Entity Linking in Queries: Efficiency vs. Effectiveness

Faegheh Hasibi[1(✉)], Krisztian Balog[2], and Svein Erik Bratsberg[1]

[1] Norwegian University of Science and Technology, Trondheim, Norway
{faegheh.hasibi,sveinbra}@idi.ntnu.no
[2] University of Stavanger, Stavanger, Norway
krisztian.balog@uis.no

Abstract. Identifying and disambiguating entity references in queries is one of the core enabling components for semantic search. While there is a large body of work on entity linking in documents, entity linking in queries poses new challenges due to the limited context the query provides coupled with the efficiency requirements of an online setting. Our goal is to gain a deeper understanding of how to approach entity linking in queries, with a special focus on how to strike a balance between effectiveness and efficiency. We divide the task of entity linking in queries to two main steps: candidate entity ranking and disambiguation, and explore both unsupervised and supervised alternatives for each step. Our main finding is that best overall performance (in terms of efficiency and effectiveness) can be achieved by employing supervised learning for the entity ranking step, while tackling disambiguation with a simple unsupervised algorithm. Using the Entity Recognition and Disambiguation Challenge platform, we further demonstrate that our recommended method achieves state-of-the-art performance.

1 Introduction

The aim of semantic search is to deliver more relevant and focused responses, and in general an improved user experience, by understanding the searcher's intent and context behind the query provided. Identifying entity mentions in text and subsequently linking them to the corresponding entries in a reference knowledge base (KB) is known as the task of *entity linking*. It can be performed on long texts (i.e., documents), or very short texts such as web search queries; the latter is referred to as *entity linking in queries* (ELQ). It has been shown that leveraging entity annotations of queries is beneficial for various information retrieval tasks including document retrieval [8,31], entity retrieval [17,28], and task understanding [32].

Entity linking has been extensively studied for long texts [7,14,15,21,24,25]. Despite the large variety of approaches, there are two main components that are present in all entity linking systems: (i) *candidate entity ranking*, i.e., identifying entities that can be possibly linked to a mention, and (ii) *disambiguation*, i.e., selecting the best entity (or none) for each detected mention. There is also a

J.M. Jose et al. (Eds.): ECIR 2017, LNCS 10193, pp. 40–53, 2017.
DOI: 10.1007/978-3-319-56608-5_4

general consensus on the two main categories of features that are needed for effective entity linking: (i) contextual similarity between a candidate entity and the surrounding text of the entity mention, and (ii) interdependence between all entity linking decisions in the text (extracted from the underlying KB). Previous studies [4,14] have investigated these aspects in a unified framework and derived general recommendations for entity linking in documents. Entity linking in queries, however, has only recently started to draw attention [3,6,16] and such systematic evaluation of the different components has not been conducted until now. With this study, we aim to fill that gap.

What is special about entity linking in queries? First, queries are short, noisy text fragments where the ambiguity of a mention may not be resolved because of the limited context. That is, a mention can possibly be linked to more than one entity (see Table 1 for examples). This is unlike entity linking in documents, where it is assumed that there is enough context for disambiguation. Second, ELQ is an online process that happens during query-time, meaning that it should be performed under serious time constraints (in contrast with traditional entity linking which is offline). The ideal solution is not necessarily the most effective one, but the one that represents the best trade-off between effectiveness and efficiency. Therefore, the same techniques that have been used for entity linking in documents may not be suitable for queries. We formulate the following two research questions:

- **RQ1.** Given the response time requirements of an online setting, what is the relative importance of candidate entity ranking vs. disambiguation? In other words, if we are to allocate the available processing time between the two, which one would yield the highest gain?
- **RQ2.** Given the limited context provided by queries, which group of features is needed the most for effective entity disambiguation: contextual similarity, interdependence between entities, or both?

To answer the above research questions, we set up a framework where different candidate entity ranking and disambiguation methods can be plugged in. For each of these components, we experiment with both unsupervised and supervised alternatives, resulting in a total of four different ELQ systems. Our candidate entity ranking and disambiguation methods draw on, and extend further, ideas from the existing literature. Supervised methods are expected to yield high effectiveness coupled with lower efficiency, while for unsupervised approaches it is the other way around. Our results reveal that it is more beneficial to use supervised learning for the candidate entity ranking step. If this step provides high-quality results, then disambiguation can be successfully tackled with a simple and elegant greedy algorithm. Moreover, our analysis shows that entity interdependencies provide little help for disambiguation. This is an interesting finding as it stands in contrast to the established postulation for entity linking in documents. Consequently, we identify a clearly preferred approach that uses supervised learning for candidate entity ranking and an unsupervised algorithm for disambiguation. Using the evaluation platform of the Entity Recognition and

Disambiguation (ERD) challenge [3], we show that our preferred approach performs on a par with the current state of the art.

The main contribution of this paper is to present the first systematic investigation of the ELQ task by bringing together the latest entity linking techniques and practices in a unified framework. In addition, we develop a novel supervised approach for entity disambiguation in ELQ, which encompasses various textual and KB-based relatedness features. Finally, we make a best practice recommendation for ELQ and demonstrate that our recommended approach achieves state-of-the-art performance. The resources developed with this paper are made available at http://bit.ly/ecir2017-elq.

Table 1. Example queries with their linked entities. Each set represents an interpretation of the query; ambiguous queries have multiple interpretations (i.e., multiple table rows).

Query	Entity linking interpretation(s)
Nashville thrift stores	{Nashville Tennessee, Charity shop}
Obama's wife	{Barack Obama}
Cambridge population	{Cambridge}
	{Cambridge (Massachusetts)}
New york pizza manhattan	{New York-style pizza, Manhattan}
	{New York, Manhattan}

2 Related Work

Early work on entity linking relied on the contextual similarity between the document and the candidate referent entities [7,24]. Milne and Witten [25] introduced the concepts of commonness and relatedness, which are generally regarded as two of the most important features for entity linking. In contrast to early systems that disambiguate one mention at a time, collective entity linking systems exploit the relatedness between entities jointly and disambiguate all entity mentions in the text simultaneously [15,19,21,29]. Since entity linking is a complex process, several attempts have been made to break it down into standard components and compare systems in a single framework [4,14,30]. Particularly, Hachey et al. [14] reimplemented three prominent entity linking systems in a single framework and found that much of the performance variation between these systems stems from the candidate entity ranking step (called *searcher* in their framework). We follow the final recommendation of their study and divide the entity linking task into two main steps, candidate entity ranking and disambiguation, to perform a systematic investigation of entity linking in queries.

Recognizing and disambiguating entities in short texts, such as tweets and search snippets, has only recently gained attention [11,13,23]. Entity linking in queries (ELQ) is particularly challenging because of the inherent ambiguity

(see Table 1). Deepak et al. [9] addressed ELQ by assigning a single entity to a mention. The Entity Recognition and Disambiguation (ERD) [3] challenge framed ELQ as the task of finding multiple interpretations of the query, and this was followed in subsequent studies [6,16,18]. Hasibi et al. [16] proposed generative models for ranking and disambiguating entities. The SMAPH system [6], on the other hand, "piggybacks" on a web search engine to rank entities, and then disambiguates them using a supervised collective approach. We consider the key features of these previous studies in a single system in order to perform a comprehensive comparison of the two main ELQ components (candidate entity ranking and disambiguation) with respect to both efficiency and effectiveness. We, however, do not include the piggybacking technique as its reliance on an external search service would seriously hinder the efficiency of the entity linking process in our setup.

3 Entity Linking in Queries

The task of entity linking in queries (ELQ) is to identify, given an input query q, a set of *entity linking interpretations* $I = \{E_1, \ldots, E_m\}$, where each interpretation $E_i = \{(m_1, e_1), \ldots, (m_k, e_k)\}$ consists of a set of mention-entity pairs. Mentions within E_i are non-overlapping and each mention m_j is linked to an entity e_j in a reference knowledge base. By way of illustration, the output of ELQ for the query "new york pizza manhattan" would be $I = \{E_1, E_2\}$, where $E_1 = \{$(new york pizza, New York-style pizza), (manhattan, Manhattan)$\}$ and $E_2 = \{$(new york, New York), (manhattan, Manhattan)$\}$. Following [3,16], we restrict ourselves to detecting proper noun entities and do not link general concepts (e.g., "Pizza").

We frame the ELQ problem as a sequence of the following two subtasks: *candidate entity ranking* (CER) and *disambiguation*. The first subtask takes the query q and outputs a ranked list of mention-entity pairs along with the corresponding scores. The second subtask takes this list as input and forms the set of entity linking interpretations I. For each subtask, we present two alternatives: unsupervised and supervised. The resulting four possible combinations are compared experimentally in Sect. 5.1.

3.1 Candidate Entity Ranking

This subtask is responsible for (i) identifying all possible entities that can be linked in the query and (ii) ranking them based on how likely they are link targets (in any interpretation of the query). The objective is to achieve both high recall and high precision at early ranks, as the top-ranked entity-mention pairs obtained here will be used directly in the subsequent disambiguation step. Using lexical matching of query n-grams against a rich dictionary of entity name variants allows for the identification of candidate entities with close to perfect recall [16]. We follow this approach to obtain a list of candidate entities together with their corresponding mentions in the query. Our focus of attention below is on ranking these candidate (m, e) pairs with respect to the query, i.e., estimating $score(m, e, q)$.

Unsupervised. For the unsupervised ranking approach, we take a state-of-the-art generative model, specifically, the MLMcg model proposed by Hasibi et al. [16]. This model considers both the likelihood of the given mention and the similarity between the query and the entity: $score(m, e, q) = P(e|m)P(q|e)$, where $P(e|m)$ is the probability of a mention being linked to an entity (a.k.a. *commonness* [22]), computed from the FACC collection [12]. The query likelihood $P(q|e)$ is estimated using the query length normalized language model similarity [20]:

$$P(q|e) = \frac{\prod_{t \in q} P(t|\theta_e)^{P(t|q)}}{\prod_{t \in q} P(t|C)^{P(t|q)}}, \tag{1}$$

where $P(t|q)$ is the term's relative frequency in the query (i.e., $n(t, q)/|q|$). The entity and collection language models, $P(t|\theta_e)$ and $P(t|C)$, are computed using the Mixture of Language Models (MLM) approach [27].

Supervised. Our supervised approach employs learning-to-rank (LTR), where each (query, mention, entity) triple is described using a set of features. The ranking function is trained on a set of mention-entity pairs with binary labels, with positive labels denoting the correctly annotated entities for the given query. We use a total of 28 features from the literature [6,23], which are summarized in Table 2.

3.2 Disambiguation

The disambiguation step is concerned with the formation of entity linking interpretations $\{E_1, \ldots, E_m\}$. Similar to the previous step, we examine both unsupervised and supervised alternatives, by adapting existing methods from the literature. We further extend the supervised approach with novel elements.

Unsupervised. We employ the greedy algorithm introduced in [16], which forms interpretations in three consecutive steps: (i) pruning, (ii) containment mention filtering, and (iii) set generation. In the first step, the algorithm takes the ranked list of mention-entity pairs and discards the ones with ranking score below the threshold τ_s. This threshold is a free parameter that controls the balance between precision and recall. The second step removes containment mentions (e.g., "kansas city mo" vs. "kansas city") by keeping only the highest scoring one. Finally, interpretations are built iteratively by processing mention-entity pairs in decreasing order of score and adding them to an existing interpretation E_i, where the mention does not overlap with other mentions already in E_i and i is minimal; if no such interpretation exists then a new interpretation $E_{|E|+1}$ is created.

Supervised. The overall idea is to generate all possible interpretations from a ranked list of mention-entity pairs, then employ a binary classifier to collectively select the most pertinent interpretations. Our approach is similar in spirit

Table 2. Feature set used for ranking entities, categorized to mention (M), entity (E), mention-entity (ME), and query (Q) features.

Feature	Description	Type
$Len(m)$	Number of terms in the entity mention	M
$NTEM(m)^{\ddagger}$	Number of entities whose title equals the mention	M
$SMIL(m)^{\ddagger}$	Number of entities whose title equals part of the mention	M
$Matches(m)$	Number of entities whose surface form matches the mention	M
$Redirects(e)$	Number of redirect pages linking to the entity	E
$Links(e)$	Number of entity out-links in DBpedia	E
$Commonness(e, m)$	Likelihood of entity e being the target link of mention m	ME
$MCT(e, m)^{\ddagger}$	True if the mention contains the title of the entity	ME
$TCM(e, m)^{\ddagger}$	True if title of the entity contains the mention	ME
$TEM(e, m)^{\ddagger}$	True if title of the entity equals the mention	ME
$Pos_1(e, m)$	Position of the 1^{st} occurrence of the mention in entity abstract	ME
$SimM_f(e, m)^{\dagger}$	Similarity between mention and field f of entity; Eq. (1)	ME
$LenRatio(m, q)$	Mention to query length ratio: $\frac{\|m\|}{\|q\|}$	Q
$QCT(e, q)$	True if the query contains the title of the entity	Q
$TCQ(e, q)$	True if the title of entity contains the query	Q
$TEQ(e, q)$	True if the title of entity is equal query	Q
$Sim(e, q)$	Similarity between query and entity; Eq. (1)	Q
$SimQ_f(e, q)^{\dagger}$	LM similarity between query and field f of entity; Eq. (1)	Q

‡ Entity title refers to the `rdfs:label` predicate of the entity in DBpedia

†Computed for all individual DBpedia fields $f \in \mathcal{F}$ and also for field *content* (cf. Sect. 4.1)

to the top performing contender in the ERD challenge [6], as they also select interpretations using a collective supervised approach. However, we generate the interpretations only from the top-K mention-entity pairs (obtained from the CER step) and generate all possible interpretations out of those. We further require that mentions within the same interpretation do not overlap with each other. The value of K is set empirically, and it largely depends on the effectiveness of the CER step. If CER has high precision then K can be low, while less effective approaches can be compensated for with higher K values.

Once the candidate sets are generated, each is represented by a feature vector. We devise two main families of features: (i) set-based features are computed for the entire interpretation set, and (ii) entity-based features are calculated for

Table 3. Feature set used in the supervised disambiguation approach. Type is either query dependent (QD) or query independent (QI).

Set-based features		Type
$CommonLinks(E)$	Number of common links in DBpedia: $\bigcap_{e \in E} out(e)$.	QI
$TotalLinks(E)$	Number of distinct links in DBpedia: $\bigcup_{e \in E} out(e)$	QI
$J_{KB}(E)$	Jaccard similarity based on DBpedia: $\frac{CommonLinks(E)}{TotalLink(E)}$	QI
$J_{corpora}(E)^{\ddagger}$	Jaccard similarity based on FACC: $\frac{\lvert\bigcap_{e \in E} doc(e)\rvert}{\lvert\bigcup_{e \in E} doc(e)\rvert}$	QI
$Rel_{MW}(E)^{\ddagger}$	Relatedness similarity [25] according to FACC	QI
$P(E)$	Co-occurrence probability based on FACC: $\frac{\lvert\bigcap_{e \in E} doc(e)\rvert}{TotalDocs}$	QI
$H(E)$	Entropy of E: $-P(E)log(P(E)) - (1-P(E))log(1-P(E))$	QI
$Completeness(E)^{\dagger}$	Completeness of set E as a graph: $\frac{\lvert edges(G_E)\rvert}{\lvert edges(K_{\lvert E\rvert})\rvert}$	QI
$LenRatioSet(E,q)^{\S}$	Ratio of mentions length to the query length: $\frac{\sum_{e \in E} \lvert m_e\rvert}{\lvert q\rvert}$	QD
$SetSim(E,q)$	Similarity between query and the entities in the set; Eq. (2)	QD
Entity-based features		
$Links(e)$	Number of entity out-links in DBpedia	QI
$Commonness(e,m)$	Likelihood of entity e being the target link of mention m	QD
$Score(e,q)$	Entity ranking score, obtained from the CER step	QD
$iRank(e,q)$	Inverse of rank, obtained from the CER step: $\frac{1}{rank(e,q)}$	QD
$Sim(e,q)$	Similarity between query and the entity; Eq. (1)	QD
$ContextSim(e,q)$	Contextual similarity between query and entity; Eq. (3)	QD

‡ $doc(e)$ represents all documents that have a link to entity e

$^{\dagger}G_E$ is a DBpedia subgraph containing only entities from E; and $K_{|E|}$ is a complete graph of $|E|$ vertices

$^{\S}m_e$ denotes the mention that corresponds to entity e

individual entities. Features in the first group are computed collectively on all entities of the set and measured as a single value, while the members of the second group need to be aggregated (we use min, max, avg as aggregators). It is worth noting that each interpretation typically consists of very few entities. Therefore, considering all entities for computing set-based features is feasible; it also captures more information than one could get from aggregated pair-wise similarity features. Table 3 summarizes our feature set.

We highlight two novel and important features. $SetSim(E,q)$ measures the similarity between all entities in the interpretation E and the query q:

$$SetSim(E,q) = P(q|\theta_E) = \frac{\prod_{t \in q} P(t|\theta_E)^{P(t|q)}}{\prod_{t \in q} P(t|C)^{P(t|q)}}. \quad (2)$$

It is calculated similar to Eq. (1), the main difference being that the probability of each term is estimated based on the interpretation's language model $P(t|\theta_E)$:

$$P(t|\theta_E) = \sum_{e \in E} \sum_{f \in F} \mu_f P(t|\theta_{e_f}).$$

In similar vein, $ContextSim(e, q)$ measures the similarity between the entity and the query context, where query context is the "rest" of the query, i.e., without the mention m_e that corresponds to entity e. Formally:

$$ContextSim(e, q) = P(q - m_e|e), \tag{3}$$

where $P(q - m_e|e)$ is computed using Eq. (1).

4 Experimental Setup

In this section we describe our data sources, settings of methods, and evaluation metrics.

4.1 Data

Knowledge base. We employ DBpedia 3.9 as our reference knowledge base and build an index of all entities that have both `rdfs:label` and `dbo:comment` predicates. The index includes the following set of fields: $\mathcal{F}$ ={*title*, *content*, `rdfs:label`, `dbo:wikiPageWikiLink`, `rdfs:comment`, `dbo:abstract`}, where *title* is the concatenation of `rdfs:label`, `foaf:name` and `dbo:wikiPage-Redirects` predicates, and *content* holds the content of all predicates of the entity; the remaining fields correspond to individual predicates.

Surface form dictionary. To recognize candidate entities in queries, we employ a rich surface form dictionary, which maps surface forms to entities. We utilize the FACC entity-annotated web corpora [12] and include surface forms above a commonness threshold of 0.1 [16]. Additionally, we add DBpedia name variants as surface forms; i.e., entity names from `rdfs:label`, `foaf:name`, and `dbo:wikiPageRedirects` predicates [7,11,16]. We confine our dictionary to entities present in the Freebase snapshot of proper named entities, provided by the ERD challenge [3].

Test Collections. We evaluate our methods on two publicly available test collections: Y-ERD [16] and ERD-dev [3]. The former is based on the Yahoo Search Query Log to Entities (YSQLE) dataset[1] and consists of 2,398 queries. All results on this collection are obtained by performing 5-fold cross validation.[2]

[1] http://webscope.sandbox.yahoo.com/.

[2] It is important to note that Y-ERD contains queries that have been reformulated (often only slightly so) during the course of a search session; we ensure that queries from the same session are assigned to the same fold when using cross-validation.

The ERD-dev collection contains 91 queries and is released as part of the ERD challenge [3]. We apply the trained models (on the whole Y-ERD collection) to ERD-dev queries and report on the results. In addition, ERD also provides an online evaluation platform which is based on a set of 500 queries (referred to as ERD-test); the corresponding annotations are not released. We evaluate the effectiveness[3] of our recommended system using ERD-test to evaluate how it performs against the current state of the art.

4.2 Methods

Candidate Entity Ranking. For the unsupervised method (**MLMcg**), we follow [26] and use title and content fields, with weights 0.2 and 0.8, respectively. For the supervised method (**LTR**), we employ the Random Forest (RF) [2] ranking algorithm and set the number of trees to 1000 and the maximum features to 10% of size of the feature set [23]. We further include two baseline methods for reference comparison: (i) **MLM** is similar to MLMcg, but without considering the commonness score; i.e., computed based on the Eq. (1); (ii) **CMNS** ranks entities based on the commonness score, while prioritizing longer mentions, and is shown to be a strong baseline [1,16,23].

Disambiguation. The unsupervised disambiguation method (**Greedy**) involves a score threshold parameter, which is set (using a parameter sweep) depending on the CER method used: 20 for MLMcg and 0.3 in case of LTR. For the supervised disambiguation method (**LTR**), we set the number of top ranked entities K to 5 (based on a parameter sweep) and use a RF classifier with similar setting to supervised CER. For baseline comparison, we consider the top-3 performing systems from the ERD challenge: SMAPH [6], NTUNLP [5], and Seznam [10].

4.3 Evaluation

As both precision and recall matter for the candidate entity ranking step, we evaluate our methods using Mean Average Precision (MAP), recall at rank 5 (R@5), and precision at position 1 (P@1). When evaluating CER, we are only concerned about the ranking of entities; therefore, we consider each entity only once with its highest scoring mention: $score(e, q) = \arg\max_{m \in q} score(m, e, q)$. For the disambiguation step, we measure the end-to-end performance using set-based metrics (precision, recall, and F-measure), according to the strict evaluation metrics in [16]. As for efficiency, we report on the average processing time for each query, measured in seconds. The experiments were conducted on a machine with an Intel Xeon E5 2.3GHz 12-core processor, running Ubuntu Linux v14.04. Statistical significance is tested using a two-tailed paired t-test. We mark improvements with $^{\triangle}(p < 0.05)$ or $^{\blacktriangle}(p < 0.01)$, detoriations with $^{\triangledown}(p < 0.05)$ or $^{\blacktriangledown}(p < 0.01)$, and no significance by $^{\circ}$.

[3] Carmel et al. [3] do not report on the efficiency of the approaches and the online leaderboard is no longer available, hence we present only effectiveness results from Cornolti et al. [6].

5 Results and Analysis

In this section we report on our experimental results and answer our research questions.

5.1 Results

We start by evaluating the *candidate entity ranking* and *disambiguation* steps and then answer our first research question: "Given the response time requirements of an online setting, what is the relative importance of candidate entity ranking vs. disambiguation?"

Table 4. Candidate entity ranking results on the Y-ERD and ERD-dev datasets. Best scores for each metric are in boldface. Significance for line $i > 1$ is tested against lines $1..i-1$.

Method	Y-ERD			ERD-dev		
	MAP	R@5	P@1	MAP	R@5	P@1
MLM	0.7507	0.8556	0.6839	0.7675	0.8622	0.7333
CMNS	$0.7831^{\blacktriangle}$	$0.8230^{\blacktriangle}$	$0.7779^{\blacktriangle}$	0.7037°	$0.7222^{\triangledown}$	0.7556°
MLMcg	$0.8536^{\blacktriangle\blacktriangle}$	$0.8997^{\blacktriangle\blacktriangle}$	$0.8280^{\blacktriangle\blacktriangle}$	$0.8543^{\triangle\blacktriangle}$	$0.9015^{\circ\blacktriangle}$	$\mathbf{0.8444}^{\circ\circ}$
LTR	$\mathbf{0.8667}^{\blacktriangle\blacktriangle\blacktriangle}$	$\mathbf{0.9022}^{\blacktriangle\blacktriangle\circ}$	$\mathbf{0.8479}^{\blacktriangle\blacktriangle\blacktriangle}$	$\mathbf{0.8606}^{\triangle\blacktriangle\circ}$	$\mathbf{0.9289}^{\triangle\blacktriangle\circ}$	$0.8222^{\circ\circ\circ}$

Candidate Entity Ranking. Table 4 presents the results for CER on the Y-ERD and ERD-dev datasets. We find that commonness is a strong performer (this is in line with the findings of [1,16]). Combining commonness with MLM in a generative model (MLMcg) delivers excellent performance, with MAP above 0.85 and R@5 around 0.9. The LTR approach can bring in further slight, but for Y-ERD significant, improvements. This means that both of our CER methods (MLMcg and LTR) are able to find the vast majority of the relevant entities and return them at the top ranks.

Disambiguation. Table 5 reports on the disambiguation results. We use the naming convention *X*-*Y*, where *X* refers to the CER method (MLMcg or LTR) and *Y* refers to the disambiguation method (Greedy or LTR) that is applied on top. Our observations are as follows. The MLM-Greedy approach is clearly the most efficient but also the least effective one. Learning is more expensive for disambiguation than for CER, see LTR-Greedy vs. MLMcg-LTR; yet, it is also clear from this comparison that more performance can be gained when learning is done for CER than when it is done for disambiguation. The most effective method is LTR-Greedy, outperforming other approaches significantly on both test sets. It is also the second most efficient one. Interestingly, even though the MLMcg and LTR entity ranking methods perform equally well according to CER evaluation (cf. Table 4), we observe a large difference in their performance when

the Greedy disambiguation approach is applied on top of them. The reason is that the absolute scores produced by LTR are more meaningful than those of MLMcg (despite the query length normalization efforts for the latter; cf. Eq. (1)). This plays a direct role in Greedy disambiguation, where score thresholding is used. We note that the reported efficiency results are meant for comparison across different approaches. For practical applications, further optimizations to our basic implementation would be needed (cf. [1]).

Table 5. End-to-end performance of ELQ systems on the Y-ERD and ERD-dev query sets. Significance for line $i > 1$ is tested against lines $1..i-1$.

Method	Y-ERD				ERD-dev			
	Prec	Recall	F1	Time	Prec	Recall	F1	Time
MLMcg-Greedy	0.709	0.709	0.709	**0.058**	0.724	0.712	0.713	**0.085**
MLMcg-LTR	0.725°	0.724°	0.724°	0.893	0.725°	0.731°	0.728°	1.185
LTR-LTR	$0.731^{\triangle\circ}$	$0.732^{\triangle\circ}$	$0.731^{\triangle\circ}$	0.881	$0.758^{\circ\circ}$	$0.748^{\circ\circ}$	$0.753^{\circ\circ}$	1.185
LTR-Greedy	$\mathbf{0.786}^{\blacktriangle\blacktriangle\blacktriangle}$	$\mathbf{0.787}^{\blacktriangle\blacktriangle\blacktriangle}$	$\mathbf{0.787}^{\blacktriangle\blacktriangle\blacktriangle}$	0.382	$\mathbf{0.852}^{\blacktriangle\blacktriangle\triangle}$	$\mathbf{0.828}^{\blacktriangle\triangle\circ}$	$\mathbf{0.840}^{\blacktriangle\blacktriangle\triangle}$	0.423

Based on the results, LTR-Greedy is our overall recommendation. We compare this method against the top performers of the ERD challenge (using the official challenge platform); see Table 6. For this comparison, we additionally applied spell checking, as this has also been handled in the top performing system (SMAPH-2) [6]. The results show that our LTR-Greedy approach performs on a par with the state-of-the-art systems. This is remarkable taking into account the simplicity of the Greedy disambiguation algorithm vs. the considerably more complex solutions employed by others.

Table 6. ELQ results on the official ERD test platform.

Method	F1
LTR-Greedy	0.699
SMAPH-2 [6]	0.708
NTUNLP [5]	0.680
Seznam [10]	0.669

Answer to RQ1. Our results reveal that candidate entity ranking is of higher importance than disambiguation for ELQ. Hence, it is more beneficial to perform the (expensive) supervised learning early on in the pipeline for the seemingly easier CER step; disambiguation can then be tackled successfully with an unsupervised (greedy) algorithm. (Note that selecting the top ranked entity does not yield an immediate solution; as shown in [16], disambiguation is an indispensable step in ELQ.)

5.2 Feature Analysis

We now analyze the features used in our supervised methods and answer our second research question: "Given the limited context provided by queries, which group of features is needed the most for effective entity disambiguation?" For

the sake of completeness, we also report feature importance for the CER step, even though that does not directly relate to the above RQ. Figure 1(a) shows the top features used in the LTR entity ranking approach in terms of Gini score. We observe that *Matches, Commonness*, and the various query similarity features play the main role in the entity ranking function. As for the supervised disambiguation step, which is our main focus here, we selected the top 15 features independently for the MLMcg-LTR and LTR-LTR methods; interestingly, we ended up with the exact same set of features. Figure 1(b) demonstrates that nearly all influential features are query dependent; the only query independent features are P and H, capturing the co-occurrence of entities in web corpora.

Answer to RQ2. We conclude that contextual similarity features are the most effective for entity disambiguation. This is based on two observations: (i) the unsupervised (Greedy) method takes only the entity ranking scores as input, which are computed based on the contextual similarity between entity and query; (ii) the supervised (LTR) method relies the most on query-dependent features. This is an interesting finding, as it stands in contrast to the common postulation in entity linking in documents that interdependence between entities help to better disambiguate entities. Entity interdependence features (and, in general, collective disambiguation methods) are more helpful when sufficiently many entities are mentioned in the text; this is not the case for queries.

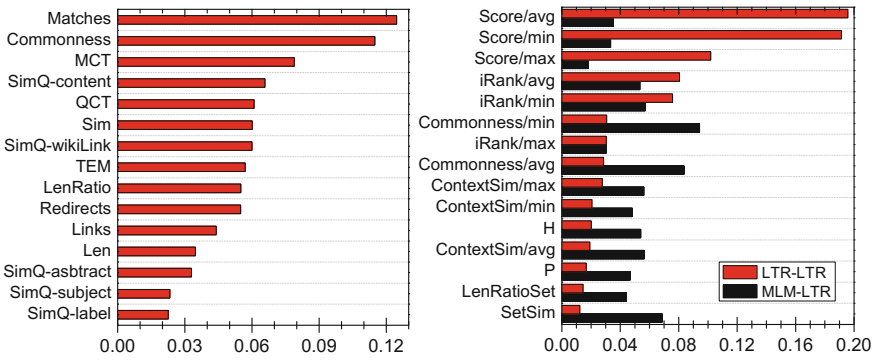

Fig. 1. Most important features used in the supervised approaches, sorted by Gini score: (Left) candidate entity ranking, (Right) disambiguation.

6 Conclusion

In this paper, we have performed the first systematic investigation of entity linking in queries (ELQ). We have developed a framework where different methods can be plugged in for two core components: *candidate entity ranking* and *disambiguation*. For each of these components, we have explored both unsupervised and supervised alternatives by employing and further extending state-of-the-art

approaches. Our experiments have led to two important findings: (i) it is more rewarding to employ supervised learning for candidate entity ranking than for disambiguation, and (ii) entity interdependence features, which are the essence of collective disambiguation methods, have little benefit for ELQ. Overall, our findings have not only revealed important insights, but also provide guidance as to where future research and development in ELQ should be focused.

References

1. Blanco, R., Ottaviano, G., Meij, E.: Fast and space-efficient entity linking in queries. In: Proceedings of WSDM, pp. 179–188 (2015)
2. Breiman, L.: Random forests. Mach. Learn. **45**, 5–32 (2001)
3. Carmel, D., Chang, M.-W., Gabrilovich, E., Hsu, B.-J.P., Wang, K.: ERD 2014: entity recognition and disambiguation challenge. In: ACM SIGIR Forum, vol. 48, pp. 63–77 (2014)
4. Ceccarelli, D., Lucchese, C., Orlando, S., Perego, R., Trani, S.: Learning relatedness measures for entity linking. In: Proceedings of CIKM, pp. 139–148 (2013)
5. Chiu, Y.-P., Shih, Y.-S., Lee, Y.-Y., Shao, C.-C., Cai, M.-L., Wei, S.-L., Chen, H.-H.: NTUNLP approaches to recognizing and disambiguating entities in long and short text at the ERD challenge 2014. In: Proceedings of ERD@SIGIR (2014)
6. Cornolti, M., Ferragina, P., Ciaramita, M., Rüd, S., Schütze, H.: A piggyback system for joint entity mention detection and linking in web queries. In: Proceedings of WWW, pp. 567–578 (2016)
7. Cucerzan, S.: Large-scale named entity disambiguation based on Wikipedia data. In: Proceedings of EMNLP-CoNLL, pp. 708–716 (2007)
8. Dalton, J., Dietz, L., Allan, J.: Entity query feature expansion using knowledge base links. In: Proceedings of SIGIR, pp. 365–374 (2014)
9. Deepak, P., Ranu, S., Banerjee, P., Mehta, S.: Entity linking for web search queries. In: Hanbury, A., Kazai, G., Rauber, A., Fuhr, N. (eds.) ECIR 2015. LNCS, vol. 9022, pp. 394–399. Springer, Cham (2015). doi:10.1007/978-3-319-16354-3_43
10. Eckhardt, A., Hreško, J., Procházka, J., Smrs, O.: Entity linking based on the co-occurrence graph and entity probability. In: Proceeding of ERD@SIGIR (2014)
11. Ferragina, P., Scaiella, U.: TAGME: on-the-fly annotation of short text fragments (by Wikipedia entities). In: Proceedings of CIKM, pp. 1625–1628 (2010)
12. Gabrilovich, E., Ringgaard, M., Subramanya, A.: FACC1: freebase annotation of ClueWeb corpora, Version 1 (2013)
13. Guo, S., Chang, M.-W., Kiciman, E.: To link or not to link? A study on end-to-end tweet entity linking. In: HLT-NAACL, pp. 1020–1030 (2013)
14. Hachey, B., Radford, W., Nothman, J., Honnibal, M., Curran, J.R.: Evaluating entity linking with Wikipedia. Artif. Intell. **194**, 130–150 (2013)
15. Han, X., Sun, L., Zhao, J.: Collective entity linking in web text: a graph-based method. In: Proceedings of SIGIR, pp. 765–774 (2011)
16. Hasibi, F., Balog, K., Bratsberg, S.E.: Entity linking in queries: tasks and evaluation. In: Proceedings of ICTIR, pp. 171–180 (2015)
17. Hasibi, F., Balog, K., Bratsberg, S.E.: Exploiting entity linking in queries for entity retrieval. In: Proceedings of ICTIR, pp. 209–218 (2016)
18. Hasibi, F., Balog, K., Bratsberg, S.E.: On the reproducibility of the TAGME entity linking system. In: Ferro, N., Crestani, F., Moens, M.-F., Mothe, J., Silvestri, F., Nunzio, G.M., Hauff, C., Silvello, G. (eds.) ECIR 2016. LNCS, vol. 9626, pp. 436–449. Springer, Cham (2016). doi:10.1007/978-3-319-30671-1_32

19. 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 EMNLP, pp. 782–792 (2011)
20. Kraaij, W., Spitters, M.: Language models for topic tracking. In: Language Modeling for Information Retrieval, pp. 95–123 (2003)
21. Kulkarni, S., Singh, A., Ramakrishnan, G., Chakrabarti, S.: Collective annotation of Wikipedia entities in web text. In: Proceedings of SIGKDD, pp. 457–466 (2009)
22. Medelyan, O., Witten, I.H., Milne, D.: Topic indexing with Wikipedia. In: Proceedings of the Wikipedia and AI Workshop at the AAAI 2008 Conference (2008)
23. Meij, E., Weerkamp, W., de Rijke, M.: Adding semantics to microblog posts. In: Proceedings of WSDM, pp. 563–572 (2012)
24. Mihalcea, R., Csomai, A.: Wikify!: linking documents to encyclopedic knowledge. In: Proceedings of CIKM, pp. 233–242 (2007)
25. Milne, D., Witten, I.H.: Learning to link with Wikipedia. In: Proceedings of CIKM, pp. 509–518 (2008)
26. Neumayer, R., Balog, K., Nørvåg, K.: When simple is (more than) good enough: effective semantic search with (almost) no semantics. In: Baeza-Yates, R., Vries, A.P., Zaragoza, H., Cambazoglu, B.B., Murdock, V., Lempel, R., Silvestri, F. (eds.) ECIR 2012. LNCS, vol. 7224, pp. 540–543. Springer, Heidelberg (2012). doi:10.1007/978-3-642-28997-2_59
27. Ogilvie, P., Callan, J.: Combining document representations for known-item search. In: Proceedings of SIGIR, pp. 143–150 (2003)
28. Schuhmacher, M., Dietz, L., Paolo Ponzetto, S.: Ranking entities for Web queries through text and knowledge. In: Proceedings of CIKM, pp. 1461–1470 (2015)
29. Sen, P.: Collective context-aware topic models for entity disambiguation. In: Proceedings of WWW, pp. 729–738 (2012)
30. Usbeck, R. et al.: GERBIL: general entity annotator benchmarking framework. In: Proceedings of WWW, pp. 1133–1143 (2015)
31. Xiong, C., Callan, J.: EsdRank: Connecting query and documents through external semi-structured data. In: Proceedings of CIKM, pp. 951–960 (2015)
32. Yilmaz, E., Verma, M., Mehrotra, R., Kanoulas, E., Carterette, B., Craswell, N.: Overview of the TREC 2015 tasks track. In: Proceedings of TREC (2015)