Entity Linking in Queries: Tasks and Evaluation

Article · September 2015

DOI: 10.1145/2808194.2809473

Faegheh Hasibi
Radboud University

Krisztian Balog
University of Stavanger (UiS)

Svein Erik Bratsberg
Norwegian University of Science and Technology

Entity Linking in Queries: Tasks and Evaluation

Faegheh Hasibi
Norwegian University of
Science and Technology
faegheh.hasibi@idi.ntnu.no

Krisztian Balog
University of Stavanger
krisztian.balog@uis.no

Svein Erik Bratsberg
Norwegian University of
Science and Technology
sveinbra@idi.ntnu.no

ABSTRACT

Annotating queries with entities is one of the core problem areas in query understanding. While seeming similar, the task of entity linking in queries is different from entity linking in documents and requires a methodological departure due to the inherent ambiguity of queries. We differentiate between two specific tasks, semantic mapping and interpretation finding, discuss current evaluation methodology, and propose refinements. We examine publicly available datasets for these tasks and introduce a new manually curated dataset for interpretation finding. To further deepen the understanding of task differences, we present a set of approaches for effectively addressing these tasks and report on experimental results.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval

Keywords

Entity linking; semantic mapping; interpretation finding; query understanding

1. INTRODUCTION

Query understanding has been a longstanding area of research in information retrieval [13, 38]. One way of capturing what queries are about is to annotate them with entities from a knowledge base. This general problem has been studied in many different forms and using a variety of techniques over the recent years [6, 8, 10, 12, 15, 27]. Approaches have been inspired by methods that recognize and disambiguate entities appearing in full-text documents by mapping them to the corresponding entries in a knowledge base, a process known as *entity linking* [29] (or *wikification* [31]). Successful approaches to entity linking incorporate context-based features in a machine learning framework to disambiguate between entities that share the same surface form [16, 30–32]. While the same techniques can be applied directly to short, noisy texts, such as microblogs or search queries, there is experimental evidence showing that the same methods perform substantially worse on short texts (tweets) than on longer documents (news) [11, 37]. One problem is the lack of proper spelling and grammar, even of the most basic sort, like capitalization and punctuation. Therefore, approaches that incorporate richer linguistic analysis of text cannot be applied.

There is, however, an even more fundamental difference concerning entity annotations in documents vs. queries that has not received due attention in the literature. When evaluating entity linking techniques for documents, it is implicitly assumed that the text provides enough context for each entity mention to be resolved unambiguously. Search queries, on the other hand, typically consist of only a few terms, providing limited context. Specifically, we focus on a setting where there is no context, such as previous queries or clicked results within a search session, available for queries. In this setting, it may be impossible to select a single most appropriate entity for a given query segment. Consider, as an illustrative example, the query "new york pizza manhattan." It could be annotated, among others, as "[NEW YORK CITY] pizza [MANHATTAN]" or as "[NEW YORK-STYLE PIZZA][MANHATTAN]," and both would be correct (linked entities are in brackets).

A cardinal question, then, is how should the inherent ambiguity of entity annotations in queries be handled? One line of prior work has dealt with this problem by adopting a retrieval-based approach: returning a ranked list of entities that are semantically related to the query [6, 27]. We refer to it as *semantic mapping*. The Entity Recognition and Disambiguation (ERD) Challenge [8] represents a different perspective by addressing the issue of ambiguity head-on: search queries can legitimately have more than a single interpretation. An interpretation is a set of entities, with non-overlapping mentions, that are semantically compatible with the query text [8]. We term this task *interpretation finding*. Both approaches have their place, but there is an important distinction to be made as they are designed to accomplish different tasks. Semantic mapping is a tool for aiding users with suggestions that could be beneficial for enhancing navigation or for contextualization. Interpretation finding is a means to machine-understanding of queries, which, in our opinion, is the ultimate goal of entity linking in queries.

Once these differences are established and the tasks are defined, our next research question concerns the evaluation methodology and metrics. The current practice of rank-based evaluation is appropriate for the semantic mapping task. As for interpretation finding, interpretations are considered as atomic units, i.e., an interpretation is correct only if it contains the exact same entities as the ground truth; partial matches are not rewarded [8]. This is a rather crude method of evaluation. We present a relaxed alternative that considers both the correctness of interpretations, as atomic units, and the set of entities, recognized in the query.

As with any problem in information retrieval, the availability of public datasets is of key importance. The recently released Yahoo! Webscope Search Query Log to Entities (YSQLE) dataset [1], is suitable for semantic mapping, but not for interpretation finding.

ICTIR '15, September 27-30, 2015, Northampton, MA, USA.
Copyright 2015 ACM ISBN 978-1-4503-3833-2/15/09...$ 15.00.
DOI: http://dx.doi.org/10.1145/2808194.2809473.

The ERD Challenge platform [8] is fitting for interpretation finding, however, only the development set (91 queries) is publicly available, which is not large enough for training purposes. We therefore, introduce and make publicly available a new dataset based on YSQLE, called Y-ERD. It contains interpretations for 2398 queries and is accompanied by a clear set of annotation guidelines.

In addition, we present simple, yet effective methods for addressing the semantic mapping and interpretation finding tasks. We introduce a pipeline architecture for both and identify shared components. Finally, we evaluate our approaches on the different datasets, which offer further insights into these tasks.

In summary, the main theoretical contribution of this work is the methodological distinction between two tasks within the problem area of entity linking in queries: semantic mapping and interpretation finding. Technical contributions include (i) the development of a dataset and evaluation methodology for interpretation finding, (ii) solid and easy-to-implement approaches for both tasks, and (iii) experimental results and insights. All resources developed within this paper are made publicly available at http://bit.ly/ictir2015-elq.

2. RELATED WORK

In this section we review related work on entity linking and on query understanding, and finally on the intersection of these two.

2.1 Entity linking

Recognizing entity mentions in text and linking them to the corresponding entries in a knowledge base provides means for understanding documents (and queries). The reference knowledge base is most commonly Wikipedia. The Wikify! system [31], one of the earliest approaches, performs concept detection by extracting all n-grams that match Wikipedia concepts and then filters them. Their most effective filtering approach utilizes link probabilities obtained from Wikipedia articles. For the entity disambiguation step, they use a combination of knowledge-based and feature-based learning approaches. In another early work, Cucerzan [14] employs contextual and category information extracted from Wikipedia and calculates the similarity between the document and candidate entities' pages. Later, Milne and Witten [32] employed a machine learning approach, using commonness and relatedness as main features. Their work gained substantial improvements over prior approaches. DBpedia Spotlight [30] is another entity linking system, which uses the Vector Space Model to disambiguate named entities.

2.2 Query understanding

Query understanding refers to process of "identifying the underlying intent of the queries, based on a particular representation" [13]. One main branch of approaches focuses on determining the "aboutness" of queries by performing a topical classification of the query contents [7, 22, 25, 39]. Segmentation represents another approach to understanding queries, where the query is divided into phrases, such that each phrase can be considered as an individual concept [4, 21, 40]. Although query segmentation is not directly related to our task, some ideas can be borrowed for grouping named entities and forming interpretation sets. For instance, Hagen et al. [21] used n-gram frequencies and Wikipedia to efficiently segment queries. They incorporated the length of each segment as a weight factor; this is done to favor long segments with low frequency to short ones with high frequency.

Recognizing named entities in queries was first addressed by Guo et al. [18]; their goal was to detect named entities in a given query and classify them into a set of predefined classes such as "movie" or "music." The proposed approach employs probabilistic methods together with a weakly supervised learning algorithm (WD-LDA). Alasiry et al. [2] proposed a processing pipeline for entity detection in queries, which involves the following steps: query pre-processing (e.g., spell checking), grammar annotation (POS and ORTH tagging), segmentation, and entity recognition (based on a small set of manually constructed rules). Importantly, these works are limited to detecting mentions of entities and do not perform disambiguation or linking; that follows in the next subsection.

2.3 Entity linking in queries

Entity linking for short texts, such as queries and tweets, has gained considerable attention recently. The TAGME system [16] extends the approach of Milne and Witten [32] by incorporating a voting schema for the relatedness feature and by discarding unrelated anchors. Meij et al. [28] proposed a two step approach for linking tweets to Wikipedia articles. In the first step, they extract candidate Wikipedia concepts for each n-gram. Next, a supervised learning algorithm with an excessive set of features is used to classify relevant concepts. Their strategy is to first obtain high recall and then improve precision by employing machine learning. Guo et al. [19] also studied microblog texts and employed a structural SVM algorithm in a single end-to-end task for mention detection and entity disambiguation.

Unlike these works, which revolve around ranking entities for query spans, the Entity Recognition and Disambiguation (ERD) Challenge [8] viewed entity linking in queries as the problem of finding multiple query interpretations. The task advances the conventional entity linking task (as it is known for long texts) and finds set(s) of (semantically related) linked entities, where each set reflects a possible meaning (interpretation) of the query. Even though it was one of the main considerations behind the ERD Challenge to capture multiple query interpretations, only a handful of systems actually attempted to address that; out of these, [15] performed best and was the third best performing system in overall. In this paper, we discuss why entity linking in queries should be addressed as an interpretation finding task and how the other tasks studied in the literature are different from it.

3. TASKS AND EVALUATION

In this section we first discuss the entity linking task for documents in Section 3.1. Next, in Section 3.2, we identify some principal differences when the same task is to be performed for queries and point out why the same evaluation methodology cannot be used. In Section 3.3 we look at the semantic mapping task and show that albeit the current practice of rank-based evaluation is appropriate, the task itself is easier than entity linking and resembles more of a *related entity finding* problem. Finally, in Section 3.4 we present the interpretation finding task, which deals with the inherent ambiguity of search queries. We introduce existing evaluation metrics and also propose refinements to the evaluation metrics used in [8] that can give credit for partial correctness.

3.1 Entity linking for documents

Entity linking is the task of recognizing entity mentions in text and linking (disambiguating) them each to the most appropriate entry in a reference knowledge base. This task implicitly assumes that the input text (document) provides enough context so that all entity occurrences can be resolved unambiguously.

Evaluation is performed against a gold-standard data set that consists of manual annotations. These annotations comprise of the specific entity mentions (offsets in the text) and the corresponding links to the knowledge base. Effectiveness is measured in terms of precision and recall, where precision is defined as the number of correctly linked mentions divided by the total number of links established by the system, and recall is defined as the number of

	Entity Linking†	**Semantic Mapping**	**Interpretation Finding**
Result	set	ranked list	sets of sets
Entities explicitly mentioned	Yes	No	Yes
Mentions can overlap	No	Yes	No‡
Evaluation criteria	mentioned entities found	relevant entities found	interpretations found
Evaluation metrics	set-based	rank-based	set-based
Examples			
"obama mother"	{Barack Obama}	Ann Dunham Barack Obama	{{Barack Obama}}
"new york pizza manhattan"	{New York City, Manhattan}*	New York City New York-style pizza Manhattan Manhattan Pizza . . .	{{New York City, Manhattan}, {New York-style pizza, Manhattan}}

"the music man"	{The Music Man}*	The Music Man
		The Music Man (1962 film)
		The Music Man (2003 film)
		. . .
			{{The Music Man}
			{The Music Man (1962 film)},
			{The Music Man (2003 film)}}

†This refers to traditional entity linking (for documents) applied to queries. We argue in this paper that entity linking in this form should be avoided.

‡Not within the same interpretation.

*A single interpretation is selected arbitrarily; there are multiple options.

Table 1: Entity linking tasks.

correctly linked mentions divided by the total number of links in the gold-standard annotations [31]. For overall system evaluation the F-measure is used. Both micro- and macro-averaging can be employed [11]. Since mention segmentation is often ambiguous, and the main focus is on the disambiguation of entities, the correctness of entity mention boundaries is often relaxed [8]. On the other hand, evaluation is rather strict in that credit is only given for a given mention if the linked entity (unique entity identifier) perfectly matches the gold standard. Overlapping entity mentions in the annotations are not allowed, i.e., any given segment of the document may be linked to at most a single entity.

3.2 Entity linking for queries

Existing entity linking approaches can be used out-of-the-box to annotate queries with entities, analogously to how it is done for documents; after all, the input is text, which is the same as before, just shorter and less grammatical (the quality of the resulting annotations is another matter). The fundamental difference between documents and queries is that queries offer very limited context. A search query, therefore, "can legitimately have more than one interpretation," where each interpretation consists of a set of "non-overlapping linked entity mentions that are semantically compatible with the query text" [8]. Formally, let q be a query and $\hat{I}$ be the set of interpretations for this query (according to the ground truth), $\hat{I} = \{\hat{E}_1, \ldots, \hat{E}_n\}$, where n is the number of interpretations, $\hat{E}_i$ is a query interpretation, $\hat{E}_i = \{(m_1, e_1), \ldots, (m_k, e_k)\}$, and (m, e) is a mention-entity pair. For simplicity, the specific offsets of entity mentions are not considered, however, the corresponding entity mentions in $\hat{E}_i$ must not overlap. It is important to point out that the query might not have any interpretations ($\hat{I} = \emptyset$).

If the traditional evaluation methodology were to be adopted (as in Section 3.1, with the simplification of ignoring the offsets of mentions), the ground truth would need to consist of a single set of entities; we denote this set as $\hat{E}$. As long as the query has a single interpretation it is straightforward; entities in that interpretation will amount to the ground truth set. Having no valid interpretation is also painless, we set $\hat{E} = \emptyset$. For entities with multiple interpretations, there are two natural ways of setting $\hat{E}$.

Collapsing interpretations The first option is to collapse all interpretations into a single set: $\hat{E} = \bigcup_{i \in [1..n]} \hat{E}_i$. (This is similar in spirit to the approach that is followed in the semantic mapping task, see later in Section 3.3.) With this solution, however, the requirements that the linked entities within an interpretation must be semantically related and their mentions must not overlap are violated. It also ignores the element of multiple interpretations altogether.

Selecting a single interpretation The second option is to pick a single interpretation $\hat{E} = \hat{E}_j$, where $j \in [1..n]$. Given that all interpretations are of equal importance, selecting j in an arbitrary way would be unfair, as it would randomly favor certain systems over others. A better alternative would be to choose j individually for each system such that it maximizes the system's performance on a given evaluation metric, e.g., F1-score. Essentially, the system's output would be scored based on the closest matching interpretation. While the latter variant appears to be a viable solution, it still disregards the fundamental aspects of finding multiple interpretations for queries.

In summary, the entity linking task cannot be performed the same way for queries as it is done for documents, because of the element of multiple interpretations. In the remainder of this section we discuss two alternatives used in the literature and make suggestions for further methodological refinements.

3.3 Semantic mapping

Semantic mappings are primarily intended to support users in their search and browsing activities by returning entities that can help them to acquire contextual information or valuable navigational suggestions [27]. For semantic mapping, all entities from all interpretations are relevant. Beyond those, entities that are not explicitly mentioned, but referred to, may also be considered relevant; we elaborate further on this in Section 7. See Table 1 for illustrative examples. The goal, therefore, is quite different from that of finding interpretation(s) of the query for machine understanding. The requirements on the linked entities are relaxed: (i) the mentions can be overlapping, (ii) they do not need to form semantically compatible sets, (iii) they do not even need to be explicitly mentioned, as long as they are semantically related to the query.

Formally, let $\hat{E}$ denote the set of relevant entities for the semantic mapping task. This set is formed from entities across all interpretations, plus, optionally, additional entities (E^*) that are indirectly referenced from the query: $\hat{E} = \bigcup_{i \in [1..n]} \hat{E}_i \cup E^*$. Semantic mapping returns a ranked list of entities $\vec{E} = \langle e_1, \ldots, e_m \rangle$, which is compared against $\hat{E}$ using standard rank-based metrics, such as mean average precision (MAP) or mean reciprocal rank (MRR). Importantly, if $\hat{E} = \emptyset$ then the given query is ignored in the evaluation, meaning, that there is no difference made between system A that does not return anything and system B that returns meaningless or nonsense suggestions. This is undesired behavior; it also stands in contrast to standard entity linking, where false positives decrease system performance.

It is our opinion that above relaxations make semantic mapping a substantially easier task (that of finding "related entities") than what entity linking for queries entails in its entirety. Therefore, we believe that the terminological distinction is important and useful.

3.4 Interpretation finding

The inherent presence of multiple query interpretations is addressed head-on by the setup introduced at the Entity Recognition and Disambiguation (ERD) Challenge [8], where "interpretations of non-overlapping linked entity mentions" [8] are to be returned. We argue that this formulation is the proper way to go about entity linking in queries.

We write $\hat{I} = \{\hat{E}_1, \ldots, \hat{E}_m\}$ to denote the query interpretation according to the ground truth, and $I = \{E_1, \ldots, E_n\}$ is the interpretation returned by the system. Precision and recall, for a given query, are defined at the ERD Challenge as follows:

$$\mathrm{P} = \frac{|I \cap \hat{I}|}{|I|}, \qquad \mathrm{R} = \frac{|I \cap \hat{I}|}{|\hat{I}|}. \tag{1}$$

Note that according to this definition, if the query does not have any interpretations in the ground truth ($\hat{I} = \emptyset$) then precision is undefined; similarly, if the system does not return any interpretations ($I = \emptyset$), then recall is undefined. We correct for this behavior by defining precision and recall for interpretation-based evaluation:

$$\mathrm{P}_{\mathrm{int}} = \begin{cases} |I \cap \hat{I}|/|I|, & I \neq \emptyset \\ 1, & I = \emptyset, \hat{I} = \emptyset \\ 0, & I = \emptyset, \hat{I} \neq \emptyset. \end{cases} \tag{2}$$

$$\mathrm{R}_{\mathrm{int}} = \begin{cases} |I \cap \hat{I}|/|\hat{I}|, & \hat{I} \neq \emptyset \\ 1, & \hat{I} = \emptyset, I = \emptyset \\ 0, & \hat{I} = \emptyset, I \neq \emptyset. \end{cases} \tag{3}$$

This evaluation is methodologically correct, it captures the extent to which the interpretations of the query are identified. It does so, however, in a rather strict manner: partial matches are not given any credit. This strictness is also pointed out in [8]. Their alternative solution, albeit purely for analysis purposes, was to measure micro-averaged precision, recall, and F1-score on the entity level. That metric, on its own, is inappropriate as "entities belonging to different interpretations were mixed together" [8]. Further, by micro-averaging, the query borders are also collapsed. We propose an alternative "lean" evaluation for interpretation finding that rewards partial matches while respecting query boundaries.

Lean evaluation. Our proposal is to combine interpretation-based evaluations (cf. Equations 2 and 3) with the conventional entity linking evaluation, referred to as entity-based evaluation, from now on. Formally, entity-based evaluation is defined as follows:

$$\mathrm{P}_{\mathrm{ent}} = \begin{cases} |E \cap \hat{E}|/|E|, & E \neq \emptyset \\ 1, & E = \emptyset, \hat{E} = \emptyset \\ 0, & E = \emptyset, \hat{E} \neq \emptyset. \end{cases} \tag{4}$$

$$\mathrm{R}_{\mathrm{ent}} = \begin{cases} |E \cap \hat{E}|/|\hat{E}|, & \hat{E} \neq \emptyset \\ 1, & \hat{E} = \emptyset, E = \emptyset \\ 0, & \hat{E} = \emptyset, E \neq \emptyset. \end{cases} \tag{5}$$

We write $\hat{E}$ to denote the set of all entities from all interpretations in the ground truth, $\hat{E} = \bigcup_{j \in [1..m]} \hat{E}_j$, and E is a set of all entities from all interpretations returned by the entity linking system, $E = \bigcup_{i \in [1..n]} E_i$.

Finally, we define precision and recall as a linear combination of interpretation-based and entity-based precision and recall:

$$\mathrm{P} = \frac{\mathrm{P}_{\mathrm{int}} + \mathrm{P}_{\mathrm{ent}}}{2}, \qquad \mathrm{R} = \frac{\mathrm{R}_{\mathrm{int}} + \mathrm{R}_{\mathrm{ent}}}{2}. \tag{6}$$

For simplicity, we consider them with equal weight, but it could easily be controlled by adding a weight parameter. In all cases, the F-measure is computed according to:

$$\mathrm{F} = \frac{2 \cdot \mathrm{P} \cdot \mathrm{R}}{\mathrm{P} + \mathrm{R}}. \tag{7}$$

For computing precision, recall, and the F-measure on the whole evaluation set, an unweighed average over all queries are taken (i.e., macro-averaging is used). This provides an intuitive, easy-to-implement, and methodologically correct solution. A reference implementation is made publicly available.

4. TEST COLLECTIONS

We present two publicly available test collections for the semantic mapping and interpretation finding tasks, and introduce a new dataset for interpretation finding.

4.1 YSQLE

The Yahoo Search Query Log to Entities (YSQLE) dataset [1] comprises a selection of queries that are manually annotated with Wikipedia entities. Annotations are performed within the context of search sessions. Each annotation is aligned with the specific mention ("span") of the query. In addition, the linked entities may be labelled as *main*, to specify the intent or target of the user's query, regardless of whether the entity is mentioned explicitly in the query. For example, the query "france 1998 final" is annotated with three entities, FRANCE NATIONAL FOOTBALL TEAM, FRANCE, and 1998 FIFA WORLD CUP FINAL, of which only the last one is considered as the main annotation. Out of 2635 queries in the YSQLE dataset, 2583 are annotated with Wikipedia entities.

YSQLE is claimed to be designed for training and testing entity linking systems for queries. However, there is a number of issues. First and foremost, the dataset does not provide query interpretations, which is an essential part of entity linking in queries as we discussed in Section 3. Moreover, it is not possible to automatically form interpretation sets from the annotations. An example is the query "france world cup 1998", linked to the entities 1998 FIFA WORLD CUP, FRANCE NATIONAL FOOTBALL TEAM, and FRANCE. This query has two valid interpretations {1998 FIFA WORLD CUP, FRANCE NATIONAL FOOTBALL TEAM} and {1998 FIFA WORLD CUP, FRANCE}. One could assume that the main annotations would serve as interpretations, but it does not hold, as there exist queries with multiple or overlapping main annotations.

For example, the query "yahoo! finance," has two main annotations, linking the mention "yahoo!" to Yahoo! and the mention "yahoo! finance" to Yahoo! Finance. Second, the linked entities are not necessarily mentioned explicitly in the query, but sometimes are only being referred to. For example , the query "obama's mother" is linked to Barack Obama and Ann Dunham, where the latter is specified as the main annotation. Another example is "charlie sheen lohan," which is linked to Anger Management (TV series) and to the two actors Charlie Sheen and Lindsay Lohan. While this, in a way, is just a matter of how the annotation guidelines are defined, it nevertheless is non-standard behavior; entity linking should only be performed on explicit mentions, reference resolution is not part of the task. Carmel et al. [8] brings the query "Kobe Bryant's wife" as an example, which should be annotated as "[Kobe Bryant]'s wife." Accordingly, the "obama's mother" query should have a single interpretation, Barack Obama. Further, annotations are created by considering other queries from the session; this represents a different setting from what is discussed in Section 3.4. Lastly, the annotations in YSQLE are not always complete, meaning that some query spans that should be linked to entities are ignored. For instance the query "louisville courier journal" is annotated with The Courier Journal, whereas the link for the mention [louisville] (to Louisville, Kentucky) is missing.

In summary, even though the YSQLE dataset is intended for the purpose of entity linking in queries, in practice it is mostly suitable for the semantic mapping task. Nevertheless, it offers a great starting point; we show in Section 4.3 that with some manual effort, YSQLE can be adjusted to suit interpretation finding evaluation.

4.2 ERD

The Entity Recognition and Disambiguation (ERD) Challenge [8] introduced the first query entity linking evaluation platform that properly considers query interpretations. For each query, it contains all possible interpretations (from the pool of all participating systems). Human annotations are created in accordance with the following three rules [8]: (i) the longest mention is used for entities; (ii) only proper noun entities should be linked; (iii) overlapping mentions are not allowed within a single interpretation. A training set, consisting of 91 queries, is publicly available.[1] The ERD Challenge runs evaluation as a service; entity linking systems are evaluated upon sending a request to the evaluation server (hosted by the challenge organizers). Therefore, the test set, comprising of 500 queries, is unavailable for traditional offline evaluation. In order to make a distinction between the two query sets provided by the ERD Challenge, we refer to the former one (91 queries) as ERD-dev and to the latter one (500 queries) as ERD-test.

The ERD-dev dataset includes a small number of queries, of which only half (45 queries) are linked to entities; see Table 2. Therefore, the dataset cannot be used for training purposes and the need for a large entity linking test collection for queries still remains. In the following, we describe our new test collection, which aims to provide just that.

4.3 Y-ERD

To overcome the limitations of the YSQLE and ERD datasets, we set out to develop a test collection for interpretation finding based on YSQLE. Taking YSQLE as our starting point, we manually (re)annotated all queries following a set of guidelines (Section 4.3.2), which are based on the ERD Challenge. The application context is general web search. The resulting dataset, referred to as *Y-ERD*, contains 2398 queries in total; see the statistics in Table 2.

Query types	**Y-ERD**	**ERD-dev**
No entity	1142	46
Single entity	1133	34
Single set; >1 entity	114	7
Multiple sets	9	4
Total	2398	91

Table 2: Statistics of the interpretation finding test collections.

We further note that there is a small overlap between ERD-dev /test and Y-ERD (18 queries, to be precise). We removed those queries from Y-ERD for our experiments, so that it is possible to train systems using Y-ERD and evaluate them using ERD-dev/test.

4.3.1 From YSQLE to Y-ERD

Taking the YSQLE dataset as our input, we proceeded as follows. First, we filtered out duplicate queries. Recall that YSQLE queries are annotated within the context of search sessions and there are queries that appear in multiple sessions. We annotate queries on their own, regardless of search sessions, just like it was done at the ERD Challenge. Next, we created candidate interpretations using the following rules: (i) if the mentions are not overlapping, the linked entities form a single interpretation; (ii) if the entity mentions are identical, then each entity is considered as a separate interpretation (a set with a single element); (iii) queries that have been linked to a single entity, a single-element interpretation is created. Then, we asked three human annotators to judge these candidate query interpretations (including both finding interpretations and aligning the linked entities with the specific mention) following a set of annotation guidelines.

4.3.2 Annotation guidelines

These guidelines are based on those of the ERD Challenge [8], complemented by some additional rules:

R1 The annotated entities should be proper noun entities rather than general concepts [8]. E.g., the query "SUNY albany hospital location" is only linked to University at Albany, SUNY and the entity Location (geography) is ignored.

R2 The query should be linked to an entity via its longest mention [8]. E.g., in the query "penticton bc weather," the longest mention for the entity Penticton is "penticton bc." This implies that the term "bc" is not to be linked to British Columbia.

R3 Terms that are meant to restrict the search to a certain site (such as Facebook or IMDB) should not be linked. E.g., the entity Facebook is not linked in the query "facebook obama slur," while is it a valid annotation for the query "how to reactivate facebook."

R4 Linked entities must be explicitly mentioned in the query. One example is the query "charlie sheen lohan" that we already discussed for YSQLE in Section 4.1. For us, only Charlie Sheen and Lindsay Lohan are valid annotations. Another example is "Kurosawa's wife," which should be linked solely to the entity Akira Kurosawa, and not to Yōko Yaguchi.

R5 It could be argued either way whether misspelled mentions should be linked to entities or not. In our definition, misspellings that are recorded as name variants in DBpedia are not considered as spelling errors. We believe that annotating misspelled mentions would introduce noise into the training data. Therefore, we do not perform spell correction and not

[1] http://web-ngram.research.microsoft.com/erd2014/Datasets.aspx.

consider misspelled mentions in our ground truth in the experiments reported in this paper. Nevertheless, we also made a spell-corrected version of Y-ERD publicly available.

Based on the above rules, the assessors were instructed to: (i) identify mentions, (ii) drop invalid linked entities, (iii) change linked entities to different ones if they are a better match, (iv) complement existing interpretations with more entities. Note that by the last rule, we restrict annotators to not adding new entities to those originally identified in YSQLE, except for the case of erroneous interpretation sets. Recall that our application domain is general web search; we trust that the annotations in YSQLE include all entities that are "meaningful" in this context. One might argue that annotations are dominated by popular entities; while this may be the case, it is no different from how annotations for the ERD Challenge were performed.

4.3.3 Resolving disagreements

Regarding the interpretations (i.e., the sets of linked entities) all three annotators agreed on 84% of the queries, two agreed on 5%, and they all disagreed on the remaining 11%. For the entities linked by at least 2 assessors, the agreement on the mentions was 94%.

Disagreements were resolved through discussion, where the conflicting cases were categorized into *medium* and *hard* classes (unanimously agreed queries are regarded as *easy*). The former could be resolved through little discussion, while the latter was challenging to find agreements on. The difficulty levels are also recorded and released with the dataset.

5. METHODS

This section presents approaches for tackling the two tasks we have introduced in Section 3: semantic mapping (SM) and interpretation finding (IF). Recall that SM is the task of returning a ranked list of entities that are related to the query. IF is about finding (possibly multiple) interpretations, where an interpretation is a set of semantically related entities that are each mentioned in the query. We address both tasks in a pipeline architecture, shown in Figure 1. This pipeline is motivated by the canonical entity linking approach for documents; our components (mention detection, entity ranking, and interpretation finding) roughly correspond to the *extractor*, *searcher*, and *disambiguator* steps in traditional entity linking [20]. While this is a reasonable choice, it is certainly not the only one. We leave the exploration of alternative architectures to the future work. Notice that the first two components of the pipeline (discussed in Sections 5.1 and 5.2) are shared by the SM and IF tasks. For IF, there is an additional interpretation finding step to be performed (Section 5.3).

Before we continue, let us clarify the terminology. The term *span* refers to a query substring (n-gram). By *entity surface forms* (or *aliases*) we mean the names that are used to make reference to a particular entity. When we want to focus on a span that refers to (i.e., may be linked to) an entity, we use the term *mention*. A mention, therefore, is a pair, (m, e), where m is a span matching one of the surface forms of entity e.

5.1 Mention detection

The objective of the *mention detection* step is to identify query spans that can be linked to the entities. We view this as a recall-oriented task, as we do not want to miss any of the entities that are part of the query's interpretation(s). To identify entities mentioned in the query, we perform lexical matching for all possible n-grams in the query against known entity surface forms. (Given that web queries are typically short, this is manageable.) Surface forms are gathered from two sources: from a manually curated *knowledge base* and from machine-annotated *web corpora*.

Knowledge base. We consider known surface forms from DBpedia that are recorded under the `<rdfs:label>` and `<foaf:name>` predicates. The names of redirected entities are also included. We write A_e to denote the set of aliases for entity e. Let M_{kb} be the set of entity mentions in the query:

$$M_{kb} = \{(m, e) | \exists a \in A_e : a = m, m \in q\}, \tag{8}$$

where $m \in q$ is a query span (can be the entire query) that matches one of the aliases (a) of entity e. Because DBpedia is a high-quality resource, we do not perform any additional filtering or cleansing step on this set.

Web corpora. We make use of web-scale document collections in which entity mentions have been automatically linked to the Freebase knowledge base. Google, Inc. has recently created and made available this resource, referred to as Freebase Annotations of the ClueWeb Corpora (FACC), for the ClueWeb09 and ClueWeb12 datasets [17]. We create a dictionary of surface forms that contains the linked Freebase IDs along with frequencies and link entities to DBpedia via `<sameAs>` relations. Note that we aggregate data from both ClueWeb collections (hence the usage of "corpora").

Considering all entities that match a given surface form might leave us with a huge set of candidates; for example, "new york" matches over two thousand different entities. Therefore, we filter the set of matching candidate entities based on *commonness*. Commonness measures the overall popularity of entities as link targets [26]. Essentially, commonness is the maximum-likelihood probability that entity e is the link target of mention m:

$$\text{commonness}(m, e) = P(e|m) = \frac{n(m, e)}{\sum_{e'} n(m, e')}, \tag{9}$$

where $n(m, e)$ is the total number of times mention m is linked to entity e according to the FACC annotations. M_w refers to the set of mentions with a certain minimum commonness score:

$$M_w = \{(m, e) | m \in q, \text{commonness}(m, e) > c\}, \tag{10}$$

where the commonness threshold c is set empirically (or set to 0 if no pruning is to be performed). Using FACC as a source of entity surface forms and for a more reliable estimation of commonness scores is a novel approach; as we show later, it can warrant over 90% recall.

Combining sources. The final set of mentions is created by combining the entities identified using the knowledge base and the web annotations: $M = M_{kb} \cup M_w$.

5.2 Candidate entity ranking

We now turn to the second component of our pipeline, which ranks the entities identified by the mention detection step. Formally, this step takes a list of mention-entity pairs (m, e) as input and associates each with a relevance score. For the SM task, this ranking will constitute the final output. We note that (i) our methods are limited to returning entities explicitly mentioned in the query; this is not unreasonable (the same limitation is present, e.g., in [6]); (ii) for each entity we only consider its highest scoring mention. For IF, the resulting ranking provides input for the subsequent interpretation finding step (cf. Figure 1). As the entity relevance scores will be utilized in a later component, it is essential that they are comparable across queries. (This requirement is not unique to our approach; it would also be the case if one were to use supervised learning, for example.)

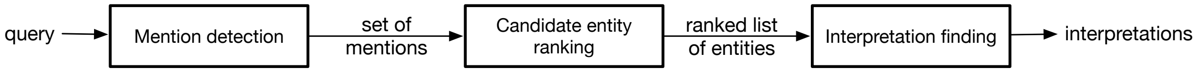

Figure 1: Pipeline for semantic mapping (first two steps) and interpretation finding (all steps).

5.2.1 *Commonness*

Commonness is shown to be a powerful baseline for the semantic mapping task [6, 28]. We follow the heuristic of looking for the longest matching mentions and ranking the corresponding entities according to their commonness score. Specifically, let l denote the length of the longest matching mention; if there are no entities mentioned in the query ($M = \emptyset$) then l is set to 0. If $l > 0$, then the matching entities are scored according to Eq. 11, otherwise no entities are returned.

$$\text{score}(e) = \max\{\text{commonness}(m, e) : |m| = l\}. \tag{11}$$

5.2.2 *Mixture of Language Models*

The predominant approach to ranking structured entity representations (typically described as a set of RDF triples) is to employ fielded extensions of standard document retrieval models, such as BM25F [5] or the Mixture of Language Models (MLM) [3, 33]. The MLM approach [36] combines language models estimated for different document fields. The model can readily be applied to ranking (document-based representations) of entities by considering different predicates as fields [23, 33]. The probability of a term t given the language model of an entity e is estimated as follows:

$$P(t|\theta_e) = \sum_{f \in F} \mu_f P(t|\theta_{e_f}), \tag{12}$$

where F is the set of possible fields, f is a specific field, μ_f is the field weight (such that $\mu_f \in [0..1]$ and $\sum_{f \in F} \mu_f = 1$), and θ_{e_f} is the field language model, which is a maximum-likelihood estimate smoothed by a field-specific background model:

$$P(t|\theta_{e_f}) = (1 - \lambda_f)\frac{n(t, e_f)}{|e_f|} + \lambda_f P(t|C_f). \tag{13}$$

Here, $n(t, e_f)$ denotes the number of occurrences of term t in field f of entity e and $|e_f|$ is the length of the field. To keep things simple, we use a single smoothing parameter for all fields: $\lambda_f = 0.1$, based on the recommendations given in [41] for title queries.

The most common approach in language modeling is to rank items (here: entities) based on query likelihood:

$$\begin{aligned} P(e|q) &= \frac{P(q|e)P(e)}{P(q)} \propto P(e)P(q|\theta_e) & (14)\\ &= P(e)\prod_{t \in q} P(t|\theta_e)^{n(t,q)}, & (15) \end{aligned}$$

where θ_e is the entity language model (defined in Eq. 12) and $n(t, q)$ denotes the number of times term t is present in query q. When a single query is considered, dropping the query probability $P(q)$ in Eq. 14 can be done conveniently. For us, however, scores (probabilities) need to be comparable across different queries, as they are utilized in the subsequent interpretation finding step (cf. Section 5.3). Therefore, the denominator, which depends on the query, should not be dropped. We perform normalization as suggested in [24] (length normalized query likelihood ratio):

$$P(e|q) = P(e)\frac{\prod_{t \in q} P(t|\theta_e)^{P(t|q)}}{\prod_{t \in q} P(t|C)^{P(t|q)}}, \tag{16}$$

where $P(t|q) = n(t, q)/|q|$ is the relative frequency of t in q. Therefore, the normalized MLM score is obtained by computing $P(t|\theta_e)$ based on Eq. 12 and $P(t|C)$ is taken to be a linear combination of collection language models $\sum_{f \in F} \mu_f P(t|C_f)$.

The specific instantiation of the model (fields and weights) is discussed in Section 6.1.

5.2.3 *Combining MLM and commonness*

Let us point out that MLM ranks entities, mentioned in the query, based on their relevance to the query. This is done irrespective of the specific surface form that is referenced in the query. There is useful prior information associated with surface forms, which is captured in commonness (Eq. 9). The commonness-based ranking method (Eq. 11), on the other hand, does not consider the query itself, which might provide additional contextual clues. It, therefore, makes good sense to combine MLM and commonness. We propose two ways of doing this.

MLMc. The first method, MLMc, simply filters the set of entities that are considered for ranking based on commonness, by applying a threshold c in Eq. 10. The setting of c is discussed in Section 6.1.

MLMcg. The second method, MLMcg, also performs filtering, exactly as MLMc does. But, in addition to that, it also integrates the commonness scores in a generative model. It ranks entities based on the highest scoring mention, i.e., ranking is dependent not only on the query but on the specific mention as well:

$$P(e|q) \propto \arg\max_{m \in q} P(e|m)P(q|e), \tag{17}$$

where $P(q|e)$ is estimated using MLM (Eq. 15) and $P(m|e)$ is the same as commonness (cf. Eq. 9). We show later experimentally that this novel method provides solid results and is more effective than MLM and MLMc.

5.3 Interpretation finding

The aim of this phase is to find the interpretations of a query, where an interpretation is a set of non-overlapping and semantically compatible entities that are mentioned in the query. Given a ranked list of mention-entity pairs from the previous step, our goal (and, as we argued, this should be the ultimate goal of entity linking in queries) is to identify all interpretations of the query.

We present an algorithm, named *Greedy Interpretation Finding* (GIF), that can detect multiple interpretations of a query; see Algorithm 1. It takes as input a list of mention-entity pairs (m, e), each associated with a relevance score. Consider an example query "jacksonville fl," for which the input for the algorithm would be {("jacksonville fl', Jacksonville Florida): 0.9, ("jacksonville", Jacksonville,Florida): 0.8, ("jacksonville fl", Naval Air Station Jacksonville): 0.2}. In the first step (line 1), GIF prunes entities based on absolute scores, controlled by the threshold parameter s. E.g., with a threshold of 0.3, ("jacksonville fl", Naval Air Station Jacksonville) would be filtered out here. We note that s is a global parameter, therefore ranking scores must be comparable across queries. (As mentioned in the previous section, our query length normalized ranking scores enable this.) In the next step (line 2), containment mentions are also filtered out, based on their retrieval scores. E.g., out of the two containment mentions "jacksonville fl" and "jacksonville", only the pair ("jacksonville fl", Jacksonville Florida) with the score of 0.9 is

Algorithm 1 Greedy Interpretation Finding (GIF)

```
Input:  Ranked list of mention-entity pairs $M$; score threshold $s$
Output: Interpretations $I = \{E_1, ..., E_m\}$

begin
 1: $M' \leftarrow$ Prune($M, s$)
 2: $M' \leftarrow$ PruneContainmentMentions($M'$)
 3: $I \leftarrow$ CreateInterpretations($M'$)
 4: return $I$
end
 1: function CREATEINTERPRETATIONS($M$)
 2:     $I \leftarrow \{\emptyset\}$
 3:     for $(m, e)$ in $M$ do
 4:         $h \leftarrow 0$
 5:         for $E$ in $I$ do
 6:             if $\neg$ hasOverlap($E$, $(m, e)$) then
 7:                 $E$.add($(m, e)$)
 8:                 $h \leftarrow 1$
 9:             end if
10:         end for
11:         if $h == 0$ then
12:             $I$.add($\{(m, e)\}$)
13:         end if
14:     end for
15:     return $I$
16: end function
```

kept. Then (in line 3), query interpretations are created in an iterative manner: adding an entity-mention pair to an existing interpretation E, such that it does not overlap with the mentions already present in E. In case it overlaps with all existing interpretations, the mention-entity pair constitutes a new interpretation; this will result in multiple interpretations for a query. The implementation of GIF is made publicly available.

6. EXPERIMENTS

In this section we present results for the semantic mapping and interpretation finding tasks, using the test collections introduced in Section 4 and the methods presented in Section 5.

6.1 Experimental setup

Knowledge base. We consider entities present in both DBpedia and Freebase as our reference knowledge base. This choice is made for pragmatic reasons: (i) existing test collections provide annotations (or ground truth) either for one or the other, (ii) Freebase-annotated ClueWeb collections (FACC) [17] are leveraged for mention detection and (reliable) commonness estimation, (iii) entity descriptions in DBpedia provide a solid basis for entity ranking.

Entity ranking. For ranking entities using MLM, we followed Neumayer et al. [34] and used an index with two fields, name and content, with a weight of 0.2 and 0.8, respectively. The *name* field holds the primary names of the entity (`<rdfs:label>`, `<foaf:name>`) and name variants extracted from redirected entities. The *content* field includes the content of the top 1000 most frequent predicates across the whole DBpedia collection. All URIs in the content fields are resolved, i.e., replaced with the name of the entity or title of the page they point to. The index is confined to the entities having a name and a short abstract (i.e., `<rdfs:label>` and `<rdfs:comment>`).

Semantic mapping. The SM task is evaluated on the YSQLE test collection. We compare commonness (CMNS), MLM, MLMc, and MLMcg from Section 5.2 and also include results for the TAGME system [16]. The commonness threshold c for MLMc and MLMcg (Eq. 10) is set to 0.1 by performing a sweep using cross-validation. For CMNS, we use FACC to compute commonness.

	YSQLE	**Y-ERD**	**ERD**
KB	0.7489	0.7976	0.8556
Web	0.9127	0.9716	0.9956
KB+Web	**0.9163**	**0.9724**	**1.0000**

Table 3: Recall of different sources for mention detection.

	MAP	**S@1**	**MRR**
CMNS	0.6334	0.5751	0.6442
MLM	0.4582	0.3601	0.4638
MLMc	0.6228	0.5413	0.6312
MLMcg	**0.7078**	**0.6403**	**0.7151**
TAGME†	0.6230	0.6016	0.6385

†TAGME is an entity linking system and should not be evaluated on the semantic mapping task using rank-based metrics.

Table 4: Semantic mapping results on the YSQLE dataset.

Interpretation finding. For the IF task, we report our results on the Y-ERD and ERD-dev test collections. In this case, our reference knowledge base is confined to the entities present in the knowledge base snapshot used at the ERD Challenge [8]. This snapshot contains 2,351,157 entities; taking its intersection with DBpedia resulted in the removal of 39,517 entities.

The GIF method (see Section 5.3) is applied on top of the four candidate entity ranking systems. We use cross-validation (5-fold for Y-ERD and leave-one-out for ERD-dev) for setting the score threshold of GIF, by performing a sweep for the parameter s. In addition, we report on two baselines. The first, called *TopRanked*, uses the best performing entity ranking approach (MLMcg) and forms a single interpretation set from the top ranked entity. The second baseline is TAGME.

TAGME. We report on TAGME [16], a state-of-the-art entity linking system for short texts. Even though TAGME is available through an API, we used our own implementation, given that our reference knowledge base is different. Specifically, we used a Wikipedia dump from June 16, 2015 to extract commonness and link probability and followed [9] in implementing semantic relatedness [32].

6.2 Mention detection

The mention detection component is shared by both the semantic mapping and interpretation finding tasks, therefore we evaluate it on its own account. Specifically, we compare three options based on the source(s) of surface forms, as described in Section 5.1: (i) DBpedia (KB), (ii) web corpora (Web), and (iii) the combination of both (KB+Web). As this step is recall oriented, (i.e., all entity matches should be retrieved), we only report on recall.

Table 3 presents the results. We find that the machine-annotated web corpora provides a rich source of entity surface forms for this task and is a better source than DBpedia alone. Not surprisingly, the combination of the two sources yields the highest recall, albeit the improvement over Web is marginal. We also note that while recall is nearly perfect on the interpretation finding datasets (Y-ERD and ERD), it is a bit lower for YSQLE. Recall that YSQLE is created for evaluating the semantic mapping task, where implicit entity mentions are also considered as relevant; these are not captured by our dictionary-based mention detection approach and would need to be identified by different means.

Method	Strict eval.			Lean eval.		
	P	**R**	**F**	**P**	**R**	**F**
TopRanked	0.4554	0.4542	0.4545	0.4771	0.465	0.4689
TAGME	0.6647	0.6642	0.6643	0.6821	0.6853	0.6815
GIF-CMNS	0.6927	0.6938	0.6929	0.7093	0.7072	0.7062
GIF-MLM	0.5259	0.5254	0.5255	0.5363	0.5387	0.5361
GIF-MLMc	0.6351	0.6354	0.6348	0.6422	0.642	0.6409
GIF-MLMcg	**0.7191**	**0.7213**	**0.7195**	**0.7305**	**0.7308**	**0.7288**

Table 5: Interpretation finding on the Y-ERD dataset.

Method	Strict eval.			Lean eval.		
	P	**R**	**F**	**P**	**R**	**F**
TopRanked	0.3846	0.3645	0.3700	0.4231	0.3837	0.3956
TAGME	0.7143	0.7015	0.7051	**0.7418**	**0.7372**	**0.7333**
GIF-CMNS	0.5824	0.5824	0.5824	0.6071	0.5962	0.5998
GIF-MLM	0.5824	0.5608	0.5659	0.5934	0.5718	0.5760
GIF-MLMc	0.7253	0.7037	0.7088	0.7445	0.7174	0.7234
GIF-MLMcg	**0.7143**	**0.7125**	**0.7114**	0.7335	0.7262	0.7260

Table 6: Interpretation finding on the ERD-dev dataset.

6.3 Semantic mapping

Table 4 presents the results for semantic mapping. Given that this is a ranking task, we report on mean average precision (MAP), success at position 1 (S@1), and mean reciprocal rank (MRR). Though we include results for TAGME, we note that this comparison, despite having been done in prior work (e.g., in [6]), is an unfair one. TAGME is an entity linking system that should not be evaluated using rank-based metrics. The very reason we include TAGME is to illustrate that one can easily achieve improvements over a state-of-the-art entity linking system on the semantic mapping task, but those claims would be false and misleading. We find that MLMcg is the most effective method; it shows that incorporating commonness in a generative model (MLMcg) is better than using commonness alone (CMNS) or as a filter before ranking entities (MLMc).

6.4 Interpretation finding

Tables 5 and 6 present the results for interpretation finding on the Y-ERD and ERD-dev datasets, respectively. Two sets of evaluation metrics are used: (i) strict (which is the same as in [8]) and (ii) lean (Section 3.4). We notice at first glance that the TopRanked baseline is considerably worse than the other approaches. This shows that, even though there are many queries containing a single entity in our data sets (cf. Table 2), forming sets of entities is a crucial aspect of the interpretation finding task. The GIF algorithm in combination with MLMcg delivers solid performance and is the best performing of all approaches in all but one setting. In comparison with TAGME, GIF performs substantially better on the Y-ERD dataset, while being in par with TAGME on ERD-dev. While our focus was not on efficiency, we also note that GIF has a considerably lower response time than TAGME. GIF utilizes the retrieval scores of the mentioned entities and involves virtually no computation. Additionally, it is able to generate multiple query interpretations. This makes it the preferred alternative over TAGME for entity linking in queries. Comparing the two evaluation metrics, lean evaluation gives higher results for all systems. This is in line with our expectations based on the theoretical definitions of the metrics (Section 5.3); when an interpretation is incomplete, yet contains relevant entities, the strict evaluation does not give any credit for returning correct entities, whereas the lean one does. We note that for the Y-ERD experiments, we also tried to use the spell-corrected queries (with the corresponding qrels), but no considerable differences were observed.

7. DISCUSSION

We now answer our research questions, and subsequent subquestions, based on the results presented in Section 6.

How should the inherent ambiguity of entity annotations in queries be handled? Entity linking in queries should ultimately be addressed as an *interpretation finding* task, where an interpretation is a set of non-overlapping entities that are semantically related to each other. If the query is ambiguous, with little or no context, there exist multiple interpretations, all of which should be found. Otherwise, a single interpretation should be detected, which is similar to the traditional entity linking task for documents. Determining when the query has no interpretations (in terms of entity annotations) is also a crucial part of the problem that should be addressed (and considered in the evaluation). The *semantic mapping* task (Section 3.3), which ranks entities based on their relevance to the query, serves a different purpose and should be considered as an entity linking task, even for the simplified scenario of finding a single interpretation. This is because relevant entities can be overlapping and are not required to be semantically related to each other. Furthermore, entity disambiguation is an essential part of entity linking, an aspect that is completely ignored in semantic mapping. A number of earlier studies refer to entity linking, while what they do in fact is semantic mapping [6, 27, 28, 35]. Comparing semantic mapping to results generated by traditional entity linking methods is inappropriate (cf. Section 6.3).

What are the similarities and differences between semantic mapping and interpretation finding in terms of approaches? The semantic mapping and interpretation finding tasks can be addressed by using a similar pipeline architecture (see Section 5). Both tasks share the first component, mention detection, which can effectively be addressed by combining surface forms stored in knowledge bases and extracted from a large machine-annotated web corpora (cf. Section 6.2). The second component, which generates a ranked list of the mentioned entities based on their relevance to the query, can also be shared. One issue that requires special attention is the question of implicit mentions, that is, entities that are referred to but not explicitly mentioned in the query (e.g., “Obama's mother”). These are not identified by the mention detection step and consequently not considered for ranking either. One interesting research challenge in semantic mapping is finding these referred entities. For interpretation finding, the third component is responsible for forming (possibly multiple) sets of entities. This is a highly nontrivial subtask that makes interpretation finding substantially more difficult than semantic mapping.

What are appropriate evaluation methodology and metrics? For interpretation finding, similar to the traditional entity linking task [16, 31, 32], evaluation uses set-based metrics (precision, recall and F-measure). However, since the output is a set of interpretations (and not entities) the evaluation methodology is different. The method presented in [8] considers the exact match between the retrieved sets and the ground truth, which is rather strict. The lean evaluation method (Section 5.3), on the other hand, combines interpretation-based and entity-based evaluations. For semantic mapping, standard rank-based metrics (MAP, MRR, S@1) can be employed.

As most queries have a single interpretation, how much effort should be expedited to find multiple interpretations? Although having multiple interpretations is an intrinsic feature of entity linking in queries, most of the queries in our test collections (both ERD and Y-ERD) have a single interpretation (see Table 2). This implies that a system can achieve high overall score by focusing on returning a single interpretation. This is also evidenced by the ERD Challenge results, where the top two performing systems [10, 12] return a single interpretation. We note that returning multiple interpretations, without hurting queries with single a interpretation, is an open research question.

8. CONCLUSIONS

In this paper we have addressed fundamental questions in the problem area of entity linking in queries. We have differentiated between two tasks, semantic mapping and interpretation finding. The former ranks entities that are related to (but not necessarily explicitly mentioned in) the query, while the latter aims to identify sets of semantically related entities that are mentioned in the query, and is able to return more than one of such sets if the query has multiple interpretations. We have discussed evaluation methodology and carefully examined publicly available test collections for both tasks, and introduced a large, manually curated test collection for interpretation finding. Technical contributions of this study include methods for effectively addressing these tasks, accompanied by a set of results. One obvious direction for further work is to evaluate the retrieval impact of entity-annotated queries.

References

[1] Yahoo! Webscope L24 dataset - Yahoo! search query log to entities, v1.0. URL http://webscope.sandbox.yahoo.com/.

[2] A. Alasiry, M. Levene, and A. Poulovassilis. Detecting candidate named entities in search queries. In *Proc. of SIGIR '12*, pages 1049–1050, 2012.

[3] K. Balog and R. Neumayer. A test collection for entity search in DBpedia. In *Proc. of SIGIR '13*, pages 737–740, 2013.

[4] S. Bergsma and Q. I. Wang. Learning noun phrase query segmentation. In *Proc. of EMNLP-CoNLL '07*, pages 819–826, 2007.

[5] R. Blanco, P. Mika, and S. Vigna. Effective and efficient entity search in RDF data. In *Proc. of ISWC '11*, pages 83–97, 2011.

[6] R. Blanco, G. Ottaviano, and E. Meij. Fast and space-efficient entity linking for queries. In *Proc. of WSDM '15*, pages 179–188, 2015.

[7] H. Cao, D. H. Hu, D. Shen, D. Jiang, J.-T. Sun, E. Chen, and Q. Yang. Context-aware query classification. In *Proc. of SIGIR '09*, pages 3–10, 2009.

[8] D. Carmel, M.-W. Chang, E. Gabrilovich, B.-J. P. Hsu, and K. Wang. ERD'14: Entity recognition and disambiguation challenge. *SIGIR Forum*, 48(2):63–77, 2014.

[9] D. Ceccarelli, C. Lucchese, S. Orlando, R. Perego, and S. Trani. Learning relatedness measures for entity linking. In *Proc. of CIKM'13*, pages 139–148, 2013.

[10] Y.-P. Chiu, Y.-S. Shih, Y.-Y. Lee, C.-C. Shao, M.-L. Cai, S.-L. Wei, and H.-H. Chen. NTUNLP approaches to recognizing and disambiguating entities in long and short text at the ERD challenge 2014. In *Proc. of Entity Recognition & Disambiguation Workshop*, pages 3–12, 2014.

[11] M. Cornolti, P. Ferragina, and M. Ciaramita. A framework for benchmarking entity-annotation systems. In *Proc. of WWW '13*, pages 249–260, 2013.

[12] M. Cornolti, P. Ferragina, M. Ciaramita, H. Schütze, and S. Rüd. The SMAPH system for query entity recognition and disambiguation. In *Proc. of Entity Recognition & Disambiguation Workshop*, pages 25–30, 2014.

[13] W. B. Croft, M. Bendersky, H. Li, and G. Xu. Query representation and understanding workshop. *SIGIR Forum*, 44(2):48–53, 2011.

[14] S. Cucerzan. Large-scale named entity disambiguation based on Wikipedia data. In *Proc. of EMNLP-CoNLL '07*, pages 708–716, 2007.

[15] A. Eckhardt, J. Hreško, J. Procházka, and O. Smrž. Entity linking based on the co-occurrence graph and entity probability. In *Proc. of Entity Recognition & Disambiguation Workshop*, pages 37–44, 2014.

[16] P. Ferragina and U. Scaiella. TAGME: On-the-fly annotation of short text fragments (by Wikipedia entities). In *Proc. of CIKM '10*, pages 1625–1628, 2010.

[17] E. Gabrilovich, M. Ringgaard, and A. Subramanya. FACC1: Freebase annotation of ClueWeb corpora, Version 1 (Release date 2013-06-26, Format version 1, Correction level 0), 2013. URL http://lemurproject.org/clueweb12/.

[18] J. Guo, G. Xu, X. Cheng, and H. Li. Named entity recognition in query. In *Proc. of SIGIR '09*, pages 267–274, 2009.

[19] S. Guo, M.-W. Chang, and E. Kiciman. To link or not to link? a study on end-to-end tweet entity linking. In *Proc. of HLT-NAACL*, pages 1020–1030, 2013.

[20] B. Hachey, W. Radford, J. Nothman, M. Honnibal, and J. R. Curran. Evaluating entity linking with Wikipedia. *Artif. Intell.*, 194:130–150, 2013.

[21] M. Hagen, M. Potthast, B. Stein, and C. Bräutigam. Query segmentation revisited. In *Proc. of WWW '11*, pages 97–106, 2011.

[22] J. Hu, G. Wang, F. Lochovsky, J.-t. Sun, and Z. Chen. Understanding user's query intent with Wikipedia. In *Proc. of WWW '09*, pages 471–480, 2009.

[23] J. Kim, X. Xue, and W. B. Croft. A probabilistic retrieval model for semistructured data. In *Proc. of ECIR '09*, pages 228–239, 2009.

[24] W. Kraaij and M. Spitters. Language models for topic tracking. In *Language Modeling for Information Retrieval*, pages 95–123, 2003.

[25] X. Li, Y.-Y. Wang, and A. Acero. Learning query intent from regularized click graphs. In *Proc. of SIGIR '08*, pages 339–346, 2008.

[26] O. Medelyan, I. H. Witten, and D. Milne. Topic indexing with Wikipedia. In *Proc. of the AAAI WikiAI workshop*, pages 19–24, 2008.

[27] E. Meij, M. Bron, L. Hollink, B. Huurnink, and M. de Rijke. Mapping queries to the Linking Open Data cloud: A case study using DBpedia. *Web Semant.*, 9(4):418–433, 2011.

[28] E. Meij, W. Weerkamp, and M. de Rijke. Adding semantics to microblog posts. In *Proc. of WSDM '12*, pages 563–572, 2012.

[29] E. Meij, K. Balog, and D. Odijk. Entity linking and retrieval for semantic search. In *Proc. of WSDM '14*, pages 683–684, 2014.

[30] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. DBpedia spotlight: Shedding light on the web of documents. In *Proc. of I-Semantics '11*, pages 1–8, 2011.

[31] R. Mihalcea and A. Csomai. Wikify!: Linking documents to encyclopedic knowledge. In *Proc. of CIKM '07*, pages 233–242, 2007.

[32] D. Milne and I. H. Witten. Learning to link with Wikipedia. In *Proc. of CIKM '08*, pages 509–518, 2008.

[33] R. Neumayer, K. Balog, and K. Nørvåg. On the modeling of entities for ad-hoc entity search in the web of data. In *Proc. of ECIR '12*, pages 133–145, 2012.

[34] R. Neumayer, K. Balog, and K. Nørvåg. When simple is (more than) good enough: Effective semantic search with (almost) no semantics. In *Proc. of ECIR '12*, pages 540–543, 2012.

[35] D. Odijk, E. Meij, and M. de Rijke. Feeding the second screen: Semantic linking based on subtitles. In *Proc. of OAIR '13*, pages 9–16, 2013.

[36] P. Ogilvie and J. Callan. Combining document representations for known-item search. In *Proc. of SIGIR '03*, pages 143–150, 2003.

[37] G. Rizzo, M. van Erp, and R. Troncy. Benchmarking the extraction and disambiguation of named entities on the semantic web. In *Proc. of LREC '14*, 2014.

[38] D. E. Rose and D. Levinson. Understanding user goals in web search. In *Proc. of WWW '04*, pages 13–19, 2004.

[39] D. Shen, J.-T. Sun, Q. Yang, and Z. Chen. Building bridges for web query classification. In *Proc. of SIGIR '06*, pages 131–138, 2006.

[40] B. Tan and F. Peng. Unsupervised query segmentation using generative language models and Wikipedia. In *Proc. of WWW '08*, pages 347–356, 2008.

[41] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2): 179–214, 2004.