E}_s$, where the surface form s is the key and it is mapped to the set $\mathcal{E}_s$ of entities.

The reference knowledge repository that entity linking is performed against might already contain a list of name variants for each entity. Below, we focus on the scenario where such lists of aliases are either unavailable or need to be expanded, and discuss how entity surface forms may be obtained from a variety of sources.

Collecting Surface Forms from Wikipedia Wikipedia is a rich resource that has been heavily utilized for extracting name variants. For a given entity, represented by a Wikipedia article, the following sources may be used for collecting aliases:

- *Page title* is the canonical (most common) name for the entity (cf. Sect. 2.2.1.1).
- *Redirect pages* exist for alternative names (including spelling variations and abbreviations) that are frequently used to refer to an entity (cf. Sect. 2.2.3.1).
- *Disambiguation pages* contain a list of entities that share the same name (cf. Sect. 2.2.3.2).
- *Anchor texts* of links pointing to the article can be regarded as aliases of the linked entity (cf. Sect. 2.2.2).
- *Bold texts from first paragraph* generally denote other name variants of the entity.

Recall that not all Wikipedia pages represent entities. With the help of a small set of heuristic rules, it is possible to retain only those Wikipedia articles that refer to named entities (i.e., entities with a proper name title) [2].

Collecting Surface Forms from Other sources The idea of using anchor texts may be generalized from inter-Wikipedia links to links from (external) web pages pointing to Wikipedia articles; one such dictionary resource is presented in Sect. 5.9.1.

The task of identifying name variants is also known as the problem of *entity synonym discovery*. Synonyms might be identified by expanding acronyms [84], or leveraging search results [7, 14] or query click logs [6, 8] from a web search engine.

5.4.2 Filtering Mentions

The surface form dictionary can easily grow (too) large, since, in principle, it contains all strings as keys that have ever been used as anchor text for a link pointing to an entity. While our main focus is on recall, it is still desirable to filter out mentions that are unlikely to be linked to any entity. In this subsection we present two Wikipedia-based measures that may be used for that. Notice that we intentionally call this procedure "filtering mentions," as opposed to "filtering surface forms:" it may be performed early on in the pipeline (i.e., even during the construction of the surface form dictionary) or later, as part of candidate selection or disambiguation.

In their seminal work, Mihalcea and Csomai [58] introduce the concept of *keyphraseness*, which is an estimate of how likely it is that a given text span will be linked to an entity:

$$P(\text{keyphrase}|m) = \frac{|\mathcal{D}_{link}(m)|}{|\mathcal{D}(m)|}, \tag{5.1}$$

where $|\mathcal{D}_{link}(m)|$ is the number of Wikipedia articles where m appears as an anchor text of a link, and $|\mathcal{D}(m)|$ is the number of Wikipedia articles that contain m.

It is essentially the same idea that is captured under the notion of *link probability* in [22]:

$$P(\text{link}|m) = \frac{n_{link}(m)}{n(m)}, \tag{5.2}$$

where $n_{link}(m)$ is the number of times mention m appears as an anchor text of a link, and $n(m)$ denotes the total number of times mention m occurs in Wikipedia (as a link or not).

The main difference between keyphraseness and link probability is that the former considers at most one occurrence (and linking) of a mention per document, while the latter counts all occurrences. (An analogy can be drawn to document frequency vs. term frequency in term importance weighting.) To get a more reliable estimate, it is common to discard mentions that are composed of a single character, made up of only numbers, appear too infrequently in Wikipedia (e.g., less than five times [58]), or have too low relative frequency (e.g., $P(\text{link}|m) < 0.001$ [22]).

5.4.3 Overlapping Mentions

It should be pointed out that the recognized mentions may be overlapping (cf. Fig. 5.3), while the final entity annotations must not overlap. To deal with this, either of two main strategies is employed: (1) containment mentions are dealt with in the mention detection phase, e.g., by dropping a mention if it is subsumed by another mention [29] or by selecting the mention with the highest link probability [22], or (2) overlapping mentions are kept and the decision is postponed to a later stage (candidate selection or disambiguation).

5.5 Candidate Selection

The detection of entity mentions is followed by the selection of candidates for each mention. Let $\mathcal{E}_m$ denote the set of candidate entities for mention m. Potentially, all entities with surface forms matching the mention are candidates: $\mathcal{E}_m = \{e : m \in \mathcal{S}_e\}$. However, as Mendes et al. [57] point out, "candidate selection offers a chance to narrow down the space of disambiguation possibilities." Selecting fewer

Fig. 5.4 Ranking candidate entities based on commonness

candidates can greatly reduce computation time, but it may hurt recall if performed too aggressively. In the process of entity linking, candidate selection plays a crucial role in balancing the trade-off between effectiveness and efficiency. Therefore, candidate selection is often approached as a ranking problem: Given a mention m, determine the prior probability of an entity e being the link target for m: $P(e|m)$. The probabilistic interpretation comes naturally here as it emphasizes the fact that this estimate is based only on the mention, a priori to observing its context. We note that this estimate does not have to be an actual probability; any monotonic scoring function may be used. The top ranked candidate entities, based on a score or rank threshold, are then selected to form $\mathcal{E}_m$.

A highly influential idea by Medelyan et al. [56] is to take into account the overall popularity of entities as targets for a given mention m in Wikipedia. The *commonness* of an entity e is defined as the number of times it is used as a link destination for m divided by the total number of times m appears as a link. In other words, commonness is the maximum-likelihood probability of entity e being the link target of mention m:[6]

$$P(e|m) = \frac{n(m,e)}{\sum_{e' \in \mathcal{E}} n(m,e')} \,. \tag{5.3}$$

Commonness, while typically estimated using Wikipedia (see, e.g., [22, 61]), is not bound to that. It can be based on any entity-annotated text that is large enough to generate meaningful statistics. Using Wikipedia is convenient as the links are of high quality and can be extracted easily from the wiki markup, but a machine-annotated corpus may also be used for the same purpose (see Sect. 5.9.2). We also note that commonness may be pre-computed and conveniently stored in the entity surface form dictionary along with the corresponding entity; see Fig. 5.4.

[6]The attentive reader may notice the similarity to link probability in Eq. (5.2). The difference is that link probability is the likelihood of a given mention being linked to *any* entity, while commonness is the likelihood of a given mention referring to a *particular* entity.

It has also been shown that commonness follows a power law distribution with a long tail of extremely unlikely aliases [61]. Thus one can safely discard entities at the tail end of the distribution (0.001 is a sensible threshold).

5.6 Disambiguation

The last step, which is the heart and soul of the entity linking process, is disambiguation: selecting a single entity, or none, from the set of candidate entities identified for each mention. The simplest solution to resolving ambiguity is to resort to the "most common sense," i.e., select the entity that is most commonly referred to by that mention. This is exactly what the commonness measure, which was discussed in the previous section, captures; see Eq. (5.3). Despite being a naïve solution, it "is a very reliable indicator of the correct disambiguation" [68]. Relying solely on commonness can yield correct answers in many cases and represents a solid baseline [43]. For accurate entity disambiguation, nevertheless, we need to incorporate additional clues.

Modern disambiguation approaches consider three types of evidence: *prior importance* of entities and mentions, *contextual similarity* between the text surrounding the mention and the candidate entity, and *coherence* among all entity linking decisions in the document.

We start off in Sect. 5.6.1 by presenting a set of features for capturing the above three types of evidence. Next, in Sect. 5.6.2, we discuss specific disambiguation approaches that combine this evidence in some way (e.g., using supervised learning or graph-based approaches). The selection of the single best entity for each mention may optionally be followed by a subsequent pruning step: rejecting low confidence or semantically meaningless annotations. We discuss pruning in Sect. 5.6.3.

5.6.1 Features

We discuss features by dividing them into three main groups:

- *Prior importance features* may rely on the entity alone, $f(e)$, or the mention and the entity in combination, $f(e,m)$. In either case, the score is estimated based on prior importance without taking the mention's context into account.
- *Contextual features* are guided by the intuition that the context surrounding an ambiguous entity mention provides valuable additional information for disambiguating it. These features could be written as $f(e,m;d)$, emphasizing that the context is based on the input document. Since we process one document at a time, we will omit d for notational convenience, and simply write $f(e,m)$.

Table 5.3 Features for entity disambiguation

Group	Feature	Description
Prior importance (context-independent)		
	$P(\text{keyphrase}\|m)$	Keyphraseness (likelihood of m being linked)
	$P(\text{link}\|m)$	Link probability (likelihood of m being linked)
	$P(e\|m)$	Commonness (the probability of e being the link target of m)
	$P_{link}(e)$	Fraction of links in the knowledge repository pointing to e
	$P_{pageviews}(e)$	Fraction of (Wikipedia) page views e receives
Contextual		
	$sim_F(m, e)$	Similarity between the context of a mention d_m and the entity's description d_e according to some similarity function F (e.g., cosine, Jaccard, dot product, KL divergence, etc.)
Entity-relatedness		
	$WLM(e, e')$	Wikipedia link-based measure, a.k.a. relatedness
	$PMI(e, e')$	Pointwise mutual information
	$Jaccard(e, e')$	Jaccard similarity
	$\chi^2(e, e')$	χ^2 statistic
	$P(e'\|e)$	Conditional probability

- *Entity-relatedness features* aim at measuring the degree of semantic relatedness between a pair of entities, $f(e, e')$. The ultimate goal is to measure the *coherence* of entity annotations in a document; as we shall see later, this boils down to pairwise entity relatedness.

We discuss these feature groups in turn, highlighting some of the most effective features within each. Table 5.3 provides an overview. The reader will note the large number of features, which reflects the broad diversity of factors that need to be taken into account for effective disambiguation. Unfortunately, there is no systematic and comprehensive feature comparison available. The decision on what features to use (or design) should take into account the characteristics of the particular dataset and examine the trade-off between effectiveness and efficiency.

5.6.1.1 Prior Importance Features

The first group of features consider a single mention m and/or entity e, where e is one of the candidate annotations for that mention, $e \in \mathcal{E}_m$. Neither the text nor other mentions in the document are taken into account, hence the context-independence. We have already introduced keyphraseness (Eq. (5.1)), link probability (Eq. (5.2)), and commonness (Eq. (5.3)), which all belong to this category. These are all related to the popularity of a mention or the popularity of a particular entity given a mention.

To measure the popularity of the entity itself, we present two simple estimates. The first feature is *link prior*, defined as the fraction of all links in the knowledge repository that are incoming links to the given entity [68]:

$$P_{link}(e) = \frac{|\mathcal{L}_e|}{\sum_{e' \in \mathcal{E}} |\mathcal{L}_{e'}|},$$

where $|\mathcal{L}_e|$ denotes the total number of incoming links entity e has. In the case of Wikipedia, $\mathcal{L}_e$ is the number of all articles that link to the entity's Wikipedia page. In the case of a knowledge base, where entities are represented as SPO triples, it is the number of triples where e stands as object.

Entity popularity may also be estimated based on traffic volume, e.g., by utilizing the Wikipedia page view statistics of the entity's page [29]:

$$P_{pageviews}(e) = \frac{pageviews(e)}{\sum_{e' \in \mathcal{E}} pageviews(e')},$$

where $pageviews(e)$ denotes the total number of page views (measured over a certain time period).

When mention detection is performed using NER as opposed to a dictionary-based approach, the match between the mention and the candidate entity's known surface forms should also be considered. Common name-based similarity features include, among others, whether (1) the mention matches exactly the entity name, (2) the mention starts or ends with the entity name, (3) the mention is contained in the entity name or vice versa, and (4) string similarity between the mention and the entity name (e.g., edit distance) [72]. Additionally, the type of the mention, as detected by the NER (i.e., PER, ORG, LOC, etc.), may be compared against the type of the entity in the knowledge repository [14].

5.6.1.2 Contextual Features

One of the simplest and earliest techniques is to compare the surrounding context of a mention with the textual representation (entity description) of the given candidate entity [2, 12]. The context of a mention, denoted as d_m, can be a window of text around the mention, such as the sentence or paragraph containing the mention, or even the entire document. The textual representation of the entity, denoted as d_e, is based on the entity's description in the knowledge repository. As disambiguation is most commonly performed against Wikipedia, it could be, e.g., the whole Wikipedia entity page [2], the first description paragraph of the Wikipedia page [49], or the top-k terms with the highest TF-IDF score from the entity's Wikipedia page [68].[7]

[7]Entity descriptions may also be assembled from a document collection, cf. Sect. 3.2.1. However, those approaches assume that some documents have already been annotated with entities.

Both the mention's context and the entity are commonly represented as bag-of-words. Let $sim_F(m,e)$ denote the contextual similarity between the mention and the entity, using some similarity function F. There is a range of options for the function F, with *cosine similarity* being the most commonly used, see, e.g., [2, 49, 57, 68]:

$$sim_{cos}(m,e) = \frac{\mathbf{d}_m \cdot \mathbf{d}_e}{\| \mathbf{d}_m \| \| \mathbf{d}_e \|},$$

where $\mathbf{d}_m$ and $\mathbf{d}_e$ are the term vectors corresponding to the mention's and entity's representations. Other options for the similarity function F include (but are not limited to): dot product [49], Kullback–Leibler divergence [37], or Jaccard similarity (between word sets) [49].

The representation of context does not have to be limited to bag-of-words. It is straightforward to extend the notion of term vectors to *concept vectors*, to better capture the semantics of the context. Concepts to embed as term vectors could include, among others, named entities (identified using NER) [14], Wikipedia categories [12], anchor text [49], or keyphrases [37].

Additional possibilities to compute context similarity include topic modeling [67, 84] and augmenting the entity's representation using an external corpus [52].

5.6.1.3 Entity-Relatedness Features

In addition to the textual context around a mention, other entities that co-occur in the document can also serve as clues for disambiguation. It can reasonably be assumed that a document focuses on one or at most a few topics. Consequently, the entities mentioned in a document should be topically related to each other. This topical coherence is captured by developing some measure of *relatedness* between a pair of entities. The pairwise entity relatedness scores are then utilized by the disambiguation algorithm to optimize coherence over the set of candidate entities in the document. Notice that we have already touched upon this idea briefly earlier, in Sect. 5.6.1.2, when considering named entities as context. The key difference is that there named entities were treated as string tokens while here we consider the actual entities (given by their identifiers) that are candidates for a particular mention.

Milne and Witten [60] formalize the notion of semantic relatedness for entity linking by introducing the *Wikipedia link-based measure* (WLM), which in later works is often referred to simply as *relatedness*. Modeled after the normalized Google distance measure [10], a close relationship is assumed between two entities if there is a large overlap between the entities linking to them:

$$WLM(e,e') = 1 - \frac{\log\left(\max(|\mathcal{L}_e|, |\mathcal{L}_{e'}|)\right) - \log(|\mathcal{L}_e \cap \mathcal{L}_{e'}|)}{\log(|\mathcal{E}|) - \log\left(\min(|\mathcal{L}_e|, |\mathcal{L}_{e'}|)\right)}, \tag{5.4}$$

where $\mathcal{L}_e$ is the set of entities that link to e and $|\mathcal{E}|$ is the total number of entities. If either of the entities has no links or the two entities have no common links, the score

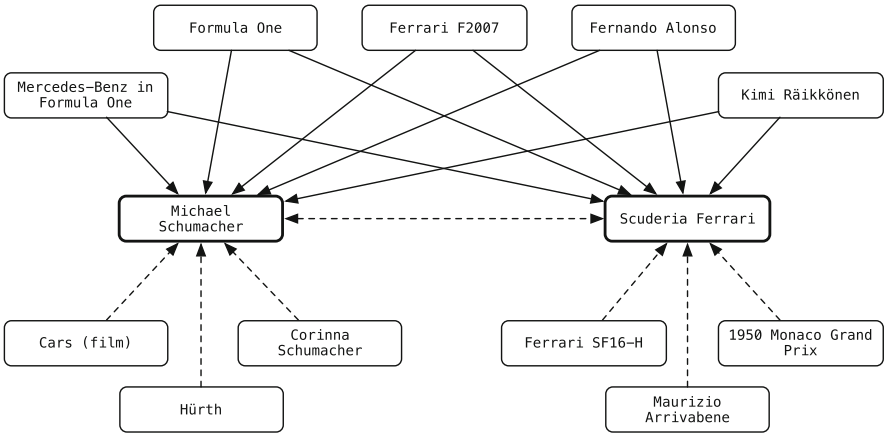

Fig. 5.5 Obtaining the Wikipedia link-based measure between MICHAEL SCHUMACHER and SCUDERIA FERRARI from incoming Wikipedia links (only a selection of links is shown). Solid arrows represent shared links

is set to zero. Figure 5.5 provides an illustration. While relatedness has originally been proposed for incoming Wikipedia links, it may also be considered for outgoing links [68] or for the union of incoming and outgoing links [5]. Also notice that we can equivalently work with relationships in a knowledge base.[8]

Milne and Witten's relatedness measure is the most widely used one and is regarded as the state of the art (see, e.g., [22, 34, 37, 49, 61, 68]), but there are other options, including the Jaccard similarity [30], pointwise mutual information (PMI) [68], or the χ^2 statistic [5]. Notice that all these are symmetric, i.e., $f(e, e') = f(e', e)$ for a particular relatedness function f.

Ceccarelli et al. [5] argue that "a relatedness function should not be symmetric." For example, the relatedness of the UNITED STATES given NEIL ARMSTRONG is intuitively larger than the relatedness of NEIL ARMSTRONG given the UNITED STATES. One effective asymmetric feature they introduce is the conditional probability of an entity given another entity:

$$P(e'|e) = \frac{|\mathcal{L}_{e'} \cap \mathcal{L}_e|}{|\mathcal{L}_e|} .$$

There is obviously a large number of ways one could define relatedness. As we shall see later (in Sect. 5.6.2), having a single relatedness function is preferred to keep the disambiguation process simple (or at least not to make it more complicated). Ceccarelli et al. [5] show that various relatedness measures (a total of 27 in their experiments) can effectively be combined into a single relatedness score using a machine learning approach.

[8]There is a link from e_1 to e_2 if there exists an SPO triple where e_1 appears as subject and e_2 appears as object (the predicate is not considered).

All the features we have presented here so far are based on links. The main reason for favoring link-based features over content-based ones is that the former are cheaper to compute. We need to keep in mind, however, that for entities that do not have many links associated with them (e.g., long-tail entities or entities that have been only recently added to the knowledge repository), these techniques do not work very well. In those cases, one can estimate the semantic relatedness between a pair of entities based on (1) the similarity of the contexts in which they occur (e.g., using keyphrases [36] or n-grams [73]) or (2) the assigned types (e.g., by considering their distance in the type hierarchy [73]).

5.6.2 Approaches

Formally, the disambiguation task is to find the assignment of entities to mentions in a given document: $\Gamma : \mathcal{M}_d \rightarrow \mathcal{E} \bigcup \{\emptyset\}$, where $\emptyset$ denotes the NIL entity assignment. We shall now present various methods and algorithms for establishing this mapping.

> Effective disambiguation needs to combine *local compatibility* (which includes prior importance and contextual similarity) and *coherence* with the other entity linking decisions in the document.

The overall objective function thus can be written as:

$$\Gamma^* = \arg\max_{\Gamma} \Big(\sum_{(m,e)\in\Gamma} \phi(m,e) + \psi(\Gamma) \Big) ,$$

where $\phi(m,e)$ denotes the local compatibility between the mention and the assigned entity, $\psi(\Gamma)$ is the coherence function for all entity annotations in the document, and Γ is a solution (set of mention-entity pairs). This optimization problem is shown to be NP-hard [37, 49, 68, 73], therefore approaches need to resort to approximation algorithms and heuristics.

We distinguish between two main disambiguation strategies, based on whether they consider mentions (1) *individually*, one mention at a time, or (2) *collectively*, all mentions in the document jointly.

Individual disambiguation approaches most commonly cast the task of entity disambiguation as a ranking problem. Each mention is annotated with the highest scoring entity (or as NIL, if the highest score falls below a given threshold):

$$\Gamma^*_{\text{local}}(m) = \arg\max_{e\in\mathcal{E}_m} score(e;m) . \tag{5.5}$$

As discussed earlier, this ranking may be based on a prior popularity (i.e., commonness) alone: $score(e;m) = P(e|m)$. For effective disambiguation, however, it is key

Table 5.4 Entity disambiguation approaches

Approach	Context	Entity interdependence
Most common sense	None	None
Individual local disambiguation	Text	None
Individual global disambiguation	Text and entities	Pairwise
Collective disambiguation	Text and entities	Collective

to consider the context of the mention. Learning-to-rank approaches are well suited for combining multiple signals and have indeed been the most popular choice for this task, see, e.g., [2, 14, 68, 73, 84, 85]. It is important to point out that the fact that mentions are disambiguated individually does not imply that these disambiguation decisions are independent of each other. The interdependence between entity linking decisions may be ignored or may be incorporated (in a pairwise fashion). We refer to these two variants as *local* and *global* approaches, respectively.

Instead of considering each mention individually, once, one might attempt to jointly disambiguate all mentions in the text. Collective disambiguation typically involves an inference process where entity assignments are iteratively updated until some target criterion is met. Table 5.4 provides an overview of approaches.[9]

A final note before we enter into the discussion of specific methods. It is generally assumed (following the *one sense per discourse* assumption [26]) that all the instances of a mention refer to the same entity within the document. If that assumption is lifted, one might employ an iterative algorithm that shrinks the disambiguation context from document to paragraph or even to the sentence level, if necessary [12].

5.6.2.1 Individual Local Disambiguation

Early entity linking approaches [2, 58] focused on *local compatibility* based on contextual features, such as the similarity between the document and the entity's description. Statistics extracted from large-scale entity-annotated data (e.g., Wikipedia), i.e., prior importance, can also be incorporated in the local compatibility score. That is, $score(e;m) = \phi(e,m)$. The local compatibility score can be written in the form of a simple linear combination of features:

$$\phi(e,m) = \sum_i \lambda_i f_i(e,m) , \tag{5.6}$$

[9] Certain approaches from the literature are not immediately straightforward to categorize. We are guided by the following simple rule: It is individual disambiguation if a candidate entity is assigned a score once, and that score does not change. In the case of collective disambiguation, the initially assigned score changes over the course of multiple successive iterations.

where $f_i(e,m)$ can be either a context-independent or a context-dependent feature (see Sects. 5.6.1.1 and 5.6.1.2), and λ_i is the corresponding feature weight. Note that other entity assignments in the document are not taken into consideration.

The idea is to learn the "optimal" combination of features (which is not limited to being a linear combination) from training data. Working within a learning-to-rank framework, each entity-mention pair becomes an instance, described by a feature vector. In the training dataset, the target label is set to 1 for the correct entity and 0 for all other candidate entities.

5.6.2.2 Individual Global Disambiguation

Entity linking can be improved by considering what other entities are mentioned in the document, an idea that was first proposed by Cucerzan [12]. The underlying assumption is that "entities are correlated and consistent with the main topic of the document" [27]. Cucerzan [12] attempts to find an assignment of entities to mentions such that it maximizes the similarity between each entity in the assignment and all possible disambiguations of all other mentions in the document. This can be incorporated as a feature function $f(e,m;\tilde{d})$, where $\tilde{d}$ is a high-dimensional extended document vector that contains all candidate entities for all other mentions in the document.[10] The function then measures the similarity as a scalar product between the entity and the extended document given a particular representation (e.g., topic words or IDs). A disadvantage of this approach is that the extended document vector contains noisy data, as it includes all the incorrect disambiguations as well.

Another disambiguation strategy, proposed by Milne and Witten [61], is to first identify a set of unambiguous mentions. These are then used as context to disambiguate the other mentions in the document. The two main features used for disambiguation are commonness (Eq. (5.3)) and relatedness (Eq. (5.4)). The disadvantage of this approach is the assumption that there exist unambiguous mentions (which, in practice, translates to documents needing to be sufficiently long).

The general idea behind global approaches is to optimize the *coherence* of the disambiguations (entity linking decisions). A true global optimization would be NP-hard, however a good approximation can be computed efficiently by considering pairwise interdependencies for each mention independently. For this reason, the pairwise entity relatedness scores (which we have introduced in Sect. 5.6.1.3) need to be aggregated into a single number. This number will tell us how coherent the given candidate entity is with the rest of the entities in the document. We discuss two specific realizations of this idea.

Ratinov et al. [68] first perform local disambiguation and use the predictions of that system (i.e., the top ranked entity for each mention) in a second, global

[10] Following Cucerzan [12], we use the distinctive notation $\tilde{d}$ to "emphasize that this vector contains information that was not present in the original document."

disambiguation round. Let $\mathcal{E}^\star$ denote the set of linked entities identified by the local disambiguator. The coherence of entity e with all other linked entities in the document is given by:

$$\psi_j(e, \mathcal{E}^\star) = \sum_{\substack{e' \in \mathcal{E}^\star \\ e' \neq e}} g_j(e, e') \,, \tag{5.7}$$

where g_j is a particular pairwise entity relatedness function. We may now extend our scoring function with a second component consisting of global features:

$$score(e; m) = \underbrace{\sum_i \lambda_i f_i(e, m)}_{\phi(e,m)} + \sum_j \Big(\lambda_j \underbrace{\sum_{\substack{e' \in \mathcal{E}^\star \\ e' \neq e}} g_j(e, e')}_{\psi_j(e, \mathcal{E}^\star)} \Big) \,.$$

The λ_i and λ_j coefficients are trained using supervised learning.

An alternative approach is given by Ferragina and Scaiella [22], capitalizing on the fact that commonness and Milne and Witten's relatedness are the two most important features. Their system, called TAGME, introduces a voting mechanism, illustrated in Fig. 5.6, that allows for the combination of these two features, without involving supervised learning. Similarly to Cucerzan [12], a score for a given mention-entity pair is determined by a "collective agreement" between the entity and all possible disambiguations of all other mentions in the document, but in TAGME this is achieved computationally much more efficiently (specifically, time complexity is linear in the number of mentions [21]). Formally, given the set of all mentions in the document $\mathcal{M}_d$, the score of a candidate entity e for a particular mention m is defined as:

$$score(e; m) = \sum_{\substack{m' \in \mathcal{M}_d \\ m' \neq m}} vote(m', e) \,. \tag{5.8}$$

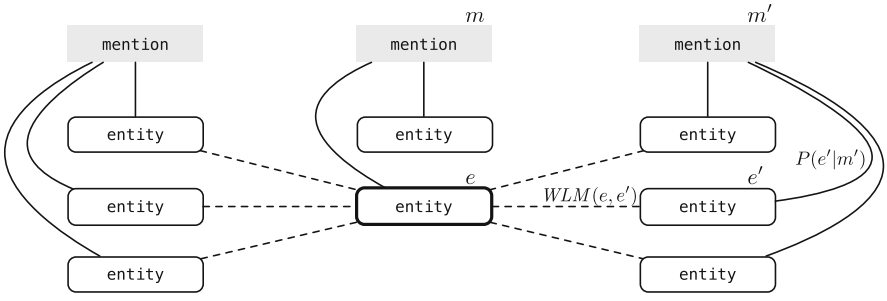

Fig. 5.6 TAGME's voting mechanism. Solid lines connect mentions with the respective candidate entities. A given candidate entity (indicated with the thick border) receives votes from all candidate entities of all mentions in the text (dashed lines)

The *vote* function estimates the agreement between the given entity e and all candidate entities of mention m'. It is computed as the average relatedness between each possible disambiguation e' of m', weighted by its commonness score:

$$vote(m', e) = \frac{\sum_{e' \in \mathcal{E}_{m'}} WLM(e, e') P(e'|m')}{|\mathcal{E}_{m'}|} .$$

Simply returning the entity with the highest score, as defined by Eq. (5.8), is insufficient to obtain an accurate disambiguation, it needs to be combined with other features. For example, this score could be plugged into Eq. (5.6) as a feature function f_i. Another possibility is proposed in [22] in the form of a simple but robust heuristic. Only the highest scoring entities are considered for a given mention, then the one with the highest commonness score among those is selected:

$$\Gamma(m) = \arg\max_{e \in \mathcal{E}_m} \{P(e|m) : e \in \text{top}_\epsilon[score(e; m)]\} . \tag{5.9}$$

That is, the *score* defined in Eq. (5.8) merely acts as a filter. According to Eq. (5.9), only entities in the top ϵ percent of the scores are retained (with ϵ set to 30% in [21]). Out of the remaining entities, the most common sense of the mention will be finally selected.

Individual disambiguation approaches are inherently limited to incorporating interdependencies between entities in a pairwise fashion. This still enforces some degree of coherence among the linked entities, while remaining computationally efficient. Next, we will look at how to model and exploit interdependencies globally.

5.6.2.3 Collective Disambiguation

The main difference when moving from individual to collective disambiguation is how the maximization of coherence between all entity linking decisions in the document is attempted. As we have already pointed out, this optimization is NP-hard. Kulkarni et al. [49] were the first to undertake direct optimization by turning it into a binary integer linear program, and then relaxing it to a linear program (LP). Coherence is measured as the sum of pairwise relatedness between all pairs of linked entities in the document. They show that LP relaxations often give optimal integral solutions. Kulkarni et al. [49] also present a direct greedy hill-climbing approach as an alternative to linear programming, which is comparable both in speed and accuracy to linear programming relaxation.

More recent approaches use a graph structure for collective disambiguation, an idea that was proposed by two independent groups, at about the same time [34, 37]. Mention–entity and entity–entity relations in a document can naturally be represented as a weighted (undirected) graph (termed *referent graph* in [34]). The node

set contains all mentions and all candidate entities corresponding to those mentions. There are two types of edges:

- *Mention–entity edges* capture the local compatibility between the mention and the entity. Edge weights $w(m, e)$ could be measured using a combination of context-independent and context-dependent features, as expressed in Eq. (5.6).
- *Entity–entity edges* represent the semantic relatedness between a pair of entities. Edge weights, $w(e, e')$, are set based on Milne and Witten's relatedness (cf. Eq. (5.4)) [34, 37], but other entity-relatedness functions may also be used.

This graph representation is illustrated in Fig. 5.7. While there is no additional type of evidence compared to what was considered before (namely, local compatibility and pairwise entity relatedness), this representation allows for various graph algorithms to be applied. We note that the graph construction might involve additional heuristics (e.g., "robustness tests" in [37]); we omit these in our discussion.

Hoffart et al. [37] pose the problem of entity disambiguation as that of finding a dense subgraph that contains "all mention nodes and exactly one mention-entity edge for each mention." They propose a greedy algorithm, shown in Algorithm 5.1, that starts from the full graph and iteratively removes the entity node with the lowest weighted degree (along with all its incident edges), provided that each mention node remains connected to at least one entity. The *weighted degree* of an entity node, $wd(e)$ is defined as the sum of the weights of its incident edges. The density of the graph is measured as the *minimum weighted degree* among its entity nodes. From the graphs that are produced in each iteration, the one with the highest density is kept as the solution. This ensures that weak links are captured and the solution is not dominated by a few prominent entities with very high weighted degree.

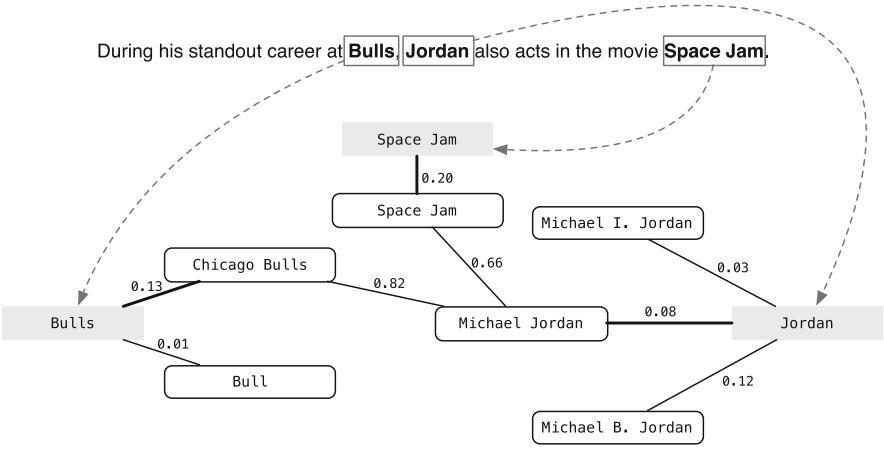

Fig. 5.7 Graph representation for collective disambiguation. Mention nodes are shaded, entity nodes are rounded rectangles. Note that the dashed arrows are not part of the graph. Thick lines indicate the correct mention-entity assignments. Example is taken from [34]

A number of heuristics are applied to ensure that the algorithm is robust. In a pre-processing phase, entities that are "too distant" from the mention nodes are removed. For each entity, the distance from the set of mention nodes is computed as a sum of the squared shortest path distances. Then, only the $k \times |\mathcal{M}_d|$ closest entities are kept ($\mathcal{E}_c$), where $|\mathcal{M}_d|$ is the number of mentions in the document, and k is set to 5 based on experiments. This smaller graph is used as the input to the greedy algorithm. At the end of the iterations, the solution graph may still contain mentions that are connected to more than one entity. The final solution, which maximizes the sum of edge weights, is selected in a post-processing phase. If the graph is sufficiently small, it is feasible to exhaustively consider all possible mention-entity pairs. Otherwise, a faster local (hill-climbing) search algorithm may be used.

Han et al. [34] employ a different graph-based algorithm, *random walk with restarts* [79], for collective disambiguation. "A random walk with restart is a stochastic process to traverse a graph, resulting in a probability distribution over the vertices corresponding to the likelihood those vertices are visited" [31].

Algorithm 5.1: Graph-based entity disambiguation [37]

Input: weighted graph G of mentions and entities
Output: result graph with one edge per mention

`/* pre-processing phase */`
foreach *entity node e* **do**
  $dist_e \leftarrow$ sum of (weighted) shortest paths to each mention
end
keep entities $\mathcal{E}_c$ with lowest $dist_e$, drop the others

`/* main loop */`
objective $\leftarrow \min_{e \in \mathcal{E}_c} wd(e)/|\mathcal{E}_c|$
while *G has non-taboo entity* **do**
  `/* entity is taboo if last candidate for any mention */`
  $e \leftarrow$ non-taboo entity with lowest $wd(e)$
  $\mathcal{E}_c \leftarrow \mathcal{E}_c \setminus e$
  remove e with all its incident edges from G
  $mwd \leftarrow \frac{\min_{e \in \mathcal{E}_c} wd(e)}{|\mathcal{E}_c|}$
  if *mwd* > *objective* **then**
    solution $\leftarrow G$
    objective $\leftarrow mwd$
  end
end

`/* post-processing phase */`
if *feasible* **then**
  process *solution* by enumerating all possible mention-entity pairs
else
  process *solution* by local search
end

Let $\mathbf{v}$ be a starting vector (initial evidence), holding the prior importance associated with each mention node:

$$\mathbf{v}(m) = \frac{TFIDF(m)}{\sum_{m' \in \mathcal{M}_d} TFIDF(m')} ,$$

where $TFIDF(m)$ is the TF-IDF score of mention m.[11] For entity nodes $\mathbf{v}(e) = 0$. Notice that this formulation assumes a directed graph, with edges going from mentions to entities, but not the other way around.

Given the initial evidence, it is propagated through the two types of edges in the graph. We write $\mathbf{T}$ to denote the evidence propagation matrix. The evidence propagation ratio from a mention to its candidate entities is defined as:

$$\mathbf{T}(m \rightarrow e) = \frac{w(m \rightarrow e)}{\sum_{e' \in \mathcal{E}_m} w(m \rightarrow e')} ,$$

and between entities is defined as:

$$\mathbf{T}(e \rightarrow e') = \frac{w(e \rightarrow e')}{\sum_{e'' \in \mathcal{E}_d} w(e \rightarrow e'')} .$$

Let $\mathbf{r}^i$ be a vector holding the probability distribution over nodes at iteration i, corresponding to the likelihood that those nodes are visited. Initially, it is set to be the starting (initial evidence) vector: $\mathbf{r}^0 = \mathbf{v}$. Then, the probability distribution is updated iteratively until convergence:

$$\mathbf{r}^{i+1} = (1 - \alpha)\, \mathbf{r}^i\, \mathbf{T} + \alpha\, \mathbf{v} ,$$

where α is the restart probability (set to 0.1 in [34]).

Once the random walk process has converged to a stationary distribution $\mathbf{r}$, the referent entity for mention m is determined according to:

$$\Gamma(m) = \arg\max_{e \in \mathcal{E}_m} \phi(m, e)\, \mathbf{r}(e) ,$$

where $\phi(m, e)$ is the local compatibility between the mention and the entity, according to Eq. (5.6).

In conclusion, collective disambiguation approaches tend to perform better than individual ones, and they work especially well "when a text contains mentions of a sufficiently large number of entities within a thematically homogeneous context" [37]. On the other hand, the space of possible entity assignments grows combinatorially, which takes a toll on efficiency, in particular for long documents.

[11] The TF component is the normalized frequency of the mention in the document, while the IDF part can be computed using Wikipedia or the Google N-gram dataset.

5.6.3 *Pruning*

Candidate annotations produced by the disambiguation phase can possibly be pruned to discard meaningless or low-confidence annotations. The simplest possible solution is to control this by a confidence threshold; if $score(e; m) < \tau$ then back off from annotating mention m. The threshold τ can be learned from training data.

More advanced ways to pruning are also conceivable; we highlight three of those here. Milne and Witten [60] employ a machine learned classifier to retain only entities that are "relevant enough" to be linked in the sense of what a human editor would consider annotation-worthy (for instance, in Wikipedia, only the first occurrence of an entity is linked). The set of features includes link probability, relatedness, disambiguation confidence, and location and spread of mentions. Ratinov et al. [68] approach pruning as an optimization problem: They decide, for each mention, whether switching the top-ranked disambiguation to NIL would improve the objective function. Finally, Ferragina and Scaiella [22] define the coherence of entity e with all other candidate entity annotations in the text as:

$$coherence(e, \mathcal{E}^\star) = \frac{1}{|\mathcal{E}^\star| - 1} \sum_{\substack{e' \in \mathcal{E}^\star \\ e' \neq e}} WLM(e, e') \,,$$

where $\mathcal{E}^\star$ is the set of linked entities. For each entity, this coherence score is combined with link probability (either as a simple average or as a linear combination) into a pruning score ρ, which is then checked against the pruning threshold.

5.7 Entity Linking Systems

Table 5.5 presents a selection of prominent entity linking systems that have been made publicly available. Their brief summaries follow below.

Table 5.5 Overview of publicly available entity linking systems

System	Reference KR	Online demo	Web API	Source code
AIDA[a]	YAGO2	Yes	Yes	Yes (Java)
DBpedia Spotlight[b]	DBpedia	Yes	Yes	Yes (Java)
Illinois Wikifier[c]	Wikipedia	No	No	Yes (Java)
TAGME[d]	Wikipedia	Yes	Yes	Yes (Java)
Wikipedia Miner[e]	Wikipedia	No	No	Yes (Java)

[a] http://www.mpi-inf.mpg.de/yago-naga/aida/
[b] http://spotlight.dbpedia.org/
[c] http://cogcomp.cs.illinois.edu/page/download_view/Wikifier
[d] https://tagme.d4science.org/tagme/
[e] https://github.com/dnmilne/wikipediaminer

- *AIDA* [37] performs collective disambiguation using a graph-based approach, which we detailed in Sect. 5.6.2.3. Annotations are done against the YAGO2 knowledge base.
- *DBpedia Spotlight* [57] implements a rather straightforward local disambiguation approach; vector space representations of entities are compared against the paragraphs of their mentions using the cosine similarity. Instead of using standard IDF, Mendes et al. [57] introduce the inverse candidate frequency (ICF) weight and employ TF-ICF term weighting. They annotate with DBpedia entities, which can be restricted to certain types or even to a custom entity set defined by a SPARQL query.
- *Illinois Wikifier* [68], a.k.a. GLOW, implements both local and (individual) global disambiguation; their global disambiguation approach is discussed in Sect. 5.6.2.2. In version 2 of their system, Cheng and Roth [9] focus on eliminating mistakes that are "obvious" (to humans) by better understanding the relational structure of the text (e.g., resolving coreference).
- *TAGME* [22] is one of the most popular entity linking systems. It has been designed specifically for efficient annotation of short texts, but it is shown to deliver competitive results on long texts as well. TAGME's one-mention-at-a-time global disambiguation approach is detailed in Sect. 5.6.2.2. The authors have also published an extended report [21] with more algorithmic details and experiments. We further refer to [35] for additional notes on reproducibility.
- *Wikipedia Miner* [61] is a seminal entity linking system that was first to combine commonness and relatedness for (local) disambiguation. It was also the first system with an open-sourced implementation and with wikification provided as a web service (at the time of writing, it is no longer available). See [62] for more technical details and experimental results.

The above selection concentrates on systems that are accompanied by a scholarly publication detailing the underlying methods and approaches. There is a large number of annotation services offered by commercial parties, including but not limited to: AlchemyAPI,[12] AYLIEN Text Analysis API,[13] Google Cloud Natural Language API,[14] Microsoft Entity Linking service,[15] Open Calais,[16] and Rosette Entity Linking API.[17]

[12] http://www.alchemyapi.com/.

[13] http://aylien.com/.

[14] https://cloud.google.com/natural-language/.

[15] https://www.microsoft.com/cognitive-services/en-us/entity-linking-intelligence-service.

[16] http://www.opencalais.com.

[17] https://www.rosette.com/function/entity-linking/.

5.8 Evaluation

In this section, we introduce evaluation measures and test collections.

5.8.1 Evaluation Measures

The overall (end-to-end) performance of an entity linking system is evaluated by comparing the system-generated annotations against a human-annotated gold standard. The measures are set-based: *precision*, *recall*, and *F-measure*. Precision is computed as the fraction of correctly linked entities that have been annotated by the system, while recall is the fraction of correctly linked entities that should be annotated. Since these measures are typically computed over a collection of documents, they can be either *micro-averaged* (aggregated across mentions) or *macro-averaged* (aggregated across documents).

Let us formalize these notions. We write $\mathcal{A}_d$ to denote the annotations generated by the entity linking system and $\hat{\mathcal{A}}_d$ to denote the reference (ground truth) annotations for a single document d. Further, let $\mathcal{A}_\mathcal{D}$ include all annotations for a set $\mathcal{D}$ of documents: $\mathcal{A}_\mathcal{D} = \cup_{d\in\mathcal{D}}\mathcal{A}_d$. Analogously, $\hat{\mathcal{A}}_\mathcal{D}$ is the collection of reference annotations for $\mathcal{D}$. Micro-averaged precision and recall are then defined as:

$$\mathrm{P}_{mic} = \frac{|\mathcal{A}_\mathcal{D} \cap \hat{\mathcal{A}}_\mathcal{D}|}{|\mathcal{A}_\mathcal{D}|}\,, \quad \mathrm{R}_{mic} = \frac{|\mathcal{A}_\mathcal{D} \cap \hat{\mathcal{A}}_\mathcal{D}|}{|\hat{\mathcal{A}}_\mathcal{D}|}\,,$$

where $|\mathcal{A}_\mathcal{D} \cap \hat{\mathcal{A}}_\mathcal{D}|$ denotes the number of matching annotations between the systems and the gold standard (to be defined more precisely later).

Macro-averaged precision and recall are computed as follows:

$$\mathrm{P}_{mac} = \frac{1}{|\mathcal{D}|}\sum_{d\in\mathcal{D}}\frac{|\mathcal{A}_d \cap \hat{\mathcal{A}}_d|}{|\mathcal{A}_d|}\,, \quad \mathrm{R}_{mac} = \frac{1}{|\mathcal{D}|}\sum_{d\in\mathcal{D}}\frac{|\mathcal{A}_d \cap \hat{\mathcal{A}}_d|}{|\hat{\mathcal{A}}_d|}\,.$$

The *F-measure* is computed from the overall precision (P) and recall (R):

$$\mathrm{F1} = \frac{2\,\mathrm{P}\,\mathrm{R}}{\mathrm{P}+\mathrm{R}}\,. \tag{5.10}$$

When entity mentions are also given as input to the entity linking system, *accuracy* is used to assess system performance. Accuracy is defined as the number of correctly linked entity mentions divided by the total number of entity mentions. Thus, in this case, accuracy = precision = recall = F1.

When comparing annotations, the linked entities must match, but we may decide to be lenient with respect to their mentions, i.e., the mention offsets. Let $a = (e, m_i, m_t)$ be an annotation generated by the system and $\hat{a} = (\hat{e}, \hat{m}_i, \hat{m}_t)$ be the

corresponding reference annotation. We define an indicator function for *perfect match* (PM) as follows:

$$match_{PM}(a,\hat{a}) = \begin{cases} 1, & e = \hat{e},\ m_i = \hat{m}_i,\ m_i = \hat{m}_i \\ 0, & \text{otherwise} . \end{cases}$$

Alternatively, we can lessen the requirements for mentions such that it is sufficient for them to overlap. For example, "the Madison Square" and "Madison Square Garden" would be accepted as a match as long as they link to the same entity. The indicator function for *relaxed match* (RM) is defined as:

$$match_{RM}(a,\hat{a}) = \begin{cases} 1, & e = \hat{e},\ [m_i, m_t] \text{ overlaps with } [\hat{m}_i, \hat{m}_t] \\ 0, & \text{otherwise} . \end{cases}$$

Using either flavor of the match function, the number of matching annotations is computed as:

$$|\mathcal{A} \cap \hat{\mathcal{A}}| = \sum_{a \in \mathcal{A}, \hat{a} \in \hat{\mathcal{A}}} match(a,\hat{a}) .$$

5.8.2 Test Collections

Early work used Wikipedia both as the reference KR and as the ground truth for annotations [12, 58, 61]. Using Wikipedia articles as input documents, the task is to "recover" links that were created by Wikipedia contributors. Over the years, the focus has shifted toward entity linking "in the wild," using news articles or web pages as input. This subsection presents the various test collections that have been used in entity linking evaluation. We discuss resources developed by individual researchers and those devised at world-wide evaluation campaigns separately. The main test collections and their key characteristics are summarized in Table 5.6.

5.8.2.1 Individual Researchers

Cucerzan [12] annotated 20 news articles from MSNBC with a total of 755 linkable entity mentions, out of which 113 are NIL (i.e., there is no corresponding Wikipedia article). Milne and Witten [61] used a subset of 50 documents from the AQUAINT text corpus (a collection of newswire stories). Following Wikipedia's style, only the first mention of each entity is linked and only the most important entities are retained. Unlike others, they annotated not only proper nouns but concepts as well. Kulkarni et al. [49] collected and annotated over hundred popular web pages from

Table 5.6 Entity linking and wikification test collections

Name	Reference KR	Document type(s)	#Docs	Annotations type	#Mentions	All[a]	NIL[b]
Individual researchers							
MSNBC [12]	Wikipedia	News	20	Entities	755	Yes	Yes
AQUAINT [61]	Wikipedia	News	50	Entities and concepts	727	No	No
IITB [49]	Wikipedia	Web pages	107	Entities	17,200	Yes	Yes
ACE2004 [68]	Wikipedia	News	57	Entities	306	Yes	Yes
CoNLL-YAGO [37]	YAGO2	News	1393	Entities	34,956	Yes	Yes
Evaluation campaigns							
TAC EL 2009 [74]	Wikipedia	News	3904	Entities	3904	No	Yes
TAC EL 2010 [45]	Wikipedia	News and web	2240	Entities	2240	No	Yes
TAC EL 2011 [51]	Wikipedia	News and web	2250	Entities	2250	No	Yes
TAC EL 2012 [20]	Wikipedia	News and web	2229	Entities	2229	No	Yes
TAC EL 2013 [18]	Wikipedia	News and web	2190	Entities	2190	No	Yes
TAC ELD 2014 [19]	Wikipedia	News and web	138	Entities	5598	Yes	Yes
TAC ELD 2015 [17]	Freebase	News and web	167	Entities	15,581	Yes	Yes
ERD Challenge [3]	Freebase	Web pages	200	Entities	Unknown	Yes	No

[a]Whether all entity mentions are annotated in the documents
[b]Whether out-of-KR entities are annotated as NIL

a handful of domains (sport, entertainment, science and technology, and health). Annotators were instructed to be as exhaustive as possible; this resulted in a total of 17,200 entity mentions with 40% of them annotated as NIL. Ratinov et al. [68] took a subset of the ACE 2004 Coreference dataset as a starting point and annotated mentions (specifically, "the first nominal mention of each co-reference chain" [68]) using crowdsourcing. Hoffart et al. [37] created a dataset based on the CoNLL 2003 Named Entity Recognition task. They annotated 1393 Reuters newswire articles with entities from YAGO2. The collection is split into train, test-A, and test-B partitions. Notably, the original dataset has since been extended to include the corresponding Wikipedia and Freebase entity identifiers as well. Guo and Barbosa [31] released a newer version of the MSNBC, AQUANT, and ACE2004 datasets, with the annotations aligned to the 2013 June version of Wikipedia.[18]

5.8.2.2 INEX Link-the-Wiki

The Link-the-Wiki track ran at INEX between 2007 and 2010 [40–42, 80] with the objective of evaluating link discovery methods. The assumed user scenario is that of creating a new article in Wikipedia; a link discovery system can then automatically suggest both outgoing and incoming links for that article. Note that link detection

[18]http://www.cs.ualberta.ca/~denilson/data/deos14_ualberta_experiments.tgz.

here is approached as a recommendation task, and—unlike in traditional entity linking—it is assumed that a human editor will process the results. Evaluation was performed by selecting an existing Wikipedia article and eradicating all links to and from that page ("orphaning" it), thereby simulating that this is the new document that is being added. The recommended links were assessed manually through a purpose-built interface (with the links originally present in Wikipedia also added to the pool of assessed results). The task is addressed both at the document level (i.e., document-to-document links) and at the element level (i.e., for each prospective anchor, ranking "best entry points" within target documents). System performance is measured using standard IR measures (e.g., MAP). The 2009 and 2010 editions of the track also experimented with a different encyclopedia (Te Ara). That collection, albeit much smaller in size, makes the linking task markedly more complex than using Wikipedia for the reasons that (1) it does not include hyperlinks at all and (2) articles do not represent entities. Since the INEX Link-the-Wiki setup is quite different from our interpretation of the entity linking task, we do not include it in Table 5.6. For further details, we refer to [39].

5.8.2.3 TAC Entity Linking

Entity linking has been running since 2009 at the Knowledge Base Population (KBP) track of the Text Analysis Conference (TAC) [44–47, 55]. Since the track's inception, the task setup has undergone several changes. We start by presenting the initial setting (from 2009) and then discuss briefly how it evolved since.

The Entity Linking (EL) task at TAC KBP is to determine for a given mention string, originating from a particular document, which KB entity is being referred to, or if the entity is not present in the reference KB (NIL). Thus, the focus is on evaluating a single mention per document (referred to as *query*), which is identified in advance, rather than systematically annotating all mentions. The mentions are selected manually by "cherry-picking" those that they are sufficiently confusable, i.e., they have either zero or several KB matches. Additionally, entities with numerous nicknames and shortened or misspelled name variants are also targeted. Entities are either of type person (PER), organization (ORG), or geopolitical entities (GPE). The reference KB is derived from Wikipedia and is further restricted to entities having an infobox. From 2010, web data was also included in addition to newswire documents. An optional entity linking task was also conducted, where systems can only utilize the attributes present in the KB and may not consult the associated Wikipedia page (thereby simulating a setting where a salient and novel entity appears that does not yet have a Wikipedia page). The 2011 edition saw the introduction of two new elements: (1) clustering together NIL mentions (referring to the same out-of-KB entity) and (2) cross-lingual entity linking ("link a given entity from a Chinese document to an English KB" [44]). The cross-lingual version was extended with Spanish in 2012.

In 2014, the task was broadened to end-to-end entity linking and also got re-branded as Entity Discovery and Linking (EDL). Participating systems were

required to automatically identify (and classify) all entity mentions, link each entity mention to the KB, and cluster all NIL mentions [46]. In 2015, the EDL task was expanded from monolingual to trilingual coverage (English, Chinese, and Spanish), extended with two additional entity types (location and facility), and the knowledge repository changed from Wikipedia to a curated subset of Freebase.

Monolingual EDL corresponds to our notion of the entity linking task, except for the two additional subtasks it addresses: (1) classifying entity mentions according to entity types and (2) clustering NIL mentions. These are not intrinsic to entity linking but are important for knowledge base population (which is the ultimate goal of TAC KBP). Evaluation for these subtasks is isolated from entity linking performance. Another key difference in EDL, compared to conventional entity linking, is that EDL focuses only on a handful of entity types.

In Table 5.6 we list the number of test queries (i.e., mentions) for the monolingual entity linking task; additionally, a number of training queries were also made available both as part of the evaluation campaign and in follow-up work [14].

5.8.2.4 Entity Recognition and Disambiguation Challenge

The Entity Recognition and Disambiguation (ERD) Challenge was organized in 2014 by representatives of major web search engine companies, in an effort to make entity linking evaluation more realistic [3]. There are two main differences in contrast to TAC KBP. First, entity linking systems are evaluated in an end-to-end fashion, without providing mention segmentations. Second, each participating team was required to set up a web service for their entity linking system, such that processing times can be measured.

In the "long text" track, the documents to be annotated were pages crawled from the Web (see Sect. 7.3.4.4 for the "short text" track). The reference knowledge repository was Freebase, with entities restricted to specific types and to those having an associated English Wikipedia page. Only proper noun entities are annotated. A set of 100 documents was made available for development and a disjoint set of additional 100 documents was used for testing. Half of the documents were sampled from general web pages, the other half were news articles from msn.com. Evaluation was performed by sending an "evaluation request" to the server hosting the challenge. The evaluation server then sent a set of documents to the participating team's web service for annotation. The returned results were evaluated, with evaluation scores posted on the challenge's leaderboard. Online evaluation took place over a period of time and was divided into train and test phases.

This type of live, online evaluation has its advantages and disadvantages. Asking participants to provide their entity linking system as a service ensures an absolutely fair comparison since (1) the process is completely automated with no possibility of human intervention and (2) annotations for test documents are not released (eliminating the risks of overfitting to a particular test collection). The main drawback is that evaluation is subject to the availability of the evaluation service. Further, as only overall evaluation scores are made available, a detailed

success/failure analysis of the generated annotations is not possible. At the time of writing, the evaluation service is no longer available.[19]

5.8.3 Component-Based Evaluation

The pipeline architecture (cf. Fig. 5.2) makes the evaluation of entity linking systems especially challenging. The main research focus often lies in the disambiguation component, which is the heart of the entity linking process and lends itself to creative algorithmic solutions. However, disambiguation effectiveness is largely influenced by the preceding steps. The fact that improvements are observed on the end-to-end task does not necessarily mean that one disambiguation approach is better than another; it might be a result of more effective mention detection, candidate entity ranking, etc. In general, a fair comparison between two alternative approaches for a given component of an entity linking system can only be made if they share all other elements of the processing pipeline.

The first systematic investigation in this direction was performed by Hachey et al. [32], who implemented and compared three systems: two of the early seminal entity linking systems [2, 12] and the top performing system at TAC 2009 EL [82]. A surprising finding of their study is that much of the variation between the studied systems originates from candidate ranking and not from disambiguation.

Ceccarelli et al. [4] introduced Dexter,[20] an open source framework for entity linking, "where spotting, disambiguation and ranking are well separated and easy to isolate in order to study their performance" [4]. Dexter implements TAGME [22], Wikipedia Miner [61], and the collective linking approach of Han et al. [34].

Cornolti et al. [11] developed and made publicly available the BAT-Framework[21] for comparing publicly available entity annotation systems, namely: AIDA [37], Illinois Wikifier [68], TAGME [22], Wikipedia Miner [61], and DBpedia Spotlight [57]. These systems are evaluated on a number of test collections corresponding to different document genres (news, web pages, and tweets). Linking is done against Wikipedia. Building on top of the BAT-Framework, Usbeck et al. [81] introduced GERBIL,[22] an open-source web-based platform for comparing entity annotation systems. GERBIL extends the BAT-Framework by being able to link to any knowledge repository, not only to Wikipedia. It also includes additional evaluation measures (e.g., for dealing with NIL annotations). Any annotation service can easily be benchmarked by providing a URL to a REST interface that conforms to a given protocol specification. Finally, GERBIL provides persistent URLs for experimental settings, thereby allowing for reproducibility and archival of experimental results.

[19] http://web-ngram.research.microsoft.com/erd2014/.

[20] http://dexter.isti.cnr.it.

[21] http://acube.di.unipi.it/bat-framework/.

[22] http://gerbil.aksw.org.

5.9 Resources

Section 5.7 has introduced entity linking systems that are made publicly available as open source and/or are exposed as a web service (cf. Table 5.5). With these, anyone can annotate documents with entities from a given catalog. We have also discussed benchmarking platforms and test collections in Sect. 5.8. These are essential for those that wish to develop and evaluate their own entity linking system and compare it against existing solutions. This section presents additional resources that may prove useful when building/improving an entity linking system. It could, however, also be the case that one's interest lies not in the entity linking process itself, but rather in the resulting annotations. In Sect. 5.9.2, we present a large-scale web crawl that has been annotated with entities; this resource can be particularly of use for those that are merely "users" of entity annotations and wish to utilize them in downstream processing for some other task.

5.9.1 A Cross-Lingual Dictionary for English Wikipedia Concepts

Recall that a key source of entity surface forms is anchor texts from intra-Wikipedia links (cf. Sect. 5.4.1). The same idea could be extended to inter-Wikipedia links, by considering non-Wikipedia web pages that link to Wikipedia articles. The resource constructed by Spitkovsky and Chang [76] (Google) does exactly this. In addition, they also collect links that point to non-English versions of a given English Wikipedia article. (Notice that the mappings are to all Wikipedia articles, thus there is no distinction made between concepts and entities.) The end result is a cross-lingual surface form dictionary, with names of concepts and entities on one side and Wikipedia articles on the other. The dictionary also contains statistical information, including raw counts and mapping probabilities (i.e., commonness scores). This resource is of great value for the reason that it would be difficult to reconstruct without having access to a comprehensive web crawl. See Table 5.7 for an excerpt.

5.9.2 Freebase Annotations of the ClueWeb Corpora

ClueWeb09 and ClueWeb12 are large-scale web crawls that we discussed earlier in this book (see Sect. 2.1.1). Researchers from Google annotated the English-language web pages from these corpora with entities from the Freebase knowledge

Table 5.7 Excerpt from the dictionary entries matching the surface form (s) "Hank Williams" from the Cross-Lingual Dictionary for English Wikipedia Concepts [76]

Entity (e)	$P(e\|s)$
Hank_Williams	0.990125
Your_Cheatin'_Heart	0.006615
Hank_Williams,_Jr.	0.001629
I	0.000479
Stars_&_Hank_Forever:_The_American_Composers_Series	0.000287
I'm_So_Lonesome_I_Could_Cry	0.000191
I_Saw_the_Light_(Hank_Williams_song)	0.000191
Drifting_Cowboys	0.000095
Half_as_Much	0.000095
Hank_Williams_(Clickradio_CEO)	0.000095

Table 5.8 Excerpt from the Freebase Annotations of the ClueWeb Corpora (FACC)

Mention	Byte offsets	Entity (e)	$P(e\|m,d)$	$P(e\|d)$
PDF	21089, 21092	/m/0600q	0.997636	0.000066
FDA	21303, 21306	/m/032mx	0.999825	0.000571
Food and Drug Administration	21312, 21340	/m/032mx	0.999825	0.000571

base [25], and made these annotations publicly available.[23,24] The system that was used for generating the annotations is proprietary, and as such there is no information disclosed about the underlying algorithm and techniques. It is known, however, that the annotations strove for high precision (which, by necessity, is at the expense of recall) and are of generally high quality. Table 5.8 shows a small excerpt with the annotations created for one of the ClueWeb12 web pages. It can be seen from the table that in addition to the mention (given both as a text span and as byte offsets in the file) and the linked entity, there are two types of confidence scores. The first one ($P(e|m,d)$) is the posterior of an entity given both the mention and the context, while the second one ($P(e|d)$) is the posterior that ignores the mention string and only considers the context of the mention.

5.10 Summary

This chapter has dealt with the task of entity linking: annotating an input text with entity identifiers from a reference knowledge repository. The canonical entity linking approach consists of a pipeline of three components. The first component,

[23] http://lemurproject.org/clueweb09/.

[24] http://lemurproject.org/clueweb12/.

mention detection, is responsible for identifying text spans that may refer to an entity. This is commonly performed using an extensive dictionary of entity surface forms provided in the reference knowledge repository (and possibly augmented with additional name variants from external sources). The second component, *candidate selection*, restricts the set of candidate entities for each mention, by eliminating those that are unlikely to be good link targets for that mention (even though one of their surface forms matches the mention). The third component, *disambiguation*, selects a single entity (or none) from the set of candidate entities identified for each mention. For effective disambiguation, one needs to consider the *local compatibility* between the linked entity and its context as well as the *coherence* between the linked entity and all other entities linked in the document. Two main families of approaches have been delineated, based on whether they perform disambiguation for each mention individually in a single pass or for all entity mentions collectively, using some iterative process. The former is more efficient (an order of magnitude faster), while the latter is more accurate (up to 25% higher F1-score, depending on the particular dataset).

A direct comparison of entity linking systems, based on the reported evaluation scores, is often problematic, due to differences in task definition and evaluation methodology. Further, it is typically difficult to untangle how much each pipeline component has contributed to the observed differences. There are standardization efforts addressing these issues, such as GERBIL [81], by providing an experimental platform for evaluation and diagnostics on reference datasets. According to the results in [81], the best systems reach, depending on the dataset, an F1-score of 0.9.

We have stated that the task of entity linking is one part of the bridge between unstructured and unstructured data. So, how does entity linking enable the task of knowledge base population? Once a document has been found to mention a given entity, that document may be checked to possibly discover new facts with which the knowledge base entry of that entity may be updated. The practical details of this approach will be discussed in the next chapter. Entity annotations can also be utilized to improve document retrieval, as we shall see in Chap. 8.

5.11 Further Reading

Nadeau and Sekine [64] survey the first 15 years of named entity recognition, from 1991 to 2006. An excellent recent survey about entity linking by Shen et al. [72] covers much of the same material as this chapter, with some further pointers. Entity linking is still a very active area of research, with new approaches springing up. Due to space considerations, we did not include approaches based on topic modeling (i.e., LDA-inspired models), see, e.g., [33, 38, 48, 67]. Most recently, semantic embeddings and neural models are gaining popularity in this domain too [24, 28, 77, 87]. Instead of relying on fully automatic techniques, Demartini et al. [13] incorporate human intelligence in the entity linking process, by dynamically generating micro-tasks on an online crowdsourcing platform.

Both entity linking and word sense disambiguation address the lexical ambiguity of language; we have discussed the similarities and differences between the two tasks in Sect. 5.1.2. Moro et al. [63] bring the two tasks to a common ground and present a unified graph-based approach to entity linking and WSD. Motivated by the interdependencies of entity annotation tasks, Durrett and Klein [15] develop a joint model for coreference resolution, named entity recognition, and entity linking.

Finally, in this chapter we have concentrated entirely on a monolingual setting. Cross-lingual entity linking is currently being investigated at the TAC Knowledge Base Population track; for further details, we refer to the TAC proceedings.

References

1. Aho, A.V., Corasick, M.J.: Efficient string matching: An aid to bibliographic search. Commun. ACM **18**(6), 333–340 (1975). doi: 10.1145/360825.360855
2. Bunescu, R., Paşca, M.: Using encyclopedic knowledge for named entity disambiguation. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics, EACL '06, pp. 9–16. Association for Computational Linguistics (2006)
3. Carmel, D., Chang, M.W., Gabrilovich, E., Hsu, B.J.P., Wang, K.: ERD'14: Entity recognition and disambiguation challenge. SIGIR Forum **48**(2), 63–77 (2014). doi: 10.1145/2701583.2701591
4. Ceccarelli, D., Lucchese, C., Orlando, S., Perego, R., Trani, S.: Dexter: An open source framework for entity linking. In: Proceedings of the Sixth International Workshop on Exploiting Semantic Annotations in Information Retrieval, ESAIR '13, pp. 17–20. ACM (2013a). doi: 10.1145/2513204.2513212
5. Ceccarelli, D., Lucchese, C., Orlando, S., Perego, R., Trani, S.: Learning relatedness measures for entity linking. In: Proceedings of the 22nd ACM International Conference on Conference on Information & Knowledge Management, CIKM '13, pp. 139–148. ACM (2013b). doi: 10.1145/2505515.2505711
6. Chakrabarti, K., Chaudhuri, S., Cheng, T., Xin, D.: A framework for robust discovery of entity synonyms. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, pp. 1384–1392. ACM (2012). doi: 10.1145/2339530.2339743
7. Chaudhuri, S., Ganti, V., Xin, D.: Exploiting web search to generate synonyms for entities. In: Proceedings of the 18th International Conference on World Wide Web, WWW '09, pp. 151–160. ACM (2009). doi: 10.1145/1526709.1526731
8. Cheng, T., Lauw, H.W., Paparizos, S.: Entity synonyms for structured web search. IEEE Transactions on Knowledge and Data Engineering **24**(10), 1862–1875 (2012). doi: 10.1109/TKDE.2011.168
9. Cheng, X., Roth, D.: Relational inference for wikification. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pp. 1787–1796. Association for Computational Linguistics (2013)
10. Cilibrasi, R.L., Vitanyi, P.M.B.: The Google similarity distance. IEEE Transactions on Knowledge and Data Engineering **19**(3), 370–383 (2007). doi: 10.1109/TKDE.2007.48
11. Cornolti, M., Ferragina, P., Ciaramita, M.: A framework for benchmarking entity-annotation systems. In: Proceedings of the 22nd International Conference on World Wide Web, WWW '13, pp. 249–260. ACM (2013). doi: 10.1145/2488388.2488411

12. Cucerzan, S.: Large-scale named entity disambiguation based on Wikipedia data. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL '07, pp. 708–716. Association for Computational Linguistics (2007)
13. Demartini, G., Difallah, D.E., Cudré-Mauroux, P.: ZenCrowd: Leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In: Proceedings of the 21st International Conference on World Wide Web, WWW '12, pp. 469–478. ACM (2012). doi: 10.1145/2187836.2187900
14. Dredze, M., McNamee, P., Rao, D., Gerber, A., Finin, T.: Entity disambiguation for knowledge base population. In: Proceedings of the 23rd International Conference on Computational Linguistics, COLING '10, pp. 277–285. Association for Computational Linguistics (2010)
15. Durrett, G., Klein, D.: A joint model for entity analysis: Coreference, typing, and linking. In: Transactions of the Association for Computational Linguistics, vol. 2, pp. 477–490 (2014)
16. Eckhardt, A., Hreško, J., Procházka, J., Smrf, O.: Entity linking based on the co-occurrence graph and entity probability. In: Proceedings of the First International Workshop on Entity Recognition and Disambiguation, ERD '14, pp. 37–44. ACM (2014). doi: 10.1145/2633211.2634349
17. Ellis, J., Getman, J., Fore, D., Kuster, N., Song, Z., Bies, A., Strassel, S.M.: Overview of linguistic resources for the TAC KBP 2015 evaluations: Methodologies and results. In: Proceedings of the 2015 Text Analysis Conference, TAC '15. NIST (2015)
18. Ellis, J., Getman, J., Mott, J., Li, X., Griffitt, K., Strassel, S.M., Wright, J.: Linguistic resources for 2013 knowledge base population evaluations. In: Proceedings of the 2013 Text Analysis Conference, TAC '13. NIST (2013)
19. Ellis, J., Getman, J., Strassel, S.M.: Overview of linguistic resources for the TAC KBP 2014 evaluations: Planning, execution, and results. In: Proceedings of the 2014 Text Analysis Conference, TAC '14. NIST (2014)
20. Ellis, J., Li, X., Griffitt, K., Strassel, S.M., Wright, J.: Linguistic resources for 2012 knowledge base population evaluations. In: Proceedings of the 2012 Text Analysis Conference, TAC '12. NIST (2012)
21. Ferragina, P., Scaiella, U.: Fast and accurate annotation of short texts with Wikipedia pages. CoRR **abs/1006.3** (2010a)
22. Ferragina, P., Scaiella, U.: 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, pp. 1625–1628. ACM (2010b). doi: 10.1145/1871437.1871689
23. Finkel, J.R., Grenager, T., Manning, C.: Incorporating non-local information into information extraction systems by Gibbs sampling. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL '05, pp. 363–370. Association for Computational Linguistics (2005). doi: 10.3115/1219840.1219885
24. Francis-Landau, M., Durrett, G., Klein, D.: Capturing semantic similarity for entity linking with convolutional neural networks. In: Proceedings of the North American Association for Computational Linguistics, NAACL '16. Association for Computational Linguistics (2016)
25. Gabrilovich, E., Ringgaard, M., Subramanya, A.: FACC1: Freebase annotation of Clueweb corpora, version 1. Tech. rep., Google, Inc. (2013)
26. Gale, W.A., Church, K.W., Yarowsky, D.: One sense per discourse. In: Proceedings of the Workshop on Speech and Natural Language, HLT '91, pp. 233–237. Association for Computational Linguistics (1992). doi: 10.3115/1075527.1075579
27. Ganea, O.E., Ganea, M., Lucchi, A., Eickhoff, C., Hofmann, T.: Probabilistic bag-of-hyperlinks model for entity linking. In: Proceedings of the 25th International Conference on World Wide Web, WWW '16, pp. 927–938. International World Wide Web Conferences Steering Committee (2016). doi: 10.1145/2872427.2882988
28. Ganea, O.E., Hofmann, T.: Deep joint entity disambiguation with local neural attention. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 2619–2629. Association for Computational Linguistics (2017). doi: 10.18653/v1/D17-1277

29. Gattani, A., Lamba, D.S., Garera, N., Tiwari, M., Chai, X., Das, S., Subramaniam, S., Rajaraman, A., Harinarayan, V., Doan, A.: Entity extraction, linking, classification, and tagging for social media: A Wikipedia-based approach. Proceedings of the VLDB Endowment **6**(11), 1126–1137 (2013). doi: 10.14778/2536222.2536237
30. Guo, S., Chang, M.W., Kiciman, E.: To link or not to link? - A study on end-to-end tweet entity linking. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1020–1030. Association for Computational Linguistics (2013)
31. Guo, Z., Barbosa, D.: Robust entity linking via random walks. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM '14, pp. 499–508. ACM (2014). doi: 10.1145/2661829.2661887
32. Hachey, B., Radford, W., Nothman, J., Honnibal, M., Curran, J.R.: Evaluating entity linking with Wikipedia. Artif. Intell. **194**, 130–150 (2013). doi: 10.1016/j.artint.2012.04.005
33. Han, X., Sun, L.: An entity-topic model for entity linking. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL '12, pp. 105–115. Association for Computational Linguistics (2012)
34. Han, X., Sun, L., Zhao, J.: Collective entity linking in web text: A graph-based method. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '11, pp. 765–774. ACM (2011). doi: 10.1145/2009916.2010019
35. Hasibi, F., Balog, K., Bratsberg, S.E.: On the reproducibility of the TAGME entity linking system. In: Proceedings of the 38th European conference on Advances in Information Retrieval, ECIR '16, pp. 436–449. Springer (2016). doi: 10.1007/978-3-319-30671-1_32
36. Hoffart, J., Seufert, S., Nguyen, D.B., Theobald, M., Weikum, G.: KORE: Keyphrase overlap relatedness for entity disambiguation. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, CIKM '12, pp. 545–554. ACM (2012). doi: 10.1145/2396761.2396832
37. Hoffart, J., Yosef, M.A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., Weikum, G.: Robust disambiguation of named entities in text. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11, pp. 782–792. Association for Computational Linguistics (2011)
38. Houlsby, N., Ciaramita, M.: A scalable Gibbs sampler for probabilistic entity linking. In: Advances in Information Retrieval, *Lecture Notes in Computer Science*, vol. 8416, pp. 335–346. Springer (2014). doi: 10.1007/978-3-319-06028-6_28
39. Huang, W.C.D.: Evaluation framework for focused link discovery. Ph.D. thesis, Queensland University of Technology (2011)
40. Huang, W.C.D., Geva, S., Trotman, A.: Overview of the INEX 2008 Link the Wiki track. In: Geva, S., Kamps, J., Trotman, A. (eds.) Advances in Focused Retrieval, *Lecture Notes in Computer Science*, vol. 5631, pp. 314–325. Springer (2009). doi: 10.1007/978-3-642-03761-0_32
41. Huang, W.C.D., Geva, S., Trotman, A.: Overview of the INEX 2009 Link the Wiki track. In: Geva, S., Kamps, J., Trotman, A. (eds.) Focused Retrieval and Evaluation, *Lecture Notes in Computer Science*, vol. 6203, pp. 312–323. Springer (2010). doi: 10.1007/978-3-642-14556-8_31
42. Huang, W.C.D., Xu, Y., Trotman, A., Geva, S.: Overview of INEX 2007 Link the Wiki track. In: Fuhr, N., Kamps, J., Lalmas, M., Trotman, A. (eds.) Focused Access to XML Documents, *Lecture Notes in Computer Science*, vol. 4862, pp. 373–387. Springer (2008). doi: 10.1007/978-3-540-85902-4_32
43. Ji, H., Grishman, R.: Knowledge base population: Successful approaches and challenges. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, HLT '11, pp. 1148–1158. Association for Computational Linguistics (2011)
44. Ji, H., Grishman, R., Dang, H.T.: Overview of the TAC 2011 Knowledge Base Population track. In: Proceedings of the 2010 Text Analysis Conference, TAC '11. NIST (2011)

45. Ji, H., Grishman, R., Dang, H.T., Griffitt, K., Ellis, J.: Overview of the TAC 2010 Knowledge Base Population track. In: Proceedings of the 2010 Text Analysis Conference, TAC '10. NIST (2010)
46. Ji, H., Nothman, J., Hachey, B.: Overview of TAC-KBP2014 Entity discovery and linking tasks. In: Proceedings of the 2014 Text Analysis Conference, TAC '14. NIST (2014)
47. Ji, H., Nothman, J., Hachey, B., Florian, R.: Overview of TAC-KBP2015 Tri-lingual entity discovery and linking. In: Proceedings of the 2015 Text Analysis Conference, TAC '15. NIST (2015)
48. Kataria, S.S., Kumar, K.S., Rastogi, R.R., Sen, P., Sengamedu, S.H.: Entity disambiguation with hierarchical topic models. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '11, pp. 1037–1045. ACM (2011). doi: 10.1145/2020408.2020574
49. Kulkarni, S., Singh, A., Ramakrishnan, G., Chakrabarti, S.: 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, pp. 457–466. ACM (2009). doi: 10.1145/1557019.1557073
50. Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural architectures for named entity recognition. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 260–270. Association for Computational Linguistics (2016). doi: 10.18653/v1/N16-1030
51. Li, X., Ellis, J., Griffit, K., Strassel, S., Parker, R., Wright, J.: Linguistic resources for 2011 knowledge base population evaluation. In: Proceedings of the 2011 Text Analysis Conference, TAC '11. NIST (2011)
52. Li, Y., Wang, C., Han, F., Han, J., Roth, D., Yan, X.: Mining evidences for named entity disambiguation. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '13, pp. 1070–1078. ACM (2013). doi: 10.1145/2487575.2487681
53. Ling, X., Weld, D.S.: Fine-grained entity recognition. In: In Proceedings of the 26th AAAI Conference on Artificial Intelligence, AAAI '12 (2012)
54. Ma, X., Hovy, E.: End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1064–1074. Association for Computational Linguistics (2016). doi: 10.18653/v1/P16-1101
55. McNamee, P., Dang, H.T.: Overview of the TAC 2009 Knowledge Base Population track. In: Proceedings of the 2009 Text Analysis Conference, TAC '09. NIST (2009)
56. Medelyan, O., Witten, I.H., Milne, D.: Topic indexing with Wikipedia. In: Bunescu, R., Gabrilovich, E., Mihalcea, R. (eds.) Proceedings of AAAI Workshop on Wikipedia and Artificial Intelligence: An Evolving Synergy, vol. 1, pp. 19–24. AAAI, AAAI Press (2008)
57. Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: Dbpedia Spotlight: Shedding light on the web of documents. In: Proceedings of the 7th International Conference on Semantic Systems, I-Semantics '11, pp. 1–8 (2011)
58. Mihalcea, R., Csomai, A.: Wikify! - Linking documents to encyclopedic knowledge. In: Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management, CIKM '07, pp. 233–242. ACM (2007). doi: 10.1145/1321440.1321475
59. Miller, G.A.: WordNet: A lexical database for English. Communications of the ACM **38**(11), 39–41 (1995). doi: 10.1145/219717.219748
60. Milne, D., Witten, I.H.: An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In: Proceeding of AAAI Workshop on Wikipedia and Artificial Intelligence: An Evolving Synergy, pp. 25–30. AAAI Press (2008a)
61. Milne, D., Witten, I.H.: Learning to link with Wikipedia. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, CIKM '08, pp. 509–518 (2008b)
62. Milne, D., Witten, I.H.: An open-source toolkit for mining Wikipedia. Artificial Intelligence **194**, 222–239 (2013)

63. Moro, A., Raganato, A., Navigli, R.: Entity linking meets word sense disambiguation: A unified approach. Transactions of the Association for Computational Linguistics **2**, 231–244 (2014)
64. Nadeau, D., Sekine, S.: A survey of named entity recognition and classification. Lingvisticae Investigationes **30**(1), 3–26 (2007). doi: 10.1075/li.30.1.03nad
65. Navigli, R.: Word sense disambiguation: A survey. ACM Comput. Surv. **41**(2), 10:1–10:69 (2009). doi: 10.1145/1459352.1459355
66. Ng, V.: Supervised noun phrase coreference research: The first fifteen years. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10, pp. 1396–1411. Association for Computational Linguistics (2010)
67. Pilz, A., Paaß, G.: From names to entities using thematic context distance. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, CIKM '11, pp. 857–866. ACM (2011). doi: 10.1145/2063576.2063700
68. Ratinov, L., Roth, D., Downey, D., Anderson, M.: 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, pp. 1375–1384. Association for Computational Linguistics (2011)
69. Sekine, S.: Extended named entity ontology with attribute information. In: Proceedings of the Sixth International Language Resources and Evaluation, LREC '08. ELRA (2008)
70. Sekine, S., Nobata, C.: Definition, dictionaries and tagger for extended named entity hierarchy. In: Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC '04. ELRA (2004)
71. Sekine, S., Sudo, K., Nobata, C.: Extended named entity hierarchy. In: Third International Conference on Language Resources and Evaluation, LREC '02. ELRA (2002)
72. Shen, W., Wang, J., Han, J.: Entity linking with a knowledge base: Issues, techniques, and solutions. IEEE Trans. Knowl. Data Eng. **27**(2), 443–460 (2015). doi: 10.1109/TKDE.2014.2327028
73. Shen, W., Wang, J., Luo, P., Wang, M.: LIEGE: link entities in web lists with knowledge base. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, pp. 1424–1432. ACM (2012). doi: 10.1145/2339530.2339753
74. Simpson, H., Strassel, S., Parker, R., McNamee, P.: Wikipedia and the web of confusable entities: Experience from entity linking query creation for TAC 2009 Knowledge base population. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation, LREC '10. ELRA (2010)
75. Singh, S., Subramanya, A., Pereira, F., McCallum, A.: Large-scale cross-document coreference using distributed inference and hierarchical models. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, HLT '11, pp. 793–803. Association for Computational Linguistics (2011)
76. Spitkovsky, V.I., Chang, A.X.: A cross-lingual dictionary for English Wikipedia concepts. In: Proceedings of the Eight International Conference on Language Resources and Evaluation, LREC '12. ELRA (2012)
77. Sun, Y., Lin, L., Tang, D., Yang, N., Ji, Z., Wang, X.: Modeling mention, context and entity with neural networks for entity disambiguation. In: Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI'15, pp. 1333–1339. AAAI Press (2015)
78. Sundheim, B.M.: Overview of results of the MUC-6 evaluation. In: Message Understanding Conference, MUC-6, pp. 13–31 (1995). doi: 10.3115/1072399.1072402
79. Tong, H., Faloutsos, C., Pan, J.Y.: Fast random walk with restart and its applications. In: Proceedings of the Sixth International Conference on Data Mining, ICDM '06, pp. 613–622. IEEE Computer Society (2006). doi: 10.1109/ICDM.2006.70
80. Trotman, A., Alexander, D., Geva, S.: Overview of the INEX 2010 Link the Wiki track. In: Geva, S., Kamps, J., Schenkel, R., Trotman, A. (eds.) Comparative Evaluation of Focused Retrieval, *Lecture Notes in Computer Science*, vol. 6932, pp. 241–249. Springer (2011). doi: 10.1007/978-3-642-23577-1_22

81. Usbeck, R., Röder, M., Ngonga Ngomo, A.C., Baron, C., Both, A., Brümmer, M., Ceccarelli, D., Cornolti, M., Cherix, D., Eickmann, B., Ferragina, P., Lemke, C., Moro, A., Navigli, R., Piccinno, F., Rizzo, G., Sack, H., Speck, R., Troncy, R., Waitelonis, J., Wesemann, L.: GERBIL – general entity annotation benchmark framework. In: Proceedings of the 24th International World Wide Web Conference, WWW '15. International World Wide Web Conferences Steering Committee (2015). doi: 10.1145/2736277.2741626
82. Varma, V., Reddy, V.B., Kovelamudi, S., Bysani, P., Gsk, S., Kumar, N.K., B, K.R., Kumar, K., Maganti, N.: IIIT Hyderabad at TAC 2009. In: Proceedings of the 2009 Text Analysis Conference, TAC '09. NIST (2009)
83. Yosef, M.A., Bauer, S., Hoffart, J., Spaniol, M., Weikum, G.: HYENA: Hierarchical type classification for entity names. In: Proceedings of the 24th International Conference on Computational Linguistics, COLING '12, pp. 1361–1370. Association for Computational Linguistics (2012)
84. Zhang, W., Sim, Y.C., Su, J., Tan, C.L.: Entity linking with effective acronym expansion, instance selection and topic modeling. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three, IJCAI'11, pp. 1909–1914. AAAI Press (2011). doi: 10.5591/978-1-57735-516-8/IJCAI11-319
85. Zheng, Z., Li, F., Huang, M., Zhu, X.: Learning to link entities with knowledge base. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, HLT '10, pp. 483–491. Association for Computational Linguistics (2010)
86. Zhou, G., Su, J.: Named entity recognition using an HMM-based chunk tagger. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02, pp. 473–480. Association for Computational Linguistics (2002). doi: 10.3115/1073083.1073163
87. Zwicklbauer, S., Seifert, C., Granitzer, M.: Robust and collective entity disambiguation through semantic embeddings. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '16, pp. 425–434. ACM (2016). doi: 10.1145/2911451.2911535