Tulip: Lightweight Entity Recognition and Disambiguation Using Wikipedia-Based Topic Centroids

Marek Lipczak
Faculty of Computer Science
Dalhousie University
Halifax, Canada
lipczak@cs.dal.ca

Arash Koushkestani
Faculty of Computer Science
Dalhousie University
Halifax, Canada
arash.koushkestani@dal.ca

Evangelos Milios
Faculty of Computer Science
Dalhousie University
Halifax, Canada
eem@cs.dal.ca

ABSTRACT

This article presents Tulip, an ERD system submitted to the ERD 2014: Entity Recognition and Disambiguation Challenge. The objective of the proposed system is to spot mentions of entities in a document and link the mentions to corresponding Freebase articles. To achieve it, Tulip prunes the set of entity candidates focusing on a core subset of related entities capturing the context of the document. The relationship strength is measured as a similarity to a topic centroid generated from entity features. Each entity is represented by an accurate and compact feature vector extracted from a category graph built based on information from 120 language versions of Wikipedia. Given the core set of accepted entities Tulip uses the Wikipedia-based feature vectors to extract more related entities from the document text. Tulip received the first prize in the long document track with F1 score of 0.74, which confirms the effectiveness of our system. At the same, the system was faster than all other submissions with latency under 0.29 seconds.

Categories and Subject Descriptors

I.2.7 [**Natural Language Processing**]: Text analysis

General Terms

Algorithms, Experimentation

Keywords

Entity Recognition and Disambiguation, ERD 2014 – Long Track, Term Centroids, Wikification, Text Annotation, Linked Open Data

1. INTRODUCTION

Text documents in the form of news, web pages, blogs, and technical documents contain mentions of named entities such as people, companies or locations. The goal of Entity Recognition and Disambiguation (ERD) is to identify mentions of entities and link them to a relevant entry in an external knowledge base. This task is also known under the names of Entity Linking [19], Wikification [12] or more generally text annotation. Text annotation and interlinking documents with external knowledge bases is an interesting problem with many practical applications such as semantic search [1], faceted browsing [7], recommender systems [13], and text categorization [5]. The motivation of the ERD 2014: Entity Recognition and Disambiguation Challenge [3] was to advance the state of the art in the field for both short documents (e.g., search queries) and long documents (e.g., web pages). This article presents Tulip, an ERD system which was a submission to the challenge and the recipient of the first prize in the long documents track.

The ERD process is usually divided into two steps: *spotting* and *disambiguation*. In the first step, the system spots potential *mentions* of entities in text and links them to a list of *senses* which are the *candidate entities* that can be referred to by the given mention. Each entity stored in the system's data repository is represented by a list of *surface forms*. For example, Halifax is a surface form for a city in UK and Halifax Regional Municipality in Canada among others. In the second step, the system disambiguates the candidate entities selecting the most probable entity for each mention. Our system differs slightly from this scheme. It greatly relies on the *default sense* of the entity (i.e., the entity that is most frequently observed for the given mention), which simplifies the disambiguation process. At the same time, the system focuses on the *recognition task* – the ability to select valid mentions, while pruning mentions that are in fact common phrases or mentions of entities that are not present in the knowledge base.

The key factor that increases the difficulty of Entity Recognition and Disambiguation problem is language ambiguity. In any given language, most of the words and phrases, including names of entities, carry more than one meaning. At the same time, a single entity can be referred to by more than one name. Despite all the challenges that language ambiguity creates for computer systems, the problem of entity recognition and disambiguation in long documents is almost effortlessly solved by humans. In our opinion there are three key factors that contribute to this fact. First, a coherent topic of most documents which gives contextual clues for readers on how to disambiguate entities. Second, explicit clues left by the document author to ease the problem. Third, a well defined default sense for most phrases that can potentially be references to named entities. While designing Tulip, our objective was to leverage these facts. The main contributions of our work are:

ERD'14, July 06 - 11 2014, Gold Coast , QLD, Australia
ACM 978-1-4503-3023-7/14/07 ...$15.00.
http://dx.doi.org/10.1145/2633211.2634351

- The use of external data source – Wiktionary for selecting suspicious entity mentions that are in fact likely to be common words or phrases.
- A dedicated approach for handling personal names as a frequent source of ambiguity problems.
- A centroid-based representation of document topics used for pruning of entities that are not consistent with the document.

Entity Recognition and Disambiguation is an important problem, but typically it is just a single step in a more complex document processing system. For this reason, one of the objectives of our work was to design a lightweight ERD system – fast, adaptable, using no complex data repositories that have to be stored externally, but at the same time having low memory footprint. Tulip matches all of these criteria, since together with all external services it requires less than 4 GB of operational memory. Its latency during the evaluation was under 0.29 seconds for documents with the average length of 528 words.

2. RELATED WORK

Most of the ERD systems presented in the literature focus on the disambiguation problem. The solutions typically use a combination of two factors. Following the work by Milne and Witten [12] we refer to them as *commonness* and *relatedness*. Commonness represents the notion of how likely it is that a given surface form represents a specific concept or entity, without any contextual knowledge. Typically commonness scores are mined from a large Wikipedia corpus by extracting the frequency distribution of target articles for wiki-links with a given surface form as the anchor text [8, 12]. Another possibility is to represent commonness as the prior class probability when disambiguation is solved as a classification problem [9]. The relatedness score represents how well the given entity fits the context of the document. This problem has a much broader range of solutions. Text-based methods represent each entity as a term vector. The terms are typically mined from Wikipedia paragraphs in which the given entity was referred to with a wiki-link [8] or entity's Wikipedia articles [10]. The term vector is later compared with the full document text to obtain the relatedness score. Link-based methods focus on the pair-wise relatedness links for candidate entities extracted from the document. The link strengths are calculated independently of document text using an external repository, typically Wikipedia. Milne and Witten [12] calculate the pair-wise similarity between all entities represented by Wikipedia articles using the overlap of the article's outgoing wiki-links. This way they are able to select a sense that is most related to other entities found in the text. A similar approach is used in the TAGME system [4].

Both text-based and link-based approaches require large and complex data structures: a full-text index for text-based systems or a large and dense similarity graph for link-based systems. It creates serious storage and efficiency challenges. In addition, both approaches rely on the information gathered in the external source (Wikipedia). If the representation of an entity is insufficient (e.g., short Wikipedia article) the ERD system may not be able to find the evidence that the entity is related to the article. Because of these two reasons we decided to use the Wikipedia category graph as the base for the relatedness score. Smaller size and higher sparsity allows us to store the graph more efficiently. At the same time, even short Wikipedia articles (stubs) tend to be well connected to the category graph.

3. DATASETS AND SERVICES

3.1 Datasets

The effectiveness and efficiency of Tulip would not be possible without a number of openly available datasets. In this section we provide an overview of five datasets used in the project focusing on their applicability to the Entity Recognition and Disambiguation problem.

Wikipedia is being widely used in text annotation since it is densely structured by hundreds of millions of links among millions of articles [12]. Internal links in Wikipedia (*wiki-links*) and the anchor text associated with each one create a rich dataset with valuable statistical information. Given the number of links Wikipedia has a very high coverage of entity surface forms that can be extracted from links' anchor text. The frequency of reference to an article by a given anchor text is often used in disambiguating mentions which refer to more than one article. We refer to it as the *commonness score*.

Google's Wikilinks corpus [16] can be considered as an extension of Wikipedia's wiki-links data to the Web. It contains 40 million mentions of over three million entities. These mentions are gathered based on finding hyperlinks to Wikipedia from a web crawl of over 10 million pages.

Freebase [2] is a collaboratively created knowledge base. It contains data harvested from Wikipedia and other data sources. Each entity described in Freebase is manually assigned to one or more types. The type assignment was used to select the subset entities of entities for the ERD'14 Challenge [3]. In the challenge we worked on a subset of over two million entities extracted from Freebase. We refer to them as *Freebase sample*. All entities in the Freebase sample contained a link to corresponding Wikipedia article. The Freebase types can be also used to select entities for type specific approaches. For example, Tulip uses a special preprocessing technique for all entities of type person.

DBpedia [6] is a knowledge base of relations automatically and manually extracted from Wikipedia. Just like Freebase it contains its own type hierarchy. One of the distinctive features of DBpedia is its effort in unifying information from various language versions of Wikipedia. All relations extracted from Wikipedia articles in languages other than English are mapped to the English counterpart using Wikipedia's language links. We use this feature while computing term vectors representing entities.

Wiktionary is a free, collaboratively created dictionary. It can be considered as a data source complementary to Wikipedia [18], with a stronger emphasis on commonly used words and phrases. Wiktionary has a rich representation of common nouns and other parts of speech which we use as an evidence that a spotted entity mention can be in fact a commonly used phrase.

3.2 Services

Tulip is built with a modular structure in mind. Most of its components communicate through API calls. The complete structure of the system is described in Section 4. Here we introduce two web services (Solr Text Tagger and Sun-

flower) that are used by Tulip but were developed independently.

The main function of **Solr Text Tagger** is to spot entity mentions given a predefined lexicon of entity surface forms. Solr Text Tagger is based on the Finite State Transducers (FST) implementation available in Solr. The system was written by David Smiley and Rupert Westenthaler as a part of the OpenSextant project[1]. Its open-source version is publicly available online[2]. The key factor that makes Solr Text Tagger an efficient entity spotting approach with low memory footprint is the use of FST representation based on the algorithm proposed by Mihov and Maurel [11]. For our specific application we can perceive FST as a sorted map in which the key is the surface form and the value is the list of entities linked to it. In this setting, each surface form can be linked to multiple entities and a single entity can be represented by more than one surface form.

Solr Text Tagger spots entities in a single pass. Given the input string tokenized into words, the system iterates over the tokens. Each new token creates a cursor that is used to traverse the FST. The surface forms can overlap (e.g., "New York City" and "York"), therefore it is possible that there are more than one active cursor at a time. This situation is rare and has no impact on efficiency. At the same time, it eliminates the need for additional data structures as in case of Aho-Corasick algorithm, which could be used for the same task [8]. When the cursor successfully reaches the final state (i.e., the full surface form is matched) it outputs the list of entities together with the offsets of the spotted surface form and becomes inactive. Whenever there are no active cursors the system optionally runs a pruning algorithm to remove overlapping surface forms from the list of the most recent outputs. Tulip uses "longest dominant right" pruning option, which means that in case of an overlap the longest surface form is selected (or the rightmost in case of a tie).

Sunflower is a Wikipedia-based semantic network developed at Dalhousie University in parallel with Tulip. What makes Sunflower different from a large number of knowledge bases and semantic networks extracted from Wikipedia is the focus on the notion of importance of the represented relation. For example, the fact that Barack Obama is the President of the United States has higher importance score than Barack Obama being Grammy Award-winning artist, although both of these facts are correct.

We use Sunflower to generate a compact category profile for a given entity. The profile is generated using the *Wikipedia Category Graph*. In Wikipedia each article that represents an entity is assigned to a set of specific categories. These categories can be further assigned to a set of more general categories making a pseudo-taxonomy graph. The structure of the Wikipedia Category Graph resembles the structure or expert-made semantic networks (e.g., WordNet) [17]. Our goal is to use this structure to infer the types of an entity at different levels of generality. Unfortunately, unlike expert-made networks, inference on the Wikipedia Category Graph typically brings unsatisfactory results [14]. The reason is a large number of relations that is obscure or incorrect. It is important to mention that Wikipedia articles are typically assigned to a large number of categories (e.g., 48 for the article about Barack Obama), therefore even considering the second order neighbours in the category graph will result in a large and noisy set. To deal with this problem we decided to assign an importance score to each relation in the Wikipedia Category Graph. The importance score is calculated by unifying information from 120 language versions of Wikipedia. Each language version acts like an independent witness of relations that it contains. The witness count is later divided by the total number of language versions in which the article is present. For example, considering Barack Obama, the score for Presidents of the United States category is 0.83 and the score for Grammy Award-winning artists is 0.18. The information about the relations is extracted from DBpedia. The importance score allows us to rank categories and select the k most important outgoing links for each node, including category nodes. As a result we have a much sparser weighted graph which better represents the relations between entities and their categories. Given a maximum length d and a seed entity, we can follow all the paths in the graph accessing the categories that characterize the entity. The path connecting the entity and the category has a weight which is the product of the importance scores on all its edges. If the category can be accessed by more than one path, the path weights are added. Finally, we obtain a vector of category - weight pairs that characterize an entity. Table 1 presents the top five terms from the category vector extracted for Barack Obama for $k = 3$ and $d = 4$.

Table 1: Top terms for entity Barack Obama extracted from Sunflower's category graph.

rank	category name	score
1	Presidents of the United States	1.00
2	Politicians	0.85
3	American politicians	0.76
4	United States	0.67
5	Presidents	0.67

4. SYSTEM DESCRIPTION

Tulip is built of two independent components. We refer to them as Spotter and Recognizer. The Spotter operates on the document text. Its objective is to extract all entity mentions and assign all potential entity candidates for each mention. The Recognizer operates on the list of mentions selected by the Spotter. Its objective is to select a core set of entities that are most likely to be correctly spotted and extend this set by related entities.

4.1 Spotter

The Spotter uses a lexicon of surface forms that are associated with one or many tracked entities. Tulip creates its lexicon based on entity names extracted from the Freebase sample as well as anchor text of links found in Wikipedia or sites external to Wikipedia. The detailed information about the construction of the lexicon is presented in Section 4.1.1. In Section 4.1.2 we describe a special approach for entities of type person. For the challenge dataset, the lexicon consists of over four million surface form - entity pairs. The extraction of potential entity mentions is done by Solr Text Tagger. Given a body of text Solr Text Tagger marks the offsets (locations in the text) of all indexed surface forms

[1]http://www.opensextant.org/

[2]https://github.com/OpenSextant/SolrTextTagger

that can be found in the text together with the corresponding entities (senses). The system also retrieves additional information that was stored together with the surface - entity pair. It includes the number of occurrences of the pair in the dataset (commonness) and a flag that suggests that the given mention can be in fact a common phrase.

4.1.1 Building the Lexicon

The lexicon of surface form - entity pairs contains the vocabulary of words and phrases that can potentially indicate the mention of the entity in the text. To build a comprehensive lexicon we used information from three data sources: (1) Freebase sample extracted by the challenge organizers (2) Wikipedia, (3) Google's Wikilinks corpus (see Section 3.1 for details). First, we extracted all the entity names from the provided Freebase sample. Next we processed all internal Wikipedia links from a Wikipedia dump retrieved in February 2014 storing the anchor text and the link target. We combined this dataset with Google's Wikilinks corpus creating a repository of anchor text - entity pairs together with the commonness score. The dataset was later merged with entity names, treating all names as a single link to their entity. The objective of the challenge was to recognize only the entities provided in the sample. However, some of these entities share the name or the anchor text with other entities in Wikipedia. Therefore we filtered the repository keeping only the entities from the Freebase sample or Wikipedia articles that share at least one anchor text with any of the Freebase entities.

Tulip's lexicon contains over 4 million surface forms linked to over 2 million entities. High coverage of entities comes with the cost of a large number of common phrases that are mistakenly taken as surface forms. For example, a word *details* is used as a surface form for over 5000 entities in our dataset. We prune surface forms that are likely to be incorrect, which are words with a high number of linked entities or strings with a majority of non-letter characters. Another problem are ambiguous entity names which are also used as common words (e.g., "It" – Stephen King's novel). As these surface forms can be a cause of a large number of falsely recognized entities, we decided to mark them as potentially suspicious. If these surface forms are spotted in the text as potential entity mentions, the Spotter will assign a *suspicious* flag to the mention. The current version of the system uses a composition of three soft filters: (1) **stop-word filter** marks all stop-words or phrases composed of stop-words (e.g., *This is*); (2) **Wiktionary filter** marks all common nouns, verbs, adjectives, etc. found in Wiktionary; (3) **lower-case filter** marks all lower-case words or phrases. It is important to notice that the first two filters are case insensitive. For all filters, mentions with strong evidence for the connection with the default sense are not filtered. By strong evidence we consider a large number of links with a given mention as an anchor text leading to the same entity. For example, the word "Apple" very often leads to Apple Inc. company.

4.1.2 Handling Personal Names

Almost 50% of entities in the Freebase sample belong to the type person. Often their surface forms are either first or family name, both of which are highly ambiguous. At the same time, personal names or their parts can be frequently found in document text. Given the sample of 50 training documents released by the challenge organizers we estimated that 18% of the entities found in text is of type person. Nearly 10% of all entity mentions in the training data was recognized by us as either first or family name of a person. All these cases are very hard to process without a special treatment. To deal with this problem we created a special class of surface forms that are single words extracted from personal names. They are removed from the lexicon for all entities of type person and re-added as a generic name entry. During spotting, for any generic name mention, the system tries to find a person spotted in the text with their full name. Only when the full name contains the generic name the latter is considered as a valid mention of the entity. For example, London is only considered as a potential mention of Jack London if his full name also appears in the text.

4.2 Recognizer

The result of the Spotter is a list of entity mentions found in the document text. Each mention has a set of possible senses assigned to it. Some of the mentions are flagged as suspicious if there is a possibility that they are common words mistakenly spotted as entities. The most common sense for each mention is flagged with a *default sense* flag. The objective of the Recognizer is to select which mentions are truly referring to entities (recognition) and for them to select the proper entity from the set of senses (disambiguation). Tulip uses a "global" approach for entity recognition [15]. The possible senses from all mentions are merged together into a single entity set E, without consideration of their local context. If the entity was flagged as default sense in any of the input lists it keeps its flag. The suspicious flag is kept only if all mentions for the given entity were marked as suspicious. Given the entities, the Recognizer builds a topic centroid that is used to score the entities based on their relatedness to the document topics.

4.2.1 Topic Centroid for Entities

Using a three step procedure the Recognizer assigns a score to entities from the set E. The score represents the likelihood of the entity being related to the document. It is calculated based on the similarity of each entity to the topic centroid obtained from the term vectors of selected entities. The score is later used to select the output entities.

Step 1 – Topic Centroid Generation. In the first step, the system selects all entities that are flagged as default sense and not flagged as suspicious to build a topic centroid for the document. We refer to this set as the *entity core* – these are the entities that are most likely to be correct among all entities in set E. For each selected entity the system retrieves its category-based term vector from Sunflower. We experimented with various settings of the k and d parameters finding the best accuracy for $k = 3$ and $d = 4$. The topic centroid is a linear combination of normalized term vectors.

Step 2 – Topic Centroid Refinement. Even though the entity core contains entities that are most likely to be correct, the set can still contain incorrect entities. In the second step, the system calculates the cosine similarity between all entities in the core and the centroid. All entities, for which the similarity is lower than a predefined threshold t_{coh}, are removed from the core. The objective is to further refine the core removing the outliers that does not match the general topics of the document. The topic centroid is recalculated using the refined core.

Step 3 – Entity Scoring. Given the refined topic centroid, the system calculates the similarity between the centroid and all entities in set E marked with the default sense flag. The similarity becomes the score assigned to each entity. We have considered assigning the similarity score to all entities in set E. However there is only a small fraction of mentions that could potentially benefit from this approach. Based on the 50 training set documents we estimated that among all mentions with at least two senses the default sense is correct for 85% of mentions. For another 5% of mentions the correct entity was the default sense for another less ambiguous mention (e.g., *E72* and *Nokia E72*). Although proper disambiguation can be useful for the remaining 10% of cases, it is more likely that by considering other senses the system would incorrectly discard the default sense. We confirmed this hypothesis in preliminary experiments in which all entities were scored in this step.

4.2.2 System Output

In the last stage of the process the scores are used to rank the list of entity candidates for each mention. The system returns only entities for which the score is not lower than t_{coh}. The value of t_{coh} parameter can be used to tune the precision/recall trade-off. Higher values of the parameter result in higher precision as a lower number of mentions for which at least one entity is close enough to the topic centroid to be accepted as a valid result. Lower values result in higher recall. Although the output of the system is a ranked list of entities for each mention, typically only the first entity from the list is used. That was the case in the challenge evaluation approach.

4.2.3 System Modifications

While developing the system we tested a number of modifications. Some of them are discussed in Section 6 as future work plans. The only modification, that was used in the final submission, was using outgoing Wikipedia links to further expand the set of accepted entities. For all the entities from the refined core we selected the 10 most important outgoing links and added them to a set of linked entities L. The importance score was calculated in Sunflower using a language version criterion similar to the one used in the category graph. If an entity from set E was found in set L it was artificially given score t_{coh}. The training phase suggested that this modification gives a slight recall improvement; however, we are not confident about the effect of the modification on the final evaluation score.

5. SYSTEM EVALUATION

5.1 Evaluation Setup

All participating systems were implemented as publicly accessible web services. Given the request containing document text the services were supposed to respond with the list of mentions, their positions in the text and disambiguated senses. During the test period the organizers ran a batch test containing 100 documents for each participating system. The responses were used as a seed pool of entities that was later cleaned and refined by human annotators. The gold standard set produced this way was used to estimate the effectiveness of submitted systems. The systems were ranked based on micro-averaged F1-score; however, average precision and recall were also reported. In addition, the organizers reported the latency of all systems [3].

Table 2: Challenge results for the first ten systems in long document track.

rank	team name	F1	prec./recall	latency
1	MS_MLI	**0.76**	0.83/0.70	1.49
2	*MLNS (Tulip)*	0.74	0.76/**0.71**	**0.29**
3	Seznam Research	0.72	0.79/0.66	2.33
4	NTUNLP	0.71	0.76/0.67	7.66
5	HITS	0.70	0.77/0.65	4.97
6	Neofonie	0.70	0.76/0.65	0.53
7	WebSAIL	0.69	0.72/0.65	0.70
8	Acube Lab	0.67	**0.87**/0.54	0.86
9	ExPoSe	0.63	0.74/0.55	0.71
10	UBC	0.63	0.74/0.55	37.29

5.2 Challenge Results

The challenge results for the first ten systems are presented in Table 2. Tulip (submitted under the team name MLNS) achieved 0.735 F1 score, which was the second result among all submitted systems. Nevertheless, Tulip was granted the first prize in the competition. The top score (F1=0.759) was achieved by MS_MLI system. The system author was affiliated with one of the challenge organizers, therefore it was not taking part in the competition.

The comparison of the results shows small differences between the top participating systems. A detailed study is needed to fully understand why Tulip managed to outperform most of the competitors. Therefore, at this stage we would like to focus on other important aspects of the ERD problem efficiency and easiness of use.

5.3 Lightweight Entity Recognition and Disambiguation

In terms of system efficiency Tulip outperformed all competitors by a fair margin. Its average latency was 0.29 seconds per document, with documents of average size 528 words. The processing speed allows the system to process large-scale repositories or be used in scenarios that require real-time response (e.g., semi-supervised ERD).

The system uses no data repositories that would require hard-drive storage (e.g., full-text index) or have large memory footprint (e.g., full Wikipedia link graph). Tulip itself operates on very sparse term vectors (typically less than 100 elements), therefore its memory footprint is negligible. The two external services are very well optimized for memory use. We did not run extensive studies on the memory use of Solr Text Tagger, but we follow the author's estimates of 200 MB for a dictionary of 10 million surface forms. The Sunflower database crafted for the Freebase sample used in the challenge contains a dictionary of less than three million Wikipedia concepts and categories. Each dictionary element is mapped to a list of five categories that are used to build the entity term vector. All together we were able to run Tulip in a virtual machine with 4 GB of operational memory, but if needed it can be further optimized. Given these characteristics Tulip can be easily deployed and scaled in cloud services.

6. CONCLUSIONS AND FUTURE WORK

The presented system, Tulip represents the processed document and the spotted entity candidates in the vector space of Wikipedia categories. Accurate and compact term vector representation is possible thanks to the pre-processed Wikipedia category graph which unifies information from 120 language versions of Wikipedia. Instead of calculating pair-wise relatedness scores for entities, Tulip generates a single centroid that allows the system to quickly estimate the coherence of an entity with other candidates. The results of the challenge confirmed that category-based topic centroids can be an efficient and effective way of solving the entity recognition and disambiguation problem.

The experiments with the system demonstrated that the accuracy of extracted entities relies more on the successful recognition of correct entity mentions rather than their disambiguation. Tulip addresses the recognition process in two stages. First, it selects the mentions that in fact are likely to be common phrases mistakenly selected as entities. Second, it accepts only default senses of entities that are coherent with the document topics. As we demonstrated, about 85% of the mentions can be disambiguated with their default sense. If not, there is a large chance that the entity is disambiguated by the document author, who used a more explicit mention for the entity.

Finally, an important part of the system was special treatment of entities of type person. Personal names can be very ambiguous, therefore they should be accepted as a reference to a person only if an explicit mention is also present in the document.

In the future work on the system we plan to experiment with a more complex centroid structure that accounts for multiple independent topics that can be found in text (i.e., topic clusters). In addition, we want to experiment with other Wikipedia-based relations that can be used to represent entities. In the current version, we have used the wiki-link graph along side the category graph, but we are not yet confident that the result improvements are worth the increased complexity of the system.

Another direction for future work is moving towards local entity disambiguation in which each mention is considered individually in its closest context. We plan to use a word n-gram model to narrow down the search space for the disambiguation phase. Although this model was implemented, it was not used in the final system. This module takes previous words of a spotted mention and decides on the type of target entity.

Additional project resources can be found at:
`https://web.cs.dal.ca/~lipczak/erd/`

7. ACKNOWLEDGMENTS

We would like to thank Axel Soto, Armin Sajadi, Seyednaser Nourashrafeddin and Krzysztof Lipczak for useful discussions and help with the evaluation of the system.

8. REFERENCES

[1] K. Balog, D. Carmel, A. P. de Vries, D. M. Herzig, P. Mika, H. Roitman, R. Schenkel, P. Serdyukov, and T. T. Duc. The first joint international workshop on entity-oriented and semantic search (jiwes). *SIGIR Forum*, 46(2):87–94, Dec. 2012.

[2] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA, 2008. ACM.

[3] 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).

[4] P. Ferragina and U. Scaiella. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1625–1628, New York, NY, USA, 2010. ACM.

[5] E. Gabrilovich and S. Markovitch. Overcoming the brittleness bottleneck using wikipedia: Enhancing text categorization with encyclopedic knowledge. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, AAAI'06, pages 1301–1306. AAAI Press, 2006.

[6] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 2014.

[7] G. Marchionini. Exploratory search: From finding to understanding. *Commun. ACM*, 49(4):41–46, Apr. 2006.

[8] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. Dbpedia spotlight: Shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems*, I-Semantics '11, pages 1–8, New York, NY, USA, 2011. ACM.

[9] R. Mihalcea. Using wikipedia for automatic word sense disambiguation. In C. L. Sidner, T. Schultz, M. Stone, and C. Zhai, editors, *HLT-NAACL*, pages 196–203. The Association for Computational Linguistics, 2007.

[10] R. Mihalcea and A. Csomai. Wikify!: Linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 233–242, New York, NY, USA, 2007. ACM.

[11] S. Mihov and D. Maurel. Direct construction of minimal acyclic subsequential transducers. In *Revised Papers from the 5th International Conference on Implementation and Application of Automata*, CIAA '00, pages 217–229, London, UK, UK, 2001. Springer-Verlag.

[12] D. Milne and I. H. Witten. Learning to link with wikipedia. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, pages 509–518, New York, NY, USA, 2008. ACM.

[13] C. Musto, G. Semeraro, P. Lops, and M. de Gemmis. Combining distributional semantics and entity linking for context-aware content-based recommendation. In V. Dimitrova, T. Kuflik, D. Chin, F. Ricci, P. Dolog, and G.-J. Houben, editors, *UMAP*, volume 8538 of *Lecture Notes in Computer Science*, pages 381–392. Springer, 2014.

[14] H. Paulheim and C. Bizer. Type inference on noisy rdf data. In *ISWC*, pages 510–525. Springer, 2013.

[15] L. Ratinov, D. Roth, D. Downey, and M. Anderson. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 1375–1384, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[16] S. Singh, A. Subramanya, F. Pereira, and A. McCallum. Wikilinks: A large-scale cross-document coreference corpus labeled via links to Wikipedia. Technical Report UM-CS-2012-015, 2012.

[17] T. Zesch and I. Gurevych. Analysis of the wikipedia category graph for nlp applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT)*, pages 1–8, Rochester, Apr. 2007. Association for Computational Linguistics.

[18] T. Zesch, C. MÃijller, and I. Gurevych. Extracting lexical semantic knowledge from wikipedia and wiktionary. In *Proceedings of the Conference on Language Resources and Evaluation (LREC), electronic proceedings*. Ubiquitious Knowledge Processing, UniversitÃd't Darmstadt, Mai 2008.

[19] W. Zhang, J. Su, C. L. Tan, and W. T. Wang. Entity linking leveraging: Automatically generated annotation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 1290–1298, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.