Semanticizing Search Engine Queries

The University of Amsterdam at the ERD 2014 Challenge

David Graus
University of Amsterdam
d.p.graus@uva.nl

Daan Odijk
University of Amsterdam
d.odijk@uva.nl

Manos Tsagkias
University of Amsterdam
e.tsagkias@uva.nl

Wouter Weerkamp
904Labs
wouter@904labs.com

Maarten de Rijke
University of Amsterdam
derijke@uva.nl

ABSTRACT

This paper describes the University of Amsterdam's participation in the short track of the Entity Recognition & Disambiguation Challenge 2014 (ERD 2014). We describe how we adapt the *Semanticizer*—an open-source entity linking framework developed primarily at the University of Amsterdam—to the task of the ERD challenge: linking named entities in search engine queries. We steer the Semanticizer's linking towards named entities by adapting an existing training corpus, and extend the Semanticizer's set of features with contextual features that aim to leverage the limited context provided by search queries. With an F1 score of 0.6062 our final system run achieves median performance, and better than mean performance (0.5329).

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.3 Information Search and Retrieval

Keywords

Entity linking; semantic search; knowledge base; Wikipedia; data fusion

1. INTRODUCTION

Entity linking addresses the task of identifying and disambiguating entity occurrences in unstructured text, effectively linking an entity mention in a document to an entity in a knowledge base. Entity linking is a key component in modern-day applications such as semantic search and advanced user interfaces, and it plays a major role in accessing and populating the Web of Data. It can also help to improve NLP tasks [3], or to "anchor" a piece of text in background knowledge; authors or readers may find entity links to supply useful pointers [6]. Another application can be found in search engines, which increasingly support directly linking queries to entities and presenting entity-specific overviews [1]. As a full overview of the various approaches in entity linking is not in the scope of this paper, we refer readers to overview papers such as [13].

ERD'14, July 11, 2014, Gold Coast, Queensland, Australia.

ACM 978-1-4503-3023-7/14/07 ...$15.00.
http://dx.doi.org/10.1145/2633211.2634354

Our participation in the Entity Recognition & Disambiguation Challenge 2014 (ERD 2014) [2] centers around adapting existing technology for linking to Wikipedia to the task at hand. Our system's main component is the *Semanticizer*,[1] an open-source entity linking framework developed primarily at the University of Amsterdam. The Semanticizer was mainly developed for linking entities in short noisy texts, such as tweets, in batch [7] and streaming scenarios [12], and later it was further profiled to cope with additional types of data, such as television subtitles [8], where it has also proven successful. In the remainder of this paper we describe how we adapt it to linking named entities in another type of short text data, namely, search engine queries.

2. TASK

The short track of the ERD challenge involves linking mentions of named entities in search engine queries to entities in a custom Knowledge Base (KB). This KB consists of a subset of Freebase entities, that have English Wikipedia pages associated with them. The entities are further restricted to named entities or proper noun phrases. This sets the ERD challenge's task apart from so-called *Wikification*, where the target is to link any Wikipedia concept that is mentioned. In summary, the ERD challenge task is that of *linking named entities in search engine queries to Wikipedia.*

We model this task as follows; Given a search engine query q, we (1) identify the set of entity mentions $M = \{m_1, m_2, \ldots, m_n\}$, and; (2) link these to a set of KB entities $E = \{e_1, e_2, \ldots, e_n\}$, and finally; (3) assign the entities to interpretation sets. The latter point addresses the aspect of the ERD task that a single (ambiguous) entity mention m can be linked to multiple entities. To illustrate, in the query `[Michael Jordan basketball]` the correct interpretation is restricted to a single entity: the basketball player.[2] However, in the query `[Michael Jordan]`, the lack of context means we cannot rule out any Michael Jordan, so there are multiple correct interpretations. In this case we return all Michael Jordans in our KB (including, e.g., the football player[3]) and assign them to distinct *interpretation sets*.

3. METHOD

The core of our participation consists of adapting the Semanticizer to the short text track of ERD 2014. In what follows we briefly describe the Semanticizer (in Section 3.1) and our approach to adapting the Semanticizer to linking named entities in search en-

[1] http://semanticize.uva.nl
[2] http://en.wikipedia.org/wiki/Michael_Jordan
[3] http://en.wikipedia.org/wiki/Michael_Jordan_(footballer)

Table 1: Features used in the Semanticizer.

***mention* features**	
LEN	Number of terms in *mention*
IDF_anchor	Inverse Document Frequency of *mention* in all Wikipedia anchors
IDF_content	Inverse Document Frequency of *mention* in all Wikipedia pages
IDF_title	Inverse Document Frequency of *mention* in all Wikipedia titles
LINKPROB	Probability that *mention* is used as an anchor text in Wikipedia (all occurrences)
SNCL	Number of articles whose title match a sub-n-gram of *mention*
SNIL	Number of articles whose title equals a sub-n-gram of *mention*
***entity* features**	
INLINKS	Number of pages linking to *entity*
OUTLINKS	Number of pages *entity* links to
REDIRECT	Number of redirect pages linking to *entity*
***mention-entity* features**	
TF_paragraph	Frequency of *mention* in normalized first paragraph of *entity*'s document
TF_sentence	Frequency of *mention* in normalized first sentence of *entity*'s document
TF_title	Frequency of *mention* in normalized title of *entity*'s document
POS	Position of the first occurence in the first paragraph of *entity*'s document
NCT	True if *mention* contains the title of *entity*
TCN	True if title of *entity* contains *mention*
TEN	True if title of *entity* equals *mention*
CMNS	Probability of *entity* being target of link with *mention* as anchor
LD	Levenshtein Distance between *mention* and title of *entity*
NORM	Levenshtein Distance between normalized *mention* and *entity*
SS_MATCH_1	True if the title of *entity* begins with *mention*
SS_MATCH_2	True if the title of *entity* ends with *mention*
W_MATCH	Number of shared words between title of *entity* and *mention*
W_MISS	Number of different words between title of *entity* and *mention*
***contextual* features**	
DOCSIM	(Normalized) retrieval score of q to *entity* content
LINKSSIM	(Normalized) retrieval score of q to *entity* virtual document (titles of linked pages)

gine queries (in Section 3.2). Finally, in Section 3.3 we describe how we incorporate the Semanticizer in a pipeline that performs the end-to-end task.

3.1 The Semanticizer

The Semanticizer is a lexical-matching-based linking-to-Wikipedia framework, based on the semantic linking in microblog posts method proposed in [7]. Since then, it has been re-implemented in Python for semantic linking subtitles [8] and later published under the GNU Lesser General Public License.[4]

The framework has been applied to the entity linking track of the Text Analysis Conference (TAC) Knowledge Base Publication task in 2012 [4]. It has since been applied to domains that show similar challenges as that of search engine queries, e.g., on Twitter, for generating ground truth for discovering new entities [5], and for semantic expansion of tweets for summarization [10].

The Semanticizer follows a typical approach for entity linking consisting of two steps: (i) candidate generation, and (ii) entity disambiguation. For **candidate generation**, it leverages lexical matching of known surface forms of Wikipedia pages (also referred to as mention detection) to generate entity candidates for detected entity mentions. Because of its independence from POS tagging or information extraction methods such as named-entity recognition, the Semanticizer is in principle language and domain independent. For **entity disambiguation**, it incorporates a supervised (binary) classifier which labels entities as target entities (being mentioned in an input text) and non-target entities for either the task of entity disambiguation or linking.

To adapt the Semanticizer to the ERD 2014 challenge, we introduce several changes. The first addresses the ERD 2014 challenge's target entities; as the Semanticizer in principle links any Wikipedia concept mentioned in the input text, we need to adapt it to link only *named entities*. We address this by leveraging the Semanticizer's supervised learning component, and use an existing training corpus to steer the Semanticizer's linking towards named entities. Furthermore, we extend its set of features for supervised classification with a set of features that leverage the (little) context that search engine queries provide.

3.2 Supervised Entity Linking

To classify entity candidates into target and non-target entities, our classifier leverages several features, described in Table 1.

We distinguish between four "families" of features: those that pertain solely to the detected mention (*mention*), solely to the candidate entity (*entity*), or to the combination of mention and entity (*mention* − *entity*). These features were proposed by Meij et al. [7] and Odijk et al. [8], for linking microblog posts and subtitles respectively. For the ERD 2014 challenge, we extend these features with *contextual* features, which aim to leverage the (little) context that is provided in search engine queries. Due to both the limited textual context which is typically found in search engine queries, and the constraint on execution time (a system ought to return results in less than 20 seconds), we opt to exclude global entity linking methods that aim to disambiguate multiple entity mentions simultaneously [9].[5] However, we do incorporate a contextual feature that approximates a global approach (LINKSSIM); we measure the similarity between the query q and a virtual document created by appending the titles of all linked pages (i.e., related Wikipedia concepts). In a sense, this virtual document represents the entities that are related to the candidate entity, and thus a mention of a related entity in the query will result in a higher ranking of the entity candidate.

To train the supervised entity linker, we turn to the Yahoo! Webscope dataset: *ydata-search-query-log-to-entities-v1_0*.[6] This dataset is a natural fit to the ERD task, as its domain is identical to that of the ERD 2014 challenge: search engine queries (in the case of Yahoo!'s dataset, annotated with Wikipedia concepts). However,

[4] https://github.com/semanticize/semanticizer

[5] This feature is available in the Semanticizer as "related entity search algorithms" [4].

[6] http://labs.yahoo.com/Academic_Relations

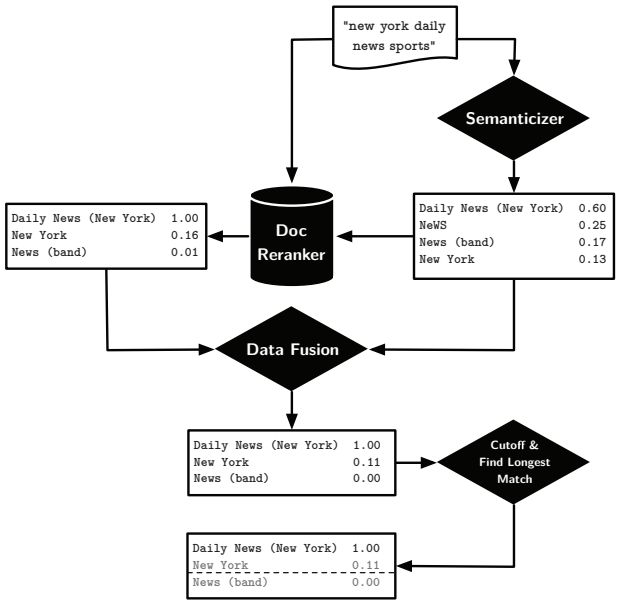

Figure 1: A schematic view of our system pipeline. A query is given as input to the Semanticizer for generating a ranked list of candidate entities using several methods. The candidate list is refined using a document re-ranker and then all ranked lists are merged together using late data fusion. The final entities for the query are selected using cutoff on their retrieval score, and longest matches are preferred over shorter ones.

since the ERD challenge's focus is on named entities, we need to fit the training data better to the task at hand. To address this, we manually removed all non-named entities from the dataset (e.g., the Wikipedia concept `Suffering`[7] in query `[ditka suffers stroke]`). In addition, to minimize the data imbalance that is common in entity linking as binary classification (i.e., a high number of negative samples with a low number of positive ones), we removed all queries that contain no positive examples (i.e., no named entities). Our final classifier is trained on 2,583 search engine queries, with a total of 15,699 labeled samples (of which 5,125 are positive examples).

3.3 System

In what follows we describe the final system that we employ for the ERD challenge; we refer the reader to Figure 1 for an illustration of the components of our system. We describe the data flow and each of these components in detail, below.

3.3.1 Semanticizer

The Semanticizer consists of two components: (i) candidate generation, and (ii) supervised disambiguation.

Candidate generation. The first step is to extract all candidate entity mentions from the query, along with their candidate entities. The semanticizer employs a lexical-matching approach and retrieves candidate mentions and entities by matching known surface forms of Wikipedia pages in the query. These surface forms consist of all anchor texts that refer to Wikipedia pages (in Wikipedia), disambiguation and redirect pages, and page titles.

This first candidate generation step is focused on achieving high recall, however, there is a tradeoff between high recall and efficiency. Since our trained classifier is aimed at increasing precision, i.e., weeds out unlikely entity candidates at a later stage, the fewer candidates we generate, the less features we have to extract, and the quicker the linking will be.

To balance this recall to efficiency we apply a preprocessing filter to prune unlikely entity candidates. The signal we employ for this initial pruning is *senseProbability*, a prior probability proposed in [8] that estimates how likely it is for an n-gram (entity mention) to refer to an entity candidate. It is derived from frequencies of known surface forms in Wikipedia (e.g., by counting the number of times the mention "Michael" is in an anchor text that links to the entity *Michael_Jackson*).

We set a lenient threshold on this *senseProbability* for the candidate generation. As an example, consider the query `[richard chamberlain death]`. Without threshold it yields a total of 636 entity candidates, of which 427 are linked to the mention `richard`, 172 to `death`, 35 to `chamberlain`, and only two to `richard chamberlain`. When we set a threshold on the retrieval score of a candidate entity, i.e., 0.01, the number of candidates drops from 636 to four, which substantially helps efficiency without hurting effectiveness. Continuing on our example, in this set of four candidates, both Richard Chamberlains[8] are included. As was to be expected, in our development period we found that increasing the threshold would decrease response time, at the cost of recall.

Supervised disambiguation. For each entity candidate, the Semanticizer extracts features and assigns a probability for it being a target entity (i.e., being mentioned in the query). To this end, we use a Random Forest classifier, trained on the adapted Yahoo! Webscope dataset (see Section 3.2).

3.3.2 Document re-ranker

In the next step, we leverage the available context in the query by including a *document re-ranking* step. The re-ranker compares the context of the entity mention (which we set to be the query q) to the documents that represent the set of generated entity candidates (i.e., the full text Wikipedia page). The intuition is that we want to assign higher scores to entity candidates whose Wikipedia pages are more similar to query q.

To perform this re-ranking, we use an ElasticSearch index with the Wikipedia river plugin.[9] Our system issues the query q to the Wikipedia index and returns the retrieval score for the entities that are in the set of candidate entities (returned by the Semanticizer). We then normalize the retrieval scores and yield an additional ranking of the (subset of) initial entity candidates.

3.3.3 Data Fusion

Given both lists of ranked entities (the candidate generation list and the document re-ranker list), we merge both lists using the *combMNZ* data fusion algorithm. We have tried other variants of the *combSUM* family during our development period, and we have found that *combMNZ* performs the best.

3.3.4 Postprocess: Pruning

The final step consists of pruning, that is, removing unlikely entities. These are entities whose mention is contained within another entity mention, and entities that are partially overlapping with other, longer entity mentions.

The initial pruning is done by applying a cutoff on the normalized retrieval scores that are returned by the data fusion step. To

[7] http://en.wikipedia.org/wiki/Suffering

[8] http://en.wikipedia.org/wiki/Richard_Chamberlain and http://en.wikipedia.org/wiki/Richard_Chamberlain_(politician).

[9] https://github.com/elasticsearch/elasticsearch-river-wikipedia

Table 2: Ranking of participating systems for the ERD 2014 challenge. We mark our system (UvA) with boldface.

Rank	Team	F1
1	SMAPH Team	0.6858
2	NTUNLP	0.6797
3	Seznam Research	0.6693
4	Magnetic_IISAS	0.6557
5	WebSAIL	0.6414
6	UBC	0.6369
7	C3	0.6194
8	ExPoSe	0.6170
9	CI_ERD	0.6130
10	**UvA**	**0.6062**
11	InriaBerlin	0.5786
12	XI-lab	0.5182
13	NTNU-UiS	0.4947
14	whu_ir	0.4864
15	TALP-UPC	0.4508
16	UIUC-GSLIS	0.4003
17	clues_ERD	0.3642
18	SIEL@ERD	0.2040
19	Pasquale	0.2040

determine the optimal cutoff point, we compare several methods in our development period. We explore **top-*n*** approaches (with several values for n), a **static** threshold (on normalized entity candidate scores), and finally a pruning strategy that cuts off the ranking at the largest **delta** between two subsequent entities (excluding the first delta, i.e., the difference between the first and second entity candidates). This latter approach was found to be optimal in our development period.

After this initial pruning, we further remove mentions that are contained in other mentions, and those that are partially overlapping with longer mentions. When two mentions have an exact match, we assign the corresponding entities into distinct interpretation sets, using a naive approach where we increment the interpretation set identifier for each overlapping entity mention.

4. RESULTS

At the time of writing, it was not possible to compare our final system run to additional submitted runs. Extensive results were not published other than the final standings, as reflected in Table 2. As can be seen, with a 10th place out of 19 participating teams, our adapted version of the Semanticizer achieves an F1 score of 0.6062. In terms of system ranking, our system ranks close to the middle of all participants. In terms of effectiveness, our performance outscores the mean performance (F1 0.5329) of all participating systems (statistically significant with our α-level set at 0.05).

With an average latency of 3.3 seconds our response time falls well within the timeout on responses (at 20 seconds), but is higher than a baseline approach we employed during development, that ranks entities by *senseProbability* and omits the supervised disambiguation step. Here the average latency remained well under a second, at the cost of accuracy (with a highest F1 score of 0.5227).

5. ANALYSIS

Whilst the lack of more fine-grained results (other than the F1 score over the aggregate of results) prevents us from performing an in-depth analysis on the results and reflect on precision and recall, we manually go through the logs of our final system and analyze its output to better understand the strengths and weaknesses of our system.

5.1 Weaknesses

In the following section, we illustrate some of the common causes for (in our opinion) wrongfully linked entities (false positives) and missed entities (false negatives).

5.1.1 False positives

The first issue we found were predictions of entities for labels that do not refer to non-named entities. Recurring examples include the mention `jobs` being linked to entity *Jobs_(film)*, and label `hair` to entity *Hair_(musical)*. The former example usually occurred in queries with location entities (i.e., cities), and the latter with celebrity names. At the same time, these false positives (musicals and films) show the preference of the trained classifier for named entities. Our system, however, did not output these entities in each query that contained the mentions. Whether or not this happened depends largely on the remaining text in the queries. With a larger amount of text surrounding the labels, the erroneous entities are more likely to rank comparatively low, resulting in them being pruned from the final output. For example, in the query `[best to buy a camera]`, the label `camera` is wrongfully linked to entity *Committee_for_Accuracy_in_Middle_East_Reporting_in_America*. However, in query `[rits camera orlando florida]`, which yields a higher number of entity candidates, the same label (`camera`) is not linked, and the incorrect entity is omitted in favor of entities *Florida* and *Orlando,_Florida*.

More strict document re-ranking, or a stronger inclusion of global features (that model coherency or relatedness between entities) could be explored to address these issues, even though the limited context will remain a challenge in short search engine queries.

5.1.2 False negatives

Missed entities, or recall errors, seemed mostly related to two underlying issues; (i) too strict initial filtering (at candidate generation), and (ii) shortcomings that relate to the lexical-matching-based candidate-generation approach.

The former subcategory exposes the challenge of linking "long-tail entities:" entities that are by definition unlikely to be referred to and hence receive low prior probabilities. Examples include ambiguous person names that generate a high number of entity candidates. For example, in case of query `[gary collins dies]` there is a large number of Gary Collins in Wikipedia, including a Canadian politician, racing driver, ice hockey player, American football player, politician from Idaho, and actor. Due to the large number of candidates, by definition some will receive a low prior probability and will not be considered for subsequent steps.

The second subcategory of false positives is where shortcomings of lexical matching cause our candidate-generation phase to miss entity candidates. One such shortcoming is dealing with omitted words, e.g., in query `[university tennessee football schedule]` the label `university tennessee` will never be linked (in full) to the right candidate (*University_of_Tennessee*), since the surface form "university tennessee" does not occur in our KB. However, the label `tennessee` is linked to the University candidate, albeit with a very low prior probability, turning the problem into the former of initial filtering. A second area where the correct entity candidates were not retrieved due to lexical-matching shortcomings was in queries where shorthands were used. In query `[b of america online banking]` the `b` shorthand (for *bank*) prevents the retrieval of the correct entity candidate (*Bank_of_America*). To make matters worse, our system outputs the incorrect entity *America_Online*, which would have been pruned out if *Bank_of*

America was properly linked (due to preferring the longest matching, non-overlapping mention). A third class of lexical matching seemed to have difficulties is a well-understood challenge in user-generated and web content: misspellings. However, misspellings were found to be intercepted in part by our system, mostly thanks to the surface forms that are extracted from Wikipedia's redirect pages. In Wikipedia editors create redirects for common misspellings to the intended pages (e.g., "snoop dog" redirects to entity *Snoop Dogg*). However, as expected coverage is not complete. e.g., `chesapeke bay` in the query `[size of chesapeke bay]` did not yield the entity *Chesapeake_Bay*.

These lexical matching shortcomings could be addressed in part by e.g., applying fuzzy matching strategies, or by including some form of "stopword normalization." However, with these strategies, both efficiency and the usefulness of the probability signals we employ (which rely on exact counts of links between surface forms and entities) will suffer. Expanding the number of known surface forms for entities is another solution. This can be achieved by e.g. considering links outside of the KB [11].

5.2 Strengths

In the majority of the queries we came across in our system's output, however, as reflected by our final system's performance, the Semanticizer has no problem successfully linking entities. Unambiguous, "popular," and otherwise common named entities (those with a high prior probability), including locations (`lake tahoe`, `rochester ny`), companies (`swarovski`, `costco`), bands (`rolling stones`, `maroon 5`), cars (`hyundai porter`, `bmw m3`), and person names (`melissa rauch`, `yoko ono`) seem to be no problem for the Semanticizer. Queries that highlighted the coverage of the lexical-matching method include (correctly) linking query `jayski silly site` to entity *Jayski's Silly_Season_Site*. And finally, we were positively surprised by our system's handling of longer queries. Longer queries can be considered more challenging, as more text typically yields larger numbers of entity candidates, but our system did not seem easy to fool into linking entities. For a query like `[if a group of chemical elements has gram atomic weights]` our system did not return any entities.

6. CONCLUSION

In this paper we describe how we adapt the Semanticizer, an open-source entity linking framework, to the task of linking named entities in search engine queries. To this end, we adapt an existing dataset to steer the Semanticizer's linking towards named entities. In addition, we introduce new contextual features to the supervised entity disambiguation, that leverage the little context that search engine queries provide.

Finally, in an error analysis we identify both the challenges that come with the domain of search engine queries, and shortcomings that are inherent to lexical-matching-based entity linking. These shortcomings and challenges include in particular dealing with omitted words, misspellings, and use of shorthands. To address these challenges, exploring methods to generate richer sets of entity mentions, or applying more lenient lexical-matching approaches, seem favourable, while at the same time retaining the advantages that are inherent to lexical matching: being efficient and (in principle) language and domain independent.

Acknowledgments. This research was supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreements nrs 288024 and 312827, the Netherlands Organisation for Scientific Research (NWO) under project nrs 727.-011.005, 612.001.116, HOR-11-10, 640.006.013, the Center for Creation, Content and Technology (CCCT), the QuaMerdes project funded by the CLARIN-nl program, the TROVe project funded by the CLARIAH program, the Dutch national program COMMIT, the ESF Research Network Program ELIAS, the Elite Network Shifts project funded by the Royal Dutch Academy of Sciences (KNAW), the Netherlands eScience Center under project nr 027.012.105, the Yahoo! Faculty Research and Engagement Program, the Microsoft Research PhD program, and the HPC Fund.

REFERENCES

[1] N. Balasubramanian and S. Cucerzan. Topic pages: An alternative to the ten blue links. In *ICSC '10*, 2010.

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

[3] E. Gabrilovich and S. Markovitch. Wikipedia-based semantic interpretation for natural language processing. *J. Artif. Intell. Res. (JAIR)*, 34:443–498, 2009.

[4] D. Graus, T. Kenter, M. Bron, E. Meij, and M. de Rijke. Context-based entity linking - University of Amsterdam at TAC 2012. In *TAC '12*. NIST, 2013.

[5] D. Graus, M. Tsagkias, L. Buitinck, and M. de Rijke. Generating pseudo-ground truth for predicting new concepts in social streams. In *ECIR '14*, volume 8416 of *LNCS*, pages 286–298. Springer, 2014.

[6] J. He, M. de Rijke, M. Sevenster, R. van Ommering, and Y. Qian. Generating links to background knowledge: a case study using narrative radiology reports. In *CIKM '11*, 2011.

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

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

[9] L. Ratinov, D. Roth, D. Downey, and M. Anderson. Local and global algorithms for disambiguation to wikipedia. In *HLT '11*, pages 1375–1384. Association for Computational Linguistics, 2011.

[10] Z. Ren, S. Liang, E. Meij, and M. de Rijke. Personalized time-aware tweets summarization. In *SIGIR '13*, pages 513–522. ACM, 2013.

[11] V. I. Spitkovsky and A. X. Chang. A cross-lingual dictionary for english wikipedia concepts. In *LREC'12*, Istanbul, Turkey, may 2012. ELRA.

[12] N. Voskarides, D. Odijk, M. Tsagkias, W. Weerkamp, and M. de Rijke. Query-dependent contextualization of streaming data. In *ECIR '14*, 04/2014 2014.

[13] J. Wang and J. Han. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Trans. Knowledge and Data Engineering*, June 12, 2014.