Knowledge and Information Systems (2022) 64:325–351
https://doi.org/10.1007/s10115-021-01642-9

REGULAR PAPER

Named entity disambiguation in short texts over knowledge graphs

Wissem Bouarroudj[1] · Zizette Boufaida[1] · Ladjel Bellatreche[2]

Received: 22 March 2021 / Revised: 8 December 2021 / Accepted: 11 December 2021 /
Published online: 3 January 2022

Abstract
The ever-growing usage of knowledge graphs (KGs) positions named entity disambiguation (NED) at the heart of designing accurate KG-driven systems such as query answering systems (QAS). According to the current research, most studies dealing with NED on KGs involve long texts, which is not the case of short text fragments, identified by their limited contexts. The accuracy of QASs strongly depends on the management of such short text. This limitation motivates this paper, which studies the NED problem on KGs, involving only short texts. First, we propose a NED approach including the following steps: (i) context expansion using WordNet to measure its similarity to the resource context. (ii) Exploiting coherence between entities in queries that contain more than one entity, such as "Is Michelle Obama the wife of Barack Obama?". (iii) Taking into account the relations between words to calculate their similarity with the properties of a resource. (iv) the use of syntactic features. The NED solution approach is compared to state-of-the-art approaches using five datasets. The experimental results show that our approach outperforms these systems by 27% in the F-measure. A system called *Welink*, implementing our proposal, is available on GitHub, and it is also accessible via a REST API.

Keywords Named entity disambiguation · Entity linking · Short texts · Queries · Semantic and syntactic features · Linked open data

✉ Ladjel Bellatreche
bellatreche@ensma.fr

Wissem Bouarroudj
wissem.bouarroudj@univ-constantine2.dz

Zizette Boufaida
zizette.boufaida@univ-constantine2.dz

[1] LIRE Laboratory, Abdelhamid Mehri Constantine 2 University, Constantine, Algeria

[2] LIAS/ISAE-ENSMA, Poitiers University, Poitiers, France

1 Introduction

The explosion of intelligent software assistants such as *Apple Siri*, *Microsoft Cortana*, *Ok Google* and *WolframAlpha* makes our daily lives part of the *query answering world*. In the latter, anyone connected to a digital tool is looking for answers to questions. These demands increase mainly after the appearance of spectacular events related to pandemics (COVID-19), tsunamis, terrorism, politics, sport, and entertainment. Query Answering Systems (QASs) are one of the serious technologies that respond to these demands. A QAS aims at providing a concise response to a natural language query (NLQ). The use of natural languages has contributed to the popularization of QASs. Their development has become a topical issue. Searching on Google Scholar, we found more than 15 surveys published over 2016–2020, mainly focusing on QASs in the age of Web of Data. A recent survey published in 2017 [30] mentions a surprising number of 62 QASs developed since 2010.

The Web of Data provides an interesting characteristic that exploits the first generation of QASs that considers corpora of structured data-repositories mainly related to a closed domain. Baseball [21] is considered one of the first QASs to answer questions about the performance of baseball in the USA. The explosion of Semantic Web technologies is helping to put a lot of structured data on the web in the form of Knowledge Graphs (KGs) [23], including Linked Open Data. One of the main objectives of the QASs over KGs is to make this valuable data accessible and usable by end-users [44]. There is a variety of KGs covering several domains. In [5], a classification of KGs is given, where three main categories are distinguished: (1) generalist KGs such as *DBpedia* [3], *FreeBase* [9], and *Google Knowledge Graph* [56]; (2) specialized KGs associated with a specific field (Facebook Knowledge Graph, Amazon Knowledge Graph, and Central Banks); and (3) enterprise KGs such as *Enterprise Knowledge Graph*.[1] This type of data is well-structured thanks to the Resource Description Framework (RDF) structure (triplet: $<subject, predicate, object>$) and can be searched using the SPARQL language. In a KG, each node represents an entity (e.g. a person, a place, or a concept), and each label represents a relation used to link entities (e.g. a birthplace that links two entities: a person and a place). Several QASs over KGs have been proposed [30], and examples include KEQA [33], Qanswer [14], GFMed [39] and SINA [53]. Developing a QAS is time-consuming, whatever the type of its corpus (structured data, semi-structured data, free textual documents, Web data, and multimodal repositories). Indeed, it comprises several complex components. Usually, a large majority of new QASs are developed from scratch. An interesting paper published in the WWW 2018 conference summarizing the results of an EU H2020 project highlights an interesting question concerning the increased reuse of already available QAS components [55]. This suggests several directions be taken into account by researchers when developing a new QAS: (i) promoting modularity in the development of a QAS; (ii) making each developed component available; and (iii) facilitating their reuse. The adoption of these directions facilitates the comparison of existing systems by reusing them in teaching activities, such as Nachos, which comprises various modules implementing the functionalities of a basic operating system. These directions should be integrated when developing new QASs. With this motivation in mind, this work focuses on the ambiguity that is a crucial component in treating an NLQ in RDF KGs. This component is a prerequisite for achieving QAS accuracy. It should be noted that this quality is limited by the ability of NLQ processors to handle the ambiguity that such queries may contain [15,52]. Ambiguity occurs when a user's query contains words that have more than one meaning. For example, in the query "*Which books were written by Jack London?*", the entity "Jack London" must be linked

[1] https://www.slideshare.net/LuMa921/enterprise-knowledge-graph.

to the novelist "dbr:Jack_London"[2] rather than the boxer "dbr:Jack_London_(boxer)". The ambiguity can be classified into four main categories [8]: lexical, syntactic, semantic, and pragmatic.

A lexical ambiguity arises when a word has more than one generally accepted meaning: the cases of homonymy (words having the same forms, but no related meaning) or polysemy (words having the same forms, referring to different but related meanings), which is the best-known case. A syntactic ambiguity consists of an ambiguity arising from how the sentence is structured. Semantic ambiguities arise when a sentence has more than one interpretation, even if no lexical or syntactic ambiguity appears in the sentence. Pragmatic ambiguity appears when a sentence can have more than one meaning in the same context. This paper addresses the polysemy of named entities. Indeed, several KGs resources may have the same name. Therefore, polysemy arises when an entity in the NLQ can be linked to several entities in the KGs. The process of selecting the correct meaning refers to Entity Linking or Named Entity Disambiguation (NED). The latter consists of assigning named entities in a text document to entity identifiers in a KG. It generally consists of two phases [54]: Candidate generation and Disambiguation.

The first phase generates candidate resources to which the entity can refer, while the second aims to classify and filter candidates to select the best one for each entity detected. The NED task concerns both long and short texts. Several research efforts have focused on long texts [7,19,37,40,47,49,51,60]. To choose the most relevant sense of a word, we generally refer to its context, i.e. the words surrounding the ambiguous word. This textual context is exploitable to provide information about the ambiguous word and plays an important role in the disambiguation. Therefore, some approaches have analysed the entity's context to calculate the textual similarity score to remove ambiguity [40]. Other studies have measured the relationship between entities in the input text to link them collectively to the corresponding resources. Recently, short texts, and particularly queries in the QASs, have attracted attention because of their limited context [26,46]. However, because of the limited information they provide, textual similarity is not a sufficient solution. Moreover, short texts generally contain only one entity, which makes approaches exploiting entities impossible in this case. Given the limited context provided by short texts, are semantic and syntactic features extracted from this context sufficient for the entity disambiguation? What is the impact of each feature on the disambiguation process? What is the best combination of these features to reach the highest accuracy?

This paper extends our previous work [11] that focused only on the context of the named entity to link it to its corresponding resource of the KGs. First, the user query was expanded by retrieving synonyms from WordNet in order to address the context shortness problem. Then, the similarity between the context of the entity and each candidate's context is computed in order to select the best one. The exploitation of the context is not sufficient to reach high accuracy.

Consequently, in the present work, additional techniques are used to reinforce semantic aspects. We use relational information to better capture the semantic relation between the entity and the candidates. Two aspects are taken into account: (a) the coherence between entities and (b) the exploitation of relations. The distance between the name of the entity and that of a candidate is also considered as well as the use of syntactic features. In this paper, a complete system named Welink, for NED for short texts in general and QASs queries in particular, is built. Next, a score-based disambiguation algorithm is presented which is based on semantic and syntactic metrics to rank the candidates. WeLink is implemented

[2] PREFIX dbr: <http://dbpedia.org/resource/>.

using two different methods of entity recognition: lexical entity recognition and n-gram. Component-based architecture is adopted to ensure the flexibility of the system. Experiments are conducted on five well-known datasets to prove the effectiveness of WeLink. The results obtained are very encouraging. Finally, WeLink is available as an open REST API[3] and its source code is published on Github.[4]

This paper is organized as follows: Sect. 2 overviews and analyses existing studies using relevant criteria. Section 3 introduces the fundamental concepts including KGs, QASs, and NED. Section 4 presents WeLink, our proposed approach for dealing with the NED problem in short texts. Section 5 details the implementation aspects of WeLink. Section 6 presents our intensive experiments comparing WeLink and state-of-the-art systems. Section 7 shows our perspectives and concludes the paper.

2 Related work

A variety of NED approaches and systems have been proposed over the years [54]. Many existing approaches link named entities in long texts (documents, news corpuses, etc.) [2,7,19, 29,34,37,40,45,47,49,51,60]. These approaches generally remove ambiguity (i) by exploiting the text around the entity and calculating contextual similarity [40] and (ii) assuming that the input document refers to coherent entities so that these approaches observe all entities in the text and exploit this coherence to perform collective entity linking [60]. One of the best-known systems for long texts is DBpedia Spotlight [40]. This system identifies the entity using a list of surface forms and then generates candidates from DBpedia. The system then uses the surrounding context (paragraphs) to disambiguate the entity. To do this, Spotlight models a Vector Space Model (VSM) representation of the resource candidates with tfidf weights. It ranks them according to the similarity score (cosine) between their context vectors and the text surrounding the entity. This approach was later integrated into a QAS [18].

Recently, short texts and especially queries within QASs have attracted more attention because of their limited context [1,46]. To perform a NED in queries, EARL [17] defines the context of the entity by observing the relations around it. The system implements two different strategies to solve the NED task. First, the NED task is formalized as an instance of the Generalized Travelling Salesman Problem (GTSP). The latter is solved using the approximate GTSP solver Lin–Kernighan–Helsgaun (LKH). Second, it uses machine learning to exploit the connection density between nodes for disambiguation. TAGME [20] is a well-known NED system for short texts. After detecting the anchors in the input text, the system performs disambiguation using a voting scheme that calculates a score for each anchor-candidate match. It then prunes all candidate annotations to filter out the least relevant candidates. This approach is based on the relation between entities, but QAS queries usually contain only one entity. Therefore, this solution may not be sufficient. Machine learning approaches such as EARL and TAGME are based on training data. The majority of existing training data is suitable for long texts. Furthermore, the performance of these approaches decreases when the input text differs from the training domain. Falcon [50] jointly carried out the relation and linking of entities in QAS questions on DBpedia. For a given question, it identifies the entity and generates a list of candidates. The system then classifies these resources by creating a triple of candidate entities and relations and checks whether these triples exist in the KG. The

[3] http://193.194.84.136:8000/.

[4] https://github.com/wissembrdj/welink.

strength of this approach is also its weakness: if the triple does not exist, neither the entity nor the relation is linked.

Hence, none of the systems mentioned extend the context of the named entity. The proposed approach enriches the user's query to overcome the shortness problem and exploits the context similarity generally used in NED for long texts. In addition, it exploits both relations and entities in the NED task. It captures the correspondence between the extended words in the query and the properties of a resource. For queries containing more than one entity, it also takes into account the coherence between entities. In the example "In which region of the United States is Georgia?", the two entities "United States" and "Georgia" must be linked to the resources "dbr:United_States" and "dbr:Georgia_(U.S._state)", respectively. Similarly, the "region" relation can be linked to the "dbo:region" property. We assume that these three aspects (context, relations, and entities) must be used together to achieve high accuracy in the NED task. Moreover, this work emphasizes the importance of syntactic aspects by prioritizing words in capital letters and using entity length. Table 1 compares the main approaches available in the literature with our proposed approach.

3 Background

This section overviews the fundamental concepts related to KGs, QASs, and NED in short texts over a KG. The same example illustrates the definitions to facilitate their presentation and especially their interaction.

3.1 Knowledge graphs

Knowledge graphs have recently garnered significant attention from both industry and academia for capturing, representing, storing, and exploring structured knowledge. Several definitions of KGs have been proposed in the literature [31]. A recent consensual definition of a KG was proposed in [31] as follows:

Definition 1 A KG is viewed as a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent relations between these entities.

In general, the RDF model is used to represent these nodes and entities through its triples. A triple is the *smallest unit of data* in RDF. A triple models a single statement about resources with the following structure $\langle subject, predicate, object \rangle$. It indicates that a relationship identified by the *predicate* (also known as property) holds between the *subject* and the *object* depicting Web resources (things, documents, concepts, numbers, strings, etc.).

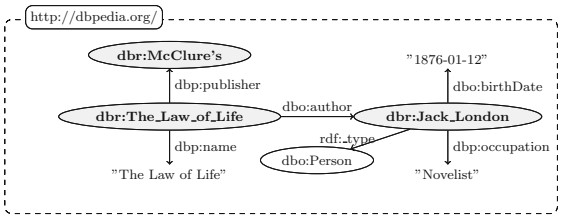
Example 1 The statement *"The author of The Law of Life is Jack London"* can be represented by a triple as $\langle The\ Law\ of\ Life, has\_author, Jack\ London \rangle$.[5] This triple can be represented logically as a graph where two nodes (subject and object) are joined with a directed arc (predicate) as shown in Fig. 1a.

To query KGs and RDF datasets, the W3C defined SPARQL [25] as the standard query language for RDF. SPARQL allows expressing queries across diverse datasets. The simpler SPARQL queries are formed as a conjunction of triple patterns (known as *Basic Graph*

[5] This triple is a part of DBpedia's Knowledge Graph.

Table 1 Comparison of the approaches available in the literature with the proposed approach

Systems	Semantic features			Syntactic features	Open source	Restful API
	Coherence	Relations exploitation	Context similarity			
EARL	×	×			×	×
DBpedia spotlight			×		×	×
TAGME	×	×				×
Falcon	×	×			×	×
WeLink	×	×	×	×	×	×



(a) RDF Graph

```
PREFIX dbr:<http://dbpedia.org/resource/>
PREFIX dbo:<http://dbpedia.org/ontology/>
PREFIX dbp:<http://dbpedia.org/property/>

SELECT ?x ?y WHERE {
dbr:The_Law_of_Life dbo:author ?x .
?x dbo:birthDate ?y
}
```

(b) SPARQL query

Fig. 1 Examples for RDF and SPARQL

Patterns BGP). Triple patterns are similar to RDF triples except that the subject, predicate, or object may be a variable.

Example 2 The query in Fig. 1b asks for the author of "The Law of Life" and his birthdate over *DBpedia KG*. The result of this query is a subgraph of the queried graph(s) in which the variable terms are mapped to the values of the resulting subgraph. Processing a SPARQL query can be viewed as a subgraph matching problem. The results of our query are the mappings of: $?x \rightarrow$ `<http://dbpedia.org/resource/Jack_London>` and $?y \rightarrow$ `"1876-01-12"`.

This variety and the wealth of KGs motivate researchers and industrials to develop intelligent services in several domains such as social networks [28], recommender systems [41], COVID-19 management [42], recruitment domain [27], and QAS [32]. In the next section, we introduce QASs and their functioning.

3.2 Question answering systems (QASs)

A QAS allows a user to ask a question q composed of a set of words $q = \{w_1, w_2, ..., w_{|q|}\}$ and provides a concise answer to textitq. A QAS aims at returning a specific answer to the user rather than a list of relevant documents. It formally transforms a question posed in an NLQ into a SPARQL query and extracts the answer by querying an information source, usually a KG. Each word in the user question w_i can correspond to a resource $w_i \in S$, or a property $w_i \in P$ or an object $w_i \in O$.

Example 3 Consider the question "*Which books were written by Jack London?*" over DBpedia KG. Listing 1 shows the SPARQL query that corresponds to this question.

Listing 1 SPARQL query example

```
PREFIX dbr : <http://dbpedia.org/resource/>
PREFIX dbo : <http://dbpedia.org/ontology/>
PREFIX rdf : <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT DISTINCT ?x WHERE {
    ?x dbo:author dbr:Jack_London .
    ?x rdf:type dbo:WrittenWork
}
```

Once the SPARQL query is executed over DBpedia, the QAS returns the books written by "Jack London" to the user, among others "The Law of Life".

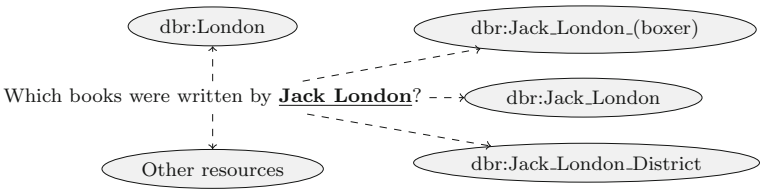


Fig. 2 An example of named entity ambiguity in a question

3.3 Named entity disambiguation

Named entities are defined as real-world objects that can be denoted by a proper name and associated with a type such as *Person*, *Organization*, *Place*. A *mention* (also called entity mention) is a span of text that refers to a named entity in a given text. A mention is often ambiguous because it can refer to different entities. The process of linking an entity mention to the corresponding KB resource is known as *Named Entity Disambiguation* or *Entity Linking*. Given a text consisting of a set of named entities $M = \{m_1, m_2, ...m_i\}$ and given a KG with a set of resources $R = \{r_1, r_2, ...r_i\}$, each entity $m_i \in M$ has to be linked to a resource $r_i \in R$.

Example 4 For our query "*Which books were written by Jack London?*" (cf. Fig. 2), the named entity $M = \{$"Jack London"$\}$ has to be correctly identified and linked to the KG resource "dbr:Jack_London" the novelist rather than the boxer "dbr:Jack_London_(boxer)".

In some cases, the input texts do not contain named entities. For example, "Who has produced the most films?". Thus, the NED system should not return any resource. In addition, the named entity may not have a corresponding KG resource. These cases are defined as unlinkable, and the system should return NIL [54].

4 The proposed approach

In this work, WeLink is proposed as an entity disambiguation approach. It consists of three modules (Fig. 3): *Query analysis*, *Candidate generation*, and *Disambiguation*. The *Query analysis* module makes the user's query exploitable. It also allows the extraction of the entity mention and its features. The *Candidate generation* module queries the KG to select candidates for an entity mention. The *Disambiguation* module uses a scoring algorithm to select the most relevant KG resource.

Example 5 Before detailing all the steps of our proposal, let us consider an illustration of each step through the previous example "*Which books were written by Jack London?*". This latter contains an entity mention that should be recognized and linked. WeLink works according to the following steps:

1. *Query analysis*: WeLink starts by pre-processing the input text, where the named entity "Jack London" is recognized. In addition, NLP tasks are applied to the query to make it exploitable for query expansion to generate the entity context (see Sect. 4.1).
2. *Candidate generation*: a set of candidates is generated using a SPARQL query over DBpedia exploiting the identified named entity "Jack London".
3. *Disambiguation*: a scoring algorithm is used to select the most relevant candidates. The algorithm assigns a weight according to semantic and syntactic features to rank the candidates. As a result, the resource "dbr:Jack_London" is selected.

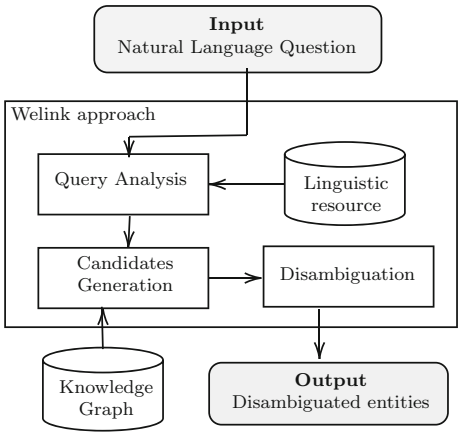


Fig. 3 The Welink approach

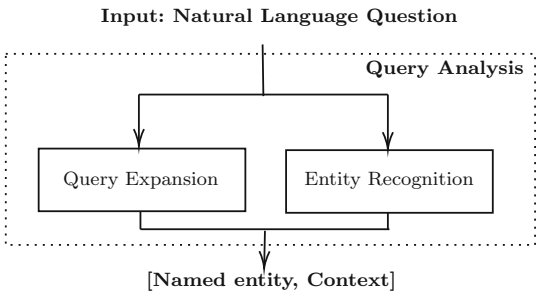


Fig. 4 Query analysis

4.1 Query analysis

Query analysis involves the use of NLP tasks to refine the input text and make it analysable. As shown in Fig. 4, the *Query analysis* phase is carried out by two separate tasks: Entity Recognition and Query Expansion.

4.1.1 Entity recognition

The objective of the named entity recognition (NER) task is to identify named entities in a given text. This important step influences the disambiguation process, i.e. if an entity is incorrectly detected, it is unlikely to be correctly linked. Moreover, if the entity is not detected, the whole process is stopped [35]. In this work, two different methods are used and analysed to identify the entities: the Lexical Entity Recognition (LER) and the n-gram. LER is mainly based on the lexical features of the input text. Thus, proper nouns are selected according to the part of speech of the words. However, this algorithm does not recognize words that are not in uppercase. N-gram is used to extract all possible tokens. It can detect entities in the input text, but generates several overlaps. The relevant entity can be detected, but isolated words (that make up the entity) are also detected and can be linked to KG resources. In the previous example, the entity "Jack London" is identified, but "Jack" and "London" are also recognized. In addition, meaningless word combinations are generated, which can take time. Therefore, to deal with this, we do not consider verbs as they cannot be named entities. Secondly, we give priority to the longest word combination. The aim is to prioritize the

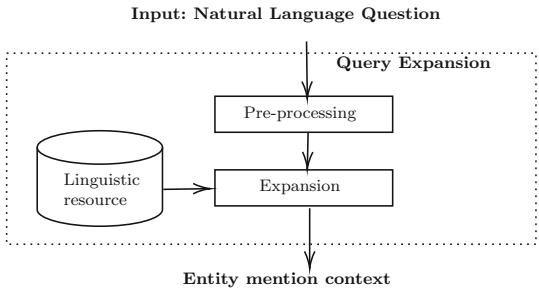


Fig. 5 Query expansion

longest token rather than the words that make it up. Thirdly, to not deprive single words, we assign a score to words in uppercase, as they usually refer to proper nouns.

4.1.2 Query expansion

Query expansion is a well-known technique for improving the efficiency of information retrieval. In general, queries are short and ambiguous. Expanding the user's query consists of adding similar terms to the initial text to retrieve more relevant information [4] and better represent the user's intention. Despite its advantages, the expansion has been widely used in information retrieval but rarely in QASs [30]. Moreover, if the terms used by the user do not match those used in KG, this leads to a lexical gap problem. Therefore, to expand the query and simultaneously reduce the lexical gap between the user's words and the resource label, we use WordNet [43] which has been widely used for query expansion [4]. To this end, the input text is first pre-processed and then expanded, as shown in Fig. 5.

- Pre-processing
  A pipeline process is used to execute the following NLP techniques: first, contractions and punctuation are removed; then, the query is tokenized into words; next, stop words are removed. A POS tagger is applied, and tags are filtered to keep only nouns, verbs, and adjectives to get their synonyms later.
- Expansion
  The obtained keywords are searched in WordNet for their synonyms. synonymy is considered to be the main relation between words in WordNet. Synonyms are grouped into Synsets with a brief description. We extract the synonyms of each word from the input text and the definition of the detected named entity to compose its context [11].

4.2 Candidate generation

Candidate generation consists of retrieving a set of KG resources that potentially match the entity mention. These candidates are then refined to retain the relevant resources during the Disambiguation (see Sect. 4.3). To accomplish this crucial step [24,54] and retrieve the relevant KG resources, we follow the steps used in [16] to have a rich treatment of entity name variations:

- The *exact match* between the mention and the resource title. The aim is to find a complete string match between the mention and the resource.
- The *partial match* between the mention and the resource title. The aim is to identify the mentions included in the resource title.

- *Acronyms* are used to retrieve resources that correspond to the first letters of a mention. For example, "US" refers to "United States".
- *Alternative names* allow the extraction of resources that have different names but refer to the same entity. It also includes synonyms, acronyms, and possible spelling mistakes.

In addition, we use entity types. In previous work, we already used entity types in two different ways:

– Exploiting types in the candidate generation phase by including the type in the SPARQL query [10]. This reduces ambiguity and, in some cases, removes it completely.
– The similarity between entity types and candidate types for filtering candidates [11], based on the assumption that an entity can be associated with several types. Thus, an entity may have multiple but related types. For example, London is a City, a Location, a Capital, etc.

In our analysis, we can note that the second method is ineffective due to the lexical gap between entity and resource types. Therefore, we use the first method, which consists of using types during the Candidate generation phase. We limit the entity types such as, but not limited to, Person, Organization and Place, to restrict the search space and thus limit the number of candidates.

4.3 Disambiguation

A score is assigned to each candidate to rank the generated candidates according to the following features: context similarity, coherence between entities, relations exploitation, entities name distance, and syntactic features. With the context similarity, we seek to calculate the semantic similarity of the context of the mention and the context of each candidate. Then, we use two semantic similarity scores: coherence between entities and exploitation of relations. Coherence between entities captures the semantic relation between entities and resources related to a candidate. In addition, the exploitation of relations measures the similarity between the words of the query and the properties of the candidate. Finally, we calculate the distance between the mention name and the candidate name and use syntactic features.

4.3.1 Context similarity

The similarity between the entity and candidate contexts is the most intuitive way to address the ambiguity problem. The context is defined as the textual information related to the entity mention (see Sect. 4.1.2) and the document associated with a candidate. The candidate context corresponds to the value of the "dbo:abstract" property, a short description extracted from the corresponding Wikipedia article. In this work, this abstract is considered the candidate context, which allows a rich contextual representation. For example, the candidate "dbr:London" has the following abstract considered to be its context: "London is the capital and most populous city of England and the United Kingdom".

The abstract's length varies from one resource to another, and it can be noticed that it is longer than the entity mention context, which is usually a single sentence. To provide a balance, we use only the first sentence of each abstract containing descriptive terms [12]. We use cosine similarity with a normalized version of tfidf (term frequency-inverse document frequency) to calculate the similarity between contexts, a weighting factor for ranking candidate entities. The tfidf weight for a term t in context c is the product of two factors: the

term frequency (tf) and the inverse document frequency (idf). As the contexts have different lengths, the measurements will have a high variance. Therefore, we use logarithmically scaled term frequency to normalize it [38]. The term frequency $tf_{(t,c)}$ is the occurrence of a t term in context c. The weighted term frequency is calculated as follows:

$$wf(t,c) = \begin{cases} log(1 + tf_{t,c}), & \text{if } tf_{t,c} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

The inverse document frequency (idf) illustrates the importance of the word in the collection, so it measures the weight of rare words in the documents. The idf weighs frequent terms and scales rare terms.

$$idf(t,C) = \begin{cases} log\frac{C}{df_t}, & \text{if } df_t > 0 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where, C is the total number of contexts and df_t is the number of contexts where the term t appears.

Consequently, tfidf weight value for a term t in a context d in a collection of contexts D is defined as:

$$tfidf_{t,c} = tf_{t,c} \times idf_{t,C} \tag{3}$$

We use a vector spatial model (VSM) to represent each context as a vector in multidimensional space [40]. Therefore, we derive a context vector weighted by the normalized tfidf. Then, we use cosine similarity to measure the angle θ between each candidate context vector $\overrightarrow{v}(c_c)$ and the vector of the entity mention context $\overrightarrow{v}(c_m)$ as follows:

$$cosine(c_c, c_m) = \frac{\overrightarrow{v}(c_c).\overrightarrow{v}(c_m)}{||\overrightarrow{v}(c_c)||\,||\overrightarrow{v}(c_m)||}. \tag{4}$$

4.3.2 Coherence between entities

When the entity "dbr:Barack_Obama" appears in a text, it is more likely that the mention "Michelle" represents his wife "dbr:Michelle_Obama" as the two entities are semantically related [36] and tend to occur more frequently together. The input text largely refers to coherent entities from a specific topic, and this coherence is exploitable for collectively disambiguating entities that appear in the same text. Therefore, for each entity mention, the entities in the same document are considered important for its disambiguation [54]. Different methods have been used to measure the entity coherence, one of which is Jaccard similarity [Eq. (5)] [22].

$$jaccard(a,b) = \frac{|a \cap b|}{|a \cup b|} \tag{5}$$

However, this study aims to determine the extent to which a candidate is similar to the entity mention, to capture the coherence between the entities in the query e_m and the resources related to a candidate e_c. Coherence is therefore measured as follows:

$$coh(e_m, e_c) = \frac{|e_m \cap e_c|}{|e_m|} \tag{6}$$

Due to the limited context provided by short texts, the measure will only be effective if the input text contains more than one named entity.

Table 2 Scores description used for candidates ranking

Score	Name	Description
$cosine(c_c, c_m)$	Context similarity	The context similarity score ranges between 0 and 1
$coh(e_m, e_c)$	Coherence between entities	The entities similarity score ranges between 0 and 1
$coh(\delta_m, \delta_c)$	Relations exploitation	The relations similarity score ranges between 0 and 1
$lev(name_m, name_c)$	Entities name distance	The name similarity score ranges between 0 and 1
$len(e)$	Entity mention length	Named entity mention length
$cl(e)$	Capital letter	1 if the mention's first letter is capitalized, 0 otherwise

4.3.3 Relations exploitation

Relations exploitation involves measuring the similarity between query terms and the properties of each candidate. In the example "Give me the birth place of Frank Sinatra", we seek to capture relatedness by exploiting the term "birth place" and the property "dbo:birthPlace". To capture this relatedness, we consider $\delta_m = rel(m_i, m_{i+1})$ the relation between two entities (i.e. terms that are not named entities) and $\delta_c = rel(c_i, r_i)$ the property that links a candidate c_i to another resource r_i. We exploit the same equation used previously [Eq. (6)] as follows:

$$coh(\delta_m, \delta_c) = \frac{|\delta_m \cap \delta_c|}{|\delta_m|}. \tag{7}$$

4.3.4 Name distance

With name distance, the aim is to capture the dissimilarity between the entity mention $name_m$ and the candidate label $name_c$. To achieve this, we use the Levenshtein distance to capture the difference between the two strings. The obtained distance is scaled by the length of the longest string [Eq. (8)]. The name distance score is subtracted from the total score to favour resources with similar labels to the named entity.

$$lev(name_m, name_c) = \frac{levenshtein(name_m, name_c)}{max(|name_m|, |name_c|)}. \tag{8}$$

4.3.5 Syntactic features

We assign a weight to the named entity mention to catch its syntactic feature. As mentioned in Sect. 4.1.1, we prioritize the longest sequences of words *len(e)* and capitalized words *cl(e)*. Finally, the overall disambiguation score α is assigned to each candidate [Eq. (9)]. Table 2 summarizes the features used to rank the entities.

$$\alpha = cosine(c_c, c_m) + coh(rel_c, rel_m) + coh(ec_c, ec_m) + len(e) + cl(e) - lev(e_c, e_m) \tag{9}$$

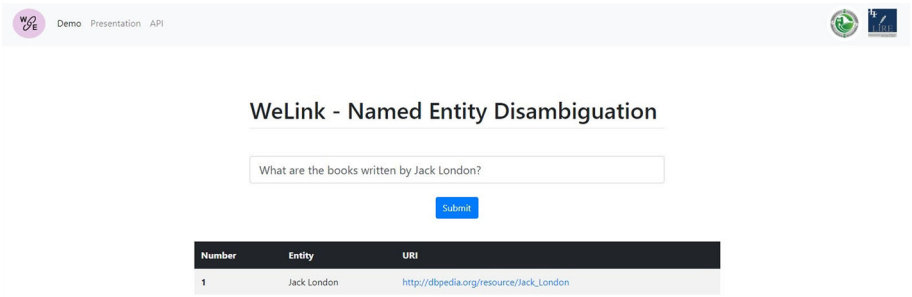


Fig. 6 WeLink web interface

Algorithm 1 presents the WeLink procedure for ranking candidates. After retrieving all the candidates, the disambiguation score α [see Eq. (9)] is computed for each one. Then, the candidates are ranked based on their α score. Finally, the candidate with the highest score is selected as the target resource.

Algorithm 1 Disambiguation Algorithm

Input : Q user query, $M = \{m_1, m_2, ...m_i\}$ named entities in Q, σ threshold
Output : D_e ranked candidates for each named entity
begin
  $C \leftarrow \emptyset$
  $D_e \leftarrow \emptyset$
  $candidates \leftarrow \emptyset$
  $candidates\_sim \leftarrow \emptyset$
  $for\ m_i \in M\ do:$
    $candidates \leftarrow StringMatch(ne)$; Retrieve all possible candidates for the named entity from the KG.
    $for\ c_i \in candidates\ do:$
      $total\_sim \leftarrow \alpha(m_i, c_i)$; Compute the total similarity score α for each candidate $c_i \in$ candidates.
      $if\ total\_sim \geq \sigma\ then$
        $candidates\_sim \leftarrow [c_i, total\_sim]$;
    $C \leftarrow sort(candidates\_sim)$; Sorting candidates according to descending total similarity score.
    $D_e.append([m_i, C])$;
  $return\ D_e$;
end.

5 Implementation

The WeLink system is a Python web application available online (Fig. 6) and is publicly accessible via a REST API.[6] The source code is published on Github.[7]

Figure 7 illustrates the Welink component diagram, including the three main components detailed above (*Query analysis*, *Candidate generation* and, *Disambiguation*). The objective

[6] http://193.194.84.136:8000/.

[7] https://github.com/wissembrdj/welink.

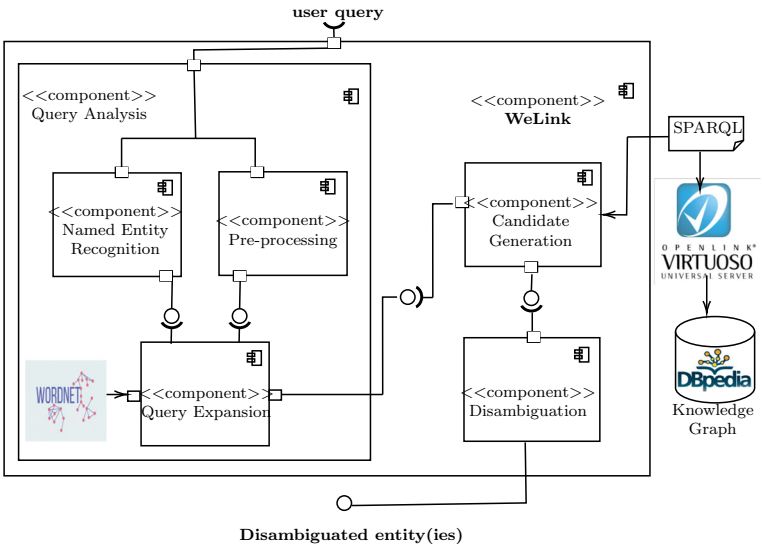


Fig. 7 Component diagram of WeLink

is to upgrade, maintain and improve each separate component without affecting the code of the overall system. This architecture ensures flexibility, allowing for easy adaptation [48] and integration of future components, thus allowing for the extension of the system. WeLink is deployed as a web service. Thus, regardless of the programming language and platform used, it can be integrated into any QAS or SOA (Service-Oriented Architecture) in general, increasing flexibility [13].

The Candidate generation component communicates with DBpedia via Virtuoso[8] to perform SPARQL queries on remote SPARQL Endpoint. In Listing 2, we detail the SPARQL query executed to retrieve candidates from DBpedia. In this query, the entity and its normalized form are used. To normalize an entity, the word's first letter is capitalized and an underscore symbol is added to link the words together (if the entity consists of several words). For example, the entity "jack london" is replaced by its normalized form "Jack_London". To manage the name variation of an entity (see Sect. 4.2), we make a union of four different properties (rdfs:label, dbo:wikiPageDisambigates, dbo:wikiPageRedirects, and foaf:name) to obtain resources that can refer to the entity. In addition, two filters are applied: first, the type filter using the property rdf:type to restrict the search space and limit the number of candidates. Second, the language filter restricts the labels of the exploited subjects, properties, and objects to English labels. Consequently, this SPARQL query returns the URIs of the retrieved resources (?y), as well as their labels (?na), abstracts (?s), properties (?props), and objects (?objs).

Listing 2 SPARQL query for candidate generation over DBpedia

```
PREFIX dbr : <http://dbpedia.org/resource/>
PREFIX rdf : <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs : <http://www.w3.org/2000/01/rdf-schema#>
```

[8] https://dbpedia.org/sparql.

```
PREFIX dbo : <http://dbpedia.org/ontology/>
PREFIX foaf : <http://xmlns.com/foaf/0.1/>

SELECT DISTINCT ?y ?s ?na
(concat(group_concat(distinct ?label; separator=' ')) as ?props)
(concat(group_concat(distinct ?olabel; separator=' ')) as ?objs)
WHERE {
        {dbr:normalized_ne dbo:wikiPageDisambiguates ?y }
        UNION
        {dbr:normalized_ne_(disambiguation) dbo:wikiPageDisambiguates ?y}
        UNION
        {?y rdfs:label "ne"@en }
        UNION
        {?y dbo:wikiPageRedirects dbr:normalized_ne }
        UNION
        {dbr:normalized_ne dbo:wikiPageRedirects ?y}
        UNION
        {?y foaf:name "ne"@en}
        ?y ?p ?o.
        ?y rdf:type ?types FILTER (contains(str(?types), "Place")
        || contains(str(?types), "Person")
        || contains(str(?types), "Organization") ).
        ?y dbo:abstract ?s FILTER (lang(?s) = "en").
        OPTIONAL
        {?p rdfs:label ?label FILTER (lang(?label) = "en").}
        OPTIONAL
        {?o rdfs:label ?olabel FILTER (lang(?olabel) = "en").}
        OPTIONAL
        {?y rdfs:label ?na FILTER (lang(?na) = "en").}
}
GROUP BY ?y ?s ?na
```

6 Experimentation

We compared the results of WeLink with those of Earl, TAGME, DBpedia Spotlight, and Falcon to evaluate the proposed approach and demonstrate its effectiveness.

6.1 Datasets

WeLink is evaluated on the following datasets that are publicly available:

– QALD-7 [59], QALD-8 [58], QALD-9 [57]: The Question Answering over Linked Data[9] challenge provides datasets containing multilingual questions to benchmark natural language processing for QASs and also information retrieval.

[9] http://qald.aksw.org.

- TREC 2014 Microblog: The Text REtrieval Conference (TREC) is a test collection that evaluates text retrieval. We use the TREC 2014 Microblog.[10] We manually annotated the search queries used to gather microblogs.
- ERD14 [6]: the Entity Recognition and Disambiguation Challenge (ERD'14) aims to promote the recognition and the disambiguation of named entities in unstructured texts. The delivered dataset (ERD14) contains web search queries and their annotation.

The particularity of these datasets is the shortness of their queries. Table 3 details the total number of queries per dataset, the number of queries that contain named entities, the total number of entities in each dataset, and the average length of queries (by word).

The experiments were conducted on a laptop machine running Windows 10 with an Intel Core i5-4300U vPro processor and 16 GB RAM.

6.2 Evaluation metrics

We report macro-Precision, macro-Recall, and macro-F-measure. We denote the entities in the query as $E = \{e_1, e_2, ..., e_n\}$ and the entities returned by the system as $\hat{E} = \{\hat{e}_1, \hat{e}_2, ...\hat{e}_n\}$.

$$P = \begin{cases} |E \cap \hat{E}|/|E|, & \text{if } |E| > 0 \\ 1\,, & \text{if E} = \varnothing \text{ and } \hat{E} = \varnothing \\ 0\,, & \text{if E} = \varnothing \text{ and } \hat{E} \neq \varnothing \end{cases}$$

$$R = \begin{cases} |E \cap \hat{E}|/|\hat{E}|, & \text{if } |\hat{E}| > 0 \\ 1\,, & \text{if E } = \varnothing \text{ and } \hat{E} = \varnothing \\ 0\,, & \text{if E } = \varnothing \text{ and } \hat{E} \neq \varnothing \end{cases}$$

Finally, the F-measure is computed based on precision and recall as follows:

$$F = \begin{cases} (2 \cdot P \cdot R)/(P + R), & \text{if } P \neq 0 and R \neq 0 \\ 0\,, & \text{Otherwise} \end{cases}$$

6.3 Evaluation results

Tables 4, 5 and 6 present the results of WeLink compared to related work systems on QALD-7, QALD-8, and QALD-9, respectively. Table 7 presents the results on TREC 2014 Microblog and ERD14.

WeLink outperforms related-work systems on QALD-7 (Test) and QALD-8 (Test) reaching an F-measure of 0.600 and 0.626, respectively. WeLink also outperforms related-work systems on the QALD-9 for both Train and Test datasets, where it achieves an F-measure of 0.729 for the Train dataset and 0.706 for the Test dataset. In addition, WeLink achieves higher recall and precision on the QALD-9 (Train and Test). On the TREC 2014 Microblog and ERD14 datasets, WeLink achieves a higher F-measure than the related-work systems by achieving an average improvement of 27% on the TREC 2014 Microblog and 47% on the ERD14. The WeLink results obtained are with two different methods of entity recognition: LER and n-gram. Although the results are consistently better with n-gram than with LER, the latter method also gives good results. WeLink with LER gives a better F-measure on QALD-9 (Train and Test) than related-work systems.

[10] https://trec.nist.gov/data/microblog2014.html.

Table 3 Dataset details, including the number of questions, questions containing NE, number of named entities per dataset

Dataset		Questions number	Questions containing NEs	NEs number	Average question length
QALD-7	Train	215	202	242	8.38
	Test	43	31	34	8.46
QALD-8	Train	219	181	210	8.33
	Test	41	38	38	9.41
QALD-9	Train	408	328	396	8.49
	Test	150	125	151	8.57
TREC 2014 Microblog		55	52	63	4.10
ERD14		91	44	56	3.49

Table 4 Evaluation of WeLink against EARL, TAGME, DBpedia Spotlight, and Falcon on QALD-7

Approach	QALD-7 train			QALD-7 test		
	P	R	F	P	R	F
EARL	0.578	0.606	0.583	0.359	0.421	0.368
TAGME	0.600	0.715	0.631	0.468	0.581	0.503
DBpedia spotlight	0.676	0.71	0.676	0.546	0.569	0.550
Falcon	**0.812**	**0.836**	**0.812**	0.437	0.488	0.453
WeLink + LER	0.739	0.723	0.725	0.558	0.546	0.550
WeLink + n-gram	0.774	0.779	0.768	**0.593**	**0.616**	**0.600**

The best values are indicated in bold

Table 5 Evaluation of WeLink against EARL, TAGME, DBpedia Spotlight, and Falcon on QALD-8

Approach	QALD-8 train			QALD-8 Test		
	P	R	F	P	R	F
EARL	0.552	0.597	0.565	0.354	0.390	0.366
TAGME	0.603	0.731	0.641	0.522	**0.683**	0.566
DBpedia spotlight	0.649	0.694	0.657	0.556	0.609	0.573
Falcon	0.758	**0.797**	**0.769**	0.524	0.561	0.537
WeLink + LER	0.719	0.710	0.710	0.475	0.487	0.479
WeLink + n-gram	**0.761**	0.772	0.760	**0.622**	0.634	**0.626**

The best values are indicated in bold

Table 6 Evaluation of WeLink against EARL, TAGME, DBpedia Spotlight, and Falcon on QALD-9

Approach	QALD-9 train			QALD-9 test		
	P	R	F	P	R	F
EARL	0.534	0.556	0.530	0.447	0.450	0.432
TAGME	0.533	0.665	0.568	0.457	0.578	0.486
DBpedia spotlight	0.69	0.707	0.684	0.604	0.619	0.594
Falcon	0.683	0.696	0.675	0.626	0.637	0.600
WeLink + LER	0.702	0.691	0.692	0.682	0.692	0.681
WeLink + n-gram	**0.736**	**0.736**	**0.729**	**0.706**	**0.719**	**0.706**

The best values are indicated in bold

We observe that Falcon performs better on QALD-7 (Train) and QALD-8 (Train). However, WeLink performs quite comparably (-5% on QALD-7 (Train) and -1% on QALD-8 (Train)). First, one possible reason is that the entities in these datasets are highly ambiguous. For example, in the query "Which types of grapes grow in Oregon?", our system returns the resource dbr:Oregon, while the dataset is annotated differently (dbr:Oregon_wine). Another example is the query "Who assassinated President McKinley?", where WeLink returns the resource dbr:William_McKinley. Nevertheless, the dataset is annotated with dbc:Assassination_of_William_McKinley. The system results can be considered correct since the disambiguation task's goal is to return the corresponding resource of an entity. However, the datasets are annotated with resources to answer the questions and are therefore better suited for QASs evaluation. We also noticed that a SPARQL query can be formulated

Table 7 Evaluation of WeLink against EARL, TAGME, DBpedia Spotlight, and Falcon on TREC 2014 Microblog and ERD14

Approach	TREC microblog 2014			ERD14		
	P	R	F	P	R	F
EARL	0.287	0.363	0.310	0.181	0.181	0.177
TAGME	0.500	0.600	0.526	0.320	0.353	0.318
DBpedia spotlight	0.542	**0.604**	0.557	0.417	0.423	0.417
Falcon	0.366	0.368	0.335	0.181	0.175	0.175
WeLink + LER	0.454	0.427	0.436	0.516	0.516	0.516
WeLink + n-gram	**0.618**	0.600	**0.600**	**0.538**	**0.514**	**0.521**

The best values are indicated in bold

differently, for example, "What other books have been written by the author of The Fault in Our Stars?". This question is annotated in the QALD-8 dataset with the following SPARQL query "SELECT ?books WHERE { ?books dbo:author dbr:John_Green_(author) }". This query contains the name of the author (John Green) which does not appear in the question and therefore cannot be exploited. This SPARQL query can be written differently: "SELECT ?books WHERE { dbr:The_Fault_in_Our_Stars dbo:author ?y. ?books dbp:author ?y }". We judge that the latter query is more adequate because it contains the entity appearing in the question (dbr:The_Fault_in_Our_Stars). Based on that, we can affirm that the annotation of the datasets influences the evaluation of the NED systems.

Second, our approach prioritizes the exact match between the entity mention and the resource by using syntactic features. This is practical in many cases, but disadvantages the right resource in other cases. In the previous example with the mention "Oregon" the "dbr:Oregon_wine", resource is disadvantaged because "dbr:Oregon" is an exact match. We will focus on these cases in future work.

Third, the entity types used in the candidate generation step are limited and may be enriched. Therefore, if a resource type is not in the list of types mentioned in the SPARQL query, the entity may not be correctly retrieved. This technique can be automatized by integrating the recognized type (retrieved during entity recognition) into the SPARQL query. Since entity recognition is not our main focus in this paper, we will focus on this step in future work to overcome this limit.

We also observe that Tagme (QALD-8 train dataset) and DBpedia Spotlight(TREC microblog 2014) have higher recall but lower precision and F-measure. These systems return numerous entities, out of which many are irrelevant. In our case, the precision and recall of WeLink are balanced, because it returns few entities that are correctly linked in most cases. Furthermore, even if WeLink has slightly poorer results on the QALD-7 train and QALD-8 train, we judge that the overall improvement on all datasets is more important (see Table 8). We consider the F-measure of WeLink stable (on average 0.663), compared to other approaches that have good results on some datasets, but their performance drops in others. As shown in Table 8, the average percentage improvement in the related-work F-measure is 27% across all datasets. It should be noted that the improvement is up to 71%. The results indicate that WeLink successfully tackles the ambiguity problem in short texts.

Figure 8 shows the impact of each metric separately on the F-measure over the QALD datasets (Train + Test). The WeLink results with the disambiguation score α are also illustrated. We observe that the exploitation of relations has a slightly higher impact on the disambiguation process than the other metrics. In addition, the distance between names has

Table 8 The improvement percentage of WeLink F-measure compared to the related-work systems

Approach	QALD-7		QALD-8		QALD-9		TREC 2014 Microblog (%)	ERD14 (%)
	Train (%)	Test (%)	Train (%)	Test (%)	Train (%)	Test (%)		
EARL	**+32**	**+63**	**+35**	**+71**	**+38**	**+63**	**+48**	**+66**
TAGME	**+22**	**+19**	**+19**	**+11**	**+28**	**+45**	**+12**	**+38**
DBpedia spotlight	**+14**	**+9**	**+16**	**+9**	**+7**	**+19**	**+7**	**+19**
Falcon	−5	**+32**	−1	**+17**	**+8**	**+18**	**+44**	**+66**

Bold indicates the improvement

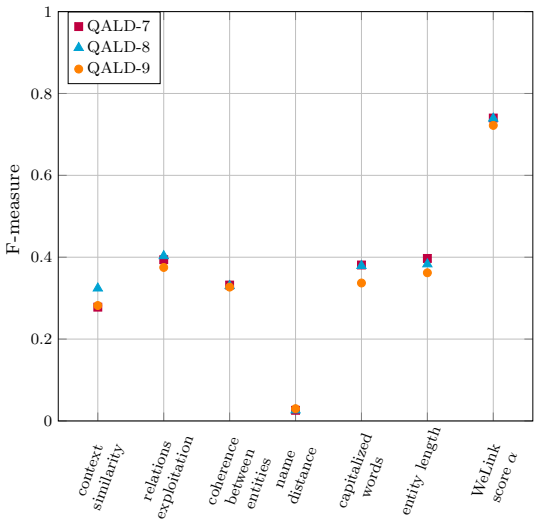


Fig. 8 The impact of each metric on the F-measure over QALD-7, QALD-8, and QALD-9

a small impact on F-measure because it is not intended to be used alone but rather with the overall score to deprive candidates with very different names of the entity.

On average, queries that do not contain any entities account for 20% (on average) of the datasets used. The NIL threshold on the total similarity score is empirically set at 0.6. As a result, WeLink reacts correctly to these queries in 56% of the cases, which increases the F-measure by 10%.

6.4 Discussion

Analysis of the results shows that WeLink successfully addresses the problem of NED in short texts. High performance is obtained experimentally with the proposed algorithm that exploits features that capture the semantics of an ambiguous entity. Therefore, we consider that WeLink overcomes the context shortness problem. After analysing the metrics' impact used in the proposed disambiguation algorithm, we assert that, these metrics do not reach a high F-measure separately. However, combined, they give good results. Furthermore, it can also be observed that the exploitation of relations has more impact on the F-measure than coherence between entities. This can be explained by the fact that queries usually contain one named entity.

Using two different approaches in the entity recognition step, it can be said that the n-gram approach generally identifies all named entities in the input text. As a result, these entities are more likely to be correctly linked by the proposed disambiguation process. On the other hand, it has a higher execution time than the lexical entity recognition (LER) method. Moreover, the LER certainly provides a considerable F-measure, but it fails, in some cases, to identify the entity, therefore stopping the disambiguation process. Thus, entity recognition is a crucial step for the NED task.

However, WeLink has certain limitations; one of the failure cases is spelling mistakes such as "Cheryl Teigs" where the corresponding resource is "dbr:Cheryl_Tiegs". WeLink also fails when the words in the question differ from the resource name, for example, the word "oscar", a benchmark for "dbr:Academy_Award".

7 Conclusion

This paper addressed Named Entity Disambiguation (NED) in a short text in general and in Question Answering Systems (QASs) in particular. NED is one of the essential components in the development of open and modular QASs. The proposed approach combines semantic and syntactic features to overcome the context shortness. We designed WeLink, a Named Entity Disambiguation system for short texts, based on (i) exploiting context similarity by expanding the entity context using WordNet, (ii) using entities for queries that contain more than one entity, (iii) exploiting relations by comparing relations between entities to candidate properties, (iv) the distance between the entity name and the resource name, and (v) the use of syntactic features. One of the most important characteristics of the proposed method is its ability to be deployed for open QASs.

Experiments were conducted on five datasets: QALD-7, QALD-8, QALD-9, TREC 2014 Microblog, and ERD14. WeLink outperforms the related-work system and increases the F-measure by an average of 27%. In addition, we detailed the impact of each metric on the disambiguation process and concluded that relations exploitation has more impact on the F-measure than coherence between entities. We believe that our proposal is complete and its code is available at: https://github.com/wissembrdj/welink. Also, it is accessible via a REST API.

Currently, we are working on improving the entity recognition stage, which impacts both the effectiveness and efficiency of NED. We are also focusing on the problems we identified during the validation of Welink, such as spelling mistakes. Finally, we would like to investigate multilingual entity linking and explore more Knowledge Graphs.

References

1. Al-Moslmi T, Ocaña MG, Opdahl AL, Veres C (2020) Named entity extraction for knowledge graphs: a literature overview. IEEE Access 8:32862–32881. https://doi.org/10.1109/ACCESS.2020.2973928
2. Alokaili A, Menai MEB (2020) SVM ensembles for named entity disambiguation. Computing 102(4):1051–1076. https://doi.org/10.1007/s00607-019-00748-x
3. Auer S, Bizer C, Kobilarov G, Lehmann J, Cyganiak R, Ives Z (2007) Dbpedia: a nucleus for a web of open data. In: The semantic web. Springer, pp 722–735. https://doi.org/10.1007/978-3-540-76298-0_52
4. Azad HK, Deepak A (2019) A new approach for query expansion using Wikipedia and wordnet. Inf Sci 492:147–163. https://doi.org/10.1016/j.ins.2019.04.019
5. Bellomarini L, Gottlob G, Pieris A, Sallinger E (2018) Swift logic for big data and knowledge graphs. In: International conference on current trends in theory and practice of informatics. Springer, pp 3–16. https://doi.org/10.1007/978-3-319-73117-9_1
6. Bennett PN, Gabrilovich E, Kamps J, Karlgren J (2014) Report on the sixth workshop on exploiting semantic annotations in information retrieval (esair'13). SIGIR Forum 48(1):13–20. https://doi.org/10.1145/2641383.2641387
7. Bentounsi I, Boufaïda Z (2015) Disambiguation of semantic types in complex noun phrases for extracting candidate terms. Int J Metadata, Semant Ontol 10(2):112–122. https://doi.org/10.1504/IJMSO.2015.070830
8. Berry DM (2007) Ambiguity in natural language requirements documents. In: Monterey workshop. Springer, pp 1–7. https://doi.org/10.1007/978-3-540-89778-1_1
9. Bollacker KD, Evans C, Paritosh P, Sturge T, Taylor J (2008) Freebase: a collaboratively created graph database for structuring human knowledge. In: ACM SIGMOD, pp 1247–1250. https://doi.org/10.1145/1376616.1376746
10. Bouarroudj W, Boufaida Z (2018) A candidate generation algorithm for named entities disambiguation using dbpedia. In: World conference on information systems and technologies. Springer, pp 712–721. https://doi.org/10.1007/978-3-319-77703-0_71

11. Bouarroudj W, Boufaida Z, Bellatreche L (2019) Welink: a named entity disambiguation approach for a qas over knowledge bases. In: International conference on flexible query answering systems. Springer, pp 85–97. https://doi.org/10.1007/978-3-030-27629-4_11
12. Burel G, Saif H, Alani H (2017) Semantic wide and deep learning for detecting crisis-information categories on social media. In: International semantic web conference. Springer, pp 138–155. https://doi.org/10.1007/978-3-319-68288-4_9
13. Dai W, Vyatkin V, Christensen JH, Dubinin VN (2015) Bridging service-oriented architecture and IEC 61499 for flexibility and interoperability. IEEE Trans Ind Inf 11(3):771–781. https://doi.org/10.1109/TII.2015.2423495
14. Diefenbach D, Giménez-García J, Both A, Singh K, Maret P (2020) Qanswer kg: Designing a portable question answering system over RDF data. In: European semantic web conference. Springer, pp 429–445. https://doi.org/10.1007/978-3-030-49461-2_25
15. Diefenbach D, Lopez V, Singh K, Maret P (2018) Core techniques of question answering systems over knowledge bases: a survey. Knowl Inf Syst 55(3):529–569. https://doi.org/10.1007/s10115-017-1100-y
16. Dredze M, McNamee P, Rao D, Gerber A, Finin T (2010) Entity disambiguation for knowledge base population. In: Proceedings of the 23rd international conference on computational linguistics. Association for Computational Linguistics, pp 277–285
17. Dubey M, Banerjee D, Chaudhuri D, Lehmann J (2018) Earl: joint entity and relation linking for question answering over knowledge graphs. In: International semantic web conference. Springer, pp 108–126. https://doi.org/10.1007/978-3-030-00671-6_7
18. Dubey M, Dasgupta S, Sharma A, Höffner K, Lehmann J (2016) Asknow: a framework for natural language query formalization in Sparql. In: European semantic web conference. Springer, pp 300–316. https://doi.org/10.1007/978-3-319-34129-3_19
19. Fang Z, Cao Y, Li Q, Zhang D, Zhang Z, Liu Y (2019) Joint entity linking with deep reinforcement learning. In: The world wide web conference, pp 438–447. https://doi.org/10.1145/3308558.3313517
20. Ferragina P, Scaiella U (2011) Fast and accurate annotation of short texts with Wikipedia pages. IEEE Softw 29(1):70–75. https://doi.org/10.1109/MS.2011.122
21. Green Jr BF, Wolf AK, Chomsky C, Laughery K (1961) Baseball: an automatic question-answerer. In: Papers presented at the May 9–11, 1961, western joint IRE-AIEE-ACM computer conference, pp 219–224. https://doi.org/10.1145/1460690.1460714
22. Guo S, Chang MW, Kiciman E (2013) 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
23. Gutierrez C, Sequeda JF (2021) Knowledge graphs. Commun ACM 64(3):96–104. https://doi.org/10.1145/3418294
24. Hachey B, Radford W, Nothman J, Honnibal M, Curran JR (2013) Evaluating entity linking with Wikipedia. Artif Intell 194:130–150. https://doi.org/10.1016/j.artint.2012.04.005
25. Harris S, Seaborne A (2013) SPARQL 1.1 overview. https://www.w3.org/TR/sparql11-overview/
26. Hasibi F, Balog K, Bratsberg SE (2017) Entity linking in queries: efficiency vs. effectiveness. In: European conference on information retrieval. Springer, pp 40–53. https://doi.org/10.1007/978-3-319-56608-5_4
27. He Q (2021) People of ACM. https://www.acm.org/articles/people-of-acm/2021/qi-he?fbclid=IwAR0wMvkvDL5k_iuShKqGluAG5z9_t5dsw-bnnwTMVf2bE03KN4sBHuwXBwc
28. He Q, Yang J, Shi B (2020) Constructing knowledge graph for social networks in A deep and holistic way. In: Companion of the web conference, pp 307–308. https://doi.org/10.1145/3366424.3383112
29. Hoffart J, Yosef MA, Bordino I, Fürstenau H, Pinkal M, Spaniol M, Taneva B, Thater S, Weikum G (2011) Robust disambiguation of named entities in text. In: Proceedings of the 2011 conference on empirical methods in natural language processing, pp 782–792
30. Höffner K, Walter S, Marx E, Usbeck R, Lehmann J, Ngonga Ngomo AC (2017) Survey on challenges of question answering in the semantic web. Semant Web 8(6):895–920. https://doi.org/10.3233/SW-160247
31. Hogan A, Blomqvist E, Cochez M, d'Amato C, de Melo G, Gutierrez C, Gayo JEL, Kirrane S, Neumaier S, Polleres A, Navigli R, Ngomo ACN, Rashid SM, Rula A, Schmelzeisen L, Sequeda J, Staab S, Zimmermann A (2021) Knowledge graphs. Synth Lect Data Semantics Knowledge 12(2):1–257
32. Hu X, Duan J, Dang D (2021) Natural language question answering over knowledge graph: the marriage of sparql query and keyword search. Knowl Inf Syst. https://doi.org/10.1007/s10115-020-01534-4
33. Huang X, Zhang J, Li D, Li P (2019) Knowledge graph embedding based question answering. In: Proceedings of the 12th ACM international conference on web search and data mining, pp 105–113. https://doi.org/10.1145/3289600.3290956
34. Klie JC, de Castilho RE, Gurevych I (2020) From zero to hero: human-in-the-loop entity linking in low resource domains. In: Proceedings of the 58th annual meeting of the association for computational linguistics, pp 6982–6993. https://doi.org/10.18653/v1/2020.acl-main.624

35. Kolitsas N, Ganea OE, Hofmann T (2018) End-to-end neural entity linking. ArXiv preprint arXiv:1808.07699. https://doi.org/10.18653/v1/K18-1050
36. Ling X, Singh S, Weld DS (2015) Design challenges for entity linking. Trans Assoc Comput Linguist 3:315–328. https://doi.org/10.1162/tacl_a_00141
37. Logeswaran L, Chang MW, Lee K, Toutanova K, Devlin J, Lee H (2019) Zero-shot entity linking by reading entity descriptions. ArXiv preprint arXiv:1906.07348
38. Manning CD, Raghavan P, Schütze H (2008) Scoring, term weighting and the vector space model. Introd Inf Retr 100:2–4. https://doi.org/10.1017/CBO9780511809071.007
39. Marginean A (2017) Question answering over biomedical linked data with grammatical framework. Semant Web 8(4):565–580. https://doi.org/10.3233/SW-160223
40. Mendes PN, Jakob M, García-Silva A, Bizer C (2011) Dbpedia spotlight: shedding light on the web of documents. In: Proceedings of the 7th international conference on semantic systems. ACM, pp 1–8. https://doi.org/10.1145/2063518.2063519
41. Mezni H, Benslimane D, Bellatreche L (2021) Context-aware service recommendation based on knowledge graph embedding. IEEE Trans Knowl Data Eng. https://doi.org/10.1109/TKDE.2021.3059506
42. Michel F, Gandon F, Ah-Kane V, Bobasheva A, Cabrio E, Corby O, Gazzotti R, Giboin A, Marro S, Mayer T, et al. (2020) Covid-on-the-web: knowledge graph and services to advance covid-19 research. In: International semantic web conference. Springer, pp 294–310. https://doi.org/10.1007/978-3-030-62466-8_19
43. Miller GA (1995) Wordnet: a lexical database for English. Commun ACM 38(11):39–41. https://doi.org/10.1145/219717.219748
44. Mishra A, Jain SK (2016) A survey on question answering systems with classification. J King Saud Univer-Comput Inf Sci 28(3):345–361. https://doi.org/10.1016/j.jksuci.2014.10.007
45. Moro A, Raganato A, Navigli R (2014) Entity linking meets word sense disambiguation: a unified approach. Trans Assoc Comput Linguist 2:231–244. https://doi.org/10.1162/tacl_a_00179
46. Oliveira IL, Fileto R, Speck R, Garcia LP, Moussallem D, Lehmann J (2021) Towards holistic entity linking: survey and directions. Inf Syst 95:101624. https://doi.org/10.1016/j.is.2020.101624
47. Parravicini A, Patra R, Bartolini DB, Santambrogio MD (2019) Fast and accurate entity linking via graph embedding. In: Proceedings of the 2nd joint international workshop on graph data management experiences and systems (GRADES) and network data analytics (NDA), pp 1–9. https://doi.org/10.1145/3327964.3328499
48. Pohl K, Böckle G, van Der Linden FJ (2005) Software product line engineering: foundations, principles and techniques. Springer
49. Rama-Maneiro E, Vidal JC, Lama M (2020) Collective disambiguation in entity linking based on topic coherence in semantic graphs. Knowl-Based Syst. https://doi.org/10.1016/j.knosys.2020.105967
50. Sakor A, Mulang IO, Singh K, Shekarpour S, Vidal ME, Lehmann J, Auer S (2019) Old is gold: linguistic driven approach for entity and relation linking of short text. In: Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, vol 1, pp 2336–2346. https://doi.org/10.18653/v1/N19-1243
51. Sevgili Ö, Panchenko A, Biemann C (2019) Improving neural entity disambiguation with graph embeddings. In: Proceedings of the 57th annual meeting of the association for computational linguistics: student research workshop, pp 315–322. https://doi.org/10.18653/v1/P19-2044
52. Sevgili O, Shelmanov A, Arkhipov M, Panchenko A, Biemann C (2020) Neural entity linking: a survey of models based on deep learning. ArXiv preprint arXiv:2006.00575
53. Shekarpour S, Marx E, Ngonga Ngomo AC, Auer S (2014) Sina: semantic interpretation of user queries for question answering on interlinked data. Web Semant: Sci, Serv Agents World Wide Web. https://doi.org/10.1016/j.websem.2014.06.002
54. Shen W, Wang J, Han J (2015) Entity linking with a knowledge base: issues, techniques, and solutions. IEEE Trans Knowl Data Eng 27(2):443–460. https://doi.org/10.1109/TKDE.2014.2327028
55. Singh K, Radhakrishna AS, Both A, Shekarpour S, Lytra I, Usbeck R, Vyas A, Khikmatullaev A, Punjani D, Lange C, et al. (2018) Why reinvent the wheel: let's build question answering systems together. In: Proceedings of the 2018 world wide web conference, pp 1247–1256. https://doi.org/10.1145/3178876.3186023
56. Singhal A (2012) Introducing the knowledge graph: things, not strings. Off Google Blog 5:16. https://blog.google/products/search/introducingknowledge-graph-things-not
57. Usbeck R, Gusmita RH, Ngomo AN, Saleem M (2018) 9th challenge on question answering over linked data (QALD-9). In: SemDeep-4 and NLIWOD-4 and ISWC vol 2241, pp 58-64. CEUR-WS.org
58. Usbeck R, Ngomo ACN, Conrads F, Röder M, Napolitano G (2018) 8th challenge on question answering over linked data (qald-8). Language 7:1

59. Usbeck R, Ngomo ACN, Haarmann B, Krithara A, Röder M, Napolitano G (2017) 7th open challenge on question answering over linked data (qald-7). In: Semantic web evaluation challenge. Springer, pp 59–69. https://doi.org/10.1007/978-3-319-69146-6_6
60. Usbeck R, Ngomo ACN, Röder M, Gerber D, Coelho SA, Auer S, Both A (2014) Agdistis-graph-based disambiguation of named entities usinglinked data. In: International semantic web conference. Springer, pp 457–471. https://doi.org/10.1007/978-3-319-11964-9_29

Wissem Bouarroudj is a Ph.D. student at LIRE Laboratory. She received her master's degree from the Faculty of Information and Communication Technology at Constantine 2 Abdehamid Mehri university, Constantine, Algeria. Her research interests include natural language processing, semantic web and entity disambiguation.

Zizette Boufaida is a Full Professor of computer science at Constantine 2 Abdehamid Mehri university, Constantine, Algeria and co-head of the SIBC research group at LIRE Laboratory. She served as an associate professor at Paris 5 university, France from 1991 to 1996. Her research interests include knowledge representation and knowledge base design, formal knowledge representation models and query languages for the semantic web.

Ladjel Bellatreche is a Full Professor at National Engineering School for Mechanics and Aerotechnics (ISAE-ENSMA), Poitiers, France. He leads the Data and Model Engineering Team of the Laboratory of Computer Science and Automatic Control for Systems (LIAS). He is also part time Professor at Harbin Institute of Technology—HIT, Harbin, China since 2019. His was a visiting Professor at the Québec en Outaouais—Canada (2009), a visiting researcher at Purdue University—USA (2001) and at Hong Kong University of Science and Technology, China (1997–1999). His research interest focuses on Data Management Systems and Semantic Web. He serves as an Associate Editor of the Data and Knowledge (DKE) Journal, Elsevier.