Revealing Entities from Textual Documents Using a Hybrid Approach

Julien Plu, Giuseppe Rizzo, Raphaël Troncy

EURECOM, Sophia Antipolis, France
{julien.plu,giuseppe.rizzo,raphael.troncy}@eurecom.fr

Abstract. The tasks of entity extraction, recognition, and linking are largely affected by the nature of the textual documents being analyzed. In fact, a lot of research efforts have focused on improving each task for both formal text (such as newswire documents) and for informal text (such as tweets). In this work, we propose a so-called hybrid approach that aims to be agnostic of the document type. Two datasets, namely the #Micropost2014 NEEL corpus and the OKE2015 test dataset, are used to benchmark the performance of our approach. The experimental results show that the approach presented in this paper outperforms the state-of-the-art systems on OKE2015 dataset and provides good results for the #Micropost2014 dataset.

Keywords: Entity Recognition, Entity Linking, Entity Filtering, Entity Resolution, Micropost, Formal Text

1 Introduction

The Web of documents has significantly grown in the last decades, resulting in an ever increasing amount of unstructured data being published, such as newswire documents, encyclopedic content or scientific papers. On the other hand, with the advent of social media services, such as Twitter and Facebook, new unstructured data has contributed to make the Web larger and wider, taping a larger human audience and embracing a different style of writing. The content being shared on social media is generally short in length, dynamic in terms of topics and events being covered, and informal in the writing style (less curated syntax and grammar). In this paper, we refer to respectively formal and informal text when talking about those two categories of textual documents.

For years, the task of entity recognition, along with word sense disambiguation and entity linking, has been tested on well-formed text (e.g newswire). The advent of microposts has introduced a breakthrough: approaches being used in the past did not fit this new type of textual data, and tailored approaches have been proposed to cope with text characterized by: *i)* less than 140 characters (Twitter), *ii)* strong ambiguity, *iii)* spelling errors, *iv)* non standard and lexical items, *v)* non standard syntactic patterns.

The aim of this work is to propose a hybrid approach for performing the tasks of entity extraction, recognition and entity linking[1] that would perform fairly robustly on

[1] We consider an *entity*, a DBpedia resource that corresponds to a Wikipedia article. We consider a *mention*, one of the possible way to name an entity.

both formal and informal textual documents. The approach is hybrid since it makes use of both linguistic and semantic algorithms working together and enclosed in a pipeline of stages that can be turn on and off. In the experimental settings, we use DBpedia2014 as knowledge base to link the entities being extracted from text, and two standard benchmark datasets: the #Microposts2014 NEEL [1] corpus which is composed of tweets and the OKE2015[2] corpus which is composed of paragraphs taken from Wikipedia articles. The proposed approach is divided in three tasks: entity extraction, entity recognition, and entity linking. The entity extraction task refers to spotting mentions from text. The entity recognition task refers to the task of giving a type to the extracted mention. Entity linking refers to the task of disambiguating the mention in a targeted knowledge base, and it is often composed of two sub-tasks: generating candidates and ranking them according to scoring functions.

The paper is organized as follows: Section 2 reviews the strengths and weaknesses of state-of-the-art approaches in processing formal and informal text. Section 3 describes our approach and its technical components. Section 4 reports the results of our evaluation. Section 5 highlights some frequent errors while Section 6 proposes some future work.

2 State of the Art

Numerous approaches have been proposed to tackle the task of extracting, typing and linking entities in formal texts and microposts. We have divided the state-of-the art in two parts: approaches that deal with formal texts and approaches dealing with microposts, generally tweets.

2.1 Formal Text

Amongst the recent and best performing systems, WAT [7] builds on top of TagME algorithms and follows the four steps approach we are advocating: extraction, typing, linking and pruning. For the extraction, a dictionary that contains titles, surface forms and redirect pages with a list of all their possible links ranked according to a probability score is used. The extraction performance can also be tuned with an optional binary classifier (SVM with linear or RBF kernel) using statistical (features) for each entity referenced in the dictionary. For typing the entities, WAT relies on OpenNLP NER and supports the three types PERSON, LOCATION and ORGANIZATION. For linking entities, WAT uses two methods, namely voting-based and graph-based algorithms. The voting-based approach assigns one score to each entity. The entity having the highest score is then selected. The graph-based approach builds a graph where the nodes correspond to mentions or candidates (entities) and the edges correspond to either mention-entity or entity-entity relationships, each of these two kinds of edges being weighted with three possible scores: *i)* identity, *ii)* commonness that is the prior probability $Pr(e|m)$ and *iii)* context similarity that is the BM25 similarity score used by Lucene[3]. The goal is to find the subgraph that interlinks as many mentions as possible.

[2] `https://github.com/anuzzolese/oke-challenge`

[3] `https://lucene.apache.org/core/`

DBpedia Spotlight [4] uses a gazetteer containing a set of labels from the DBpedia lexicalization dataset for the extraction. More precisely, the LingPipe Exact Dictionary-Based Chunker with the Aho-Corasick string distance measure is being used. Extracted mentions that only contain verbs, adjectives, adverbs and prepositions can be detected using the LingPipe part-of-speech tagger (POS Tagger) and then discarded. For the typing step, DBpedia Spotlight re-uses the type of the link provided by DBpedia. For the linking, DBpedia Spotlight relies on the so-called TF*ICF (Term Frequency-Inverse Candidate Frequency) score computed for each entity. The goal of this score is to show that the discriminative strength of a mention is inversely proportional to the number of candidates it is associated with. This means that a mention that commonly co-occurs with many candidates is less discriminative.

AIDA [3] uses a different approach: instead of having one method for the extraction and one method for the typing, the system relies on Stanford NER to combine both tasks. For the linking, AIDA uses a similar approach than the graph-based method of WAT. The graph is built in the same way but only one score for each kind of edge (mention-entity or entity-entity) is proposed. The score used to weight the mention-entity edges is a combination of similarity measure and popularity while the score used to weight the entity-entity edges is based on a combination of Wikipedia-link overlap and type distance.

Babelfy [6] uses a part-of-speech tagger in order to identify the segments in the text which contain at least one noun and that are substring of the entities referenced in BabelNet (extraction step). For the typing step, the Babelnet categories are used. For the linking, the system uses another graph-based approach where two main algorithms have been developed: random walk and a heuristic for finding the subgraph that contains most of the relations between the recognized mentions and candidates. The nodes are pairs (mention, entity) and the edges correspond to existing relationships in BabelNet which are scored. The semantic graph is built using word sense disambiguation (WSD) that extracts lexicographic concepts and entity linking for matching strings with resources described in a knowledge base.

Those four methods (and particularly WAT and Babelfy) perform well on formal text, but rather poorly on informal text such as tweets. WAT is limited as they match any entry in the dictionary, including terms that are common words such as verbs or prepositions. For typing, WAT is limited to the kind of mentions that can be typed by OpenNLP NER. Similarly, AIDA depends on Stanford NER and the specific model used by the CRF algorithm. Those four systems are also limited to the fixed knowledge base being used, thus only extracting and linking entities that are referenced which prevents to recognize emerging entities.

2.2 Informal text

E2E [5] is an end-to-end entity linking system specialized for short and noisy text. The architecture of this tool is composed of four steps: i) text normalization where the retweet symbol, special characters (such as emojis) and additional white spaces are removed. For example, the tweet *RT: I like Paris :)* is transformed into *I like Paris*; ii) candidate generation where a dictionary based on Wikipedia and Freebase is used to generate the candidate surface forms; *iii)* joint recognition and linking where the entity recognition and linking are seen as a single task. The system uses a supervised learning method where for a given message and a candidate mention, all the possible entities are ranked and one is selected. The joint recognition and disambiguation task is crucial to properly link surface forms to their corresponding entities; *iv)* overlap recognition, which aims to resolve the conflicting cases of overlapping entities, where the system uses dynamic programming by choosing the best-scoring set of non-overlapping mention entity mappings. This resolution enables to improve the performance of the model consistently.

DataTXT [8] is the TagME version adapted for tweets. The first step is to tokenize and parse the tweet content to find and identify entities using a dictionary of entities related to Wikipedia pages. The best association *entity - Wikipedia page* is selected by computing a score based on an algorithm called *collective agreement* between each page associated to the first entity and all the other pages associated to the other entities found in the text. This score actually estimates the relatedness between two Wikipedia pages.

AIDA [9] has also released a system tailored for processing tweets. Before the extraction step, the tweet content is normalized thanks to some pre-processing rules. For example, *#edSnowden* is transformed into *Edward Snowden* and *@TheRealHowardW* is transformed into *Howard Wolowitz*. The normalized (short) text is then processed as it was a formal text.

These three methods are tailored to process tweets. Like for the four methods described in the Section 2.1, they do not recognize emerging entities as they are mostly based on dictionaries or trained model for formal text (e.g. AIDA).

2.3 Summary

We summarize this state-of-the art in the Tables 1 and 2, inspired from [1], where we show the similarities and differences of each system at extraction and linking level. In Table 2, the abbreviation *LEE* means *Link Emerging Entities*.

NEE (Named Entity Extraction)				
System	External Tools	Main Features	Method	Knowledge Base
E2E	-	N-Grams, stop words removal, punctuation as token	rule-based (candidate filter), dictionary	Wikipedia, Freebase
AIDA	StanfordNER	-	-	NER dictionary
TagME and DataTXT	-	N-Grams, overlap resolution, Wikipedia statistics	dictionary, link probability	Wikipedia
WAT	OpenNLP	N-Grams, Wikipedia statistics	dictionary, SVM	Wikipedia, NER dictionary
Babelfy	-	N-Grams, POS, superstring matching	dictionary	Babelnet
Spotlight	LingPipePOS	string matching, POS	Aho-Corasick, dictionary	DBpedia

Table 1. Named Entity Extraction analysis

NEL (Named Entity Linking)				
System	Main Features	Method	Knowledge Base	LEE
E2E	N-Grams, lower case, entity graph features, popularity based on clicks and visiting information on the Web	DCD-SSVM + MART gradient boosting	Wikipedia, Freebase	No
AIDA	popularity based on Wikipdia, similarity, coherence, densest subgraph	graph-based	YAGO2	No
TagME and DataTXT	mention-entity commonness, Wikipedia statistics	collective agreement, link probability, C4.5	Wikipedia	No
WAT	string similarity, commonness, context similarity, PageRank, Personalized PageRank, HITS, SALSA	graph-based	Wikipedia	No
Babelfy	densest subgraph	graph-based	Babelnet	No
Spotlight	TF*ICF	VSM, cosine similarity	DBpedia	No

Table 2. Named Entity Linking analysis

For the extraction, we observe that systems mainly use dictionaries based on a particular knowledge base (semantic-based approach). When POS tagging is being used, it is essentially a secondary feature which aims to enforce or to discard what has been extracted with the dictionary. Contrarily to the others, AIDA uses a pure NLP approach based on Stanford NER. TagME claims to make an overlap resolution between the extracted mentions at the end of this process. Our system tackles the problem using both a linguistic-based and a semantic-based approach of equal importance which demonstrates a higher performance at extraction level as detailed in the Section 4.

For the linking, we can see two main approaches: graph-based and arithmetic combination. Contrarily to the others, E2E [5] uses a pure machine learning approach using different features. At the end of this process, TagME and WAT do a pruning. None of these systems claims to be able to handle emerging entities, that is, disambiguation such entities to NIL. This is mainly due to their extraction approach. Our system tackles the problem using an arithmetic combination inspired from TagME, to detect and link emerging entities to NIL, while using a pruning process at the end for removing the false positives.

3 Architecture and Implementation

The architecture of our system is made for being modular. It is composed of different modules that can be switched on or off according to the nature of the text to be processed. A pipeline is a sequence of activated modules that will process an input text. Modules can have different goals: extraction, typing, recognition, linking or pruning. Those modules, according to their goal, have to respect some standards. Each module has to provide an output corresponding to a specific interface in order to be understood by the others modules. For example, in our approach, we have three extraction modules

(POS, NER and gazetteer) that should provide a similar output for being understood by the candidate generator module. This system eases the overlap resolution process as it is easier to compare outputs that follow the same presentation. Furthermore, each module is run independently from the others.

The candidate generation process is based on an index. If an extracted mention does not have an entry in the index, we normally link it to NIL following the TAC KBP convention[4]. The index is created on top of the DBpedia 2014 Knowledge Base[5] and a dump of the Wikipedia articles[6] dated from October 2014. Each record of the index has a key which corresponds to a DBpedia resource, while the features are listed in Table 3.

ID	Feature	Definition
1	title	the title of the entity
2	URI	the URI associated to the entity
3	redirects	the list of all the redirect pages associated to the entity
4	disambiguation	the title of the disambiguation pages associated to the entity if there is at least one
5	types	the full type hierarchy of the entity, from the highest to the fine-grained type
6	pageRank	the PageRank score of the DBpedia resource corresponding to the entity
7	hits	the HITS score of the DBpedia resource corresponding to the entity
8	inlinks	the number of inLinks of the DBpedia resource corresponding to the entity
9	outlinks	the number of outLinks of the DBpedia resource corresponding to the entity
10	length	the length in number of characters of the associated Wikipedia page of the entity
11	numRedirects	the number of redirects links associated to the entity
12	surfaceForms	the different surface forms used to call the entity in all the Wikipedia articles
13	quotes	the direct outbound links and the number of time they appear in the article of the corresponding entity.

Table 3. List of features contained in the index and used by the pruning algorithm

To create this index, we build an index using Lucene v5.2.1. The process requires 44 hours to be completed on a 64GB RAM, 20 core CPU at 2.5Ghz machine. The feature number 13 in Table 3 is made with an in-house library to parse the Wikipedia dump. We first tried several libraries that parse Wikipedia such as Sweble[7], GWTWiki[8] and wikipedia-parser[9]. However, these libraries are either too complex to use for the simple extraction we need or too greedy in terms of memory. We have therefore developed our own library in order to extract the pairs (Wikipedia article title, number of times the title appears in the article). For example, in the Wikipedia article of *Paris* we found 3

[4] http://nlp.cs.rpi.edu/kbp/2014/

[5] http://wiki.dbpedia.org/services-resources/datasets/datasets2014

[6] https://dumps.wikimedia.org/enwiki/

[7] http://sweble.org/

[8] https://code.google.com/p/gwtwiki/

[9] https://github.com/Stratio/wikipedia-parser

links to the Wikipedia article *Eiffel Tower*, resulting in the tuple *(Eiffel Tower, 3)*, and this tuple will be associated to the corresponding entry of the DBpedia resource *Paris* in the index.

Our system uses three different modules to extract mentions: POS, NER and gazetteer. Our POS tagging system is the Stanford NLP POS Tagger. We use two different models depending if we process microposts (e.g. tweets) or formal texts (e.g. newswire). The model used for microposts is `https://gate.ac.uk/wiki/twitter-postagger.html` that is trained specifically in order to be case insensitive and to get independent tags for mentions and hashtags. The one used for formal text is *english-bidirectional-distsim* that provides a better precision but for a higher computing time. We use the POS tagger to spot all the noun phrases and numbers. Our second module is the NER which relies on Stanford NER, trained with either the OKE2015 training set for formal text or trained with the #Micropost2014 training set for tweets. Moreover, we use NER to extract dates and other time related mentions. Our last module is the gazetteer that aims to reinforce the extraction stage bringing a robust spotting for well-known nouns such as abbreviations. Those three components are actually divided in five different modules: one POS module to extract only proper-nouns, one POS module to extract only numbers, one NER module without date spotting, one NER module with only date spotting and one gazetteer module. Finally, a sixth module has been developed to dereference mentions in microposts, for example, the mention @*TheRealHowardW* will be transformed into *Howard Wolowitz*.

Once all extractors have finished to extract mentions, we process their output with an overlap resolution module. This module takes two inputs and provides one output. It is run as many times as there are activated extractor modules minus one. For example, if a pipeline uses the extractor modules POS for proper nouns (POSNNP), the NER and gazetteer (GAZ) modules, then, the overlap resolution will first take the two inputs (POSNNP and NER) to provide one output (POSNNP-NER) and it will take once again two other inputs (POSNNP-NER and GAZ) to provide one final output (POSNNP-NER-GAZ). Therefore, when three different extractor modules are used, the overlap resolution module is run twice. We have developed such a module since, sometimes, at least two extractors provide overlapping mentions. For example, given the two extracted mentions *States of America* from the NER module and *United States* from the POS for proper noun module, we detect that there is an overlap between both mentions according to their offset in the processed sentence. We take the union of both boundaries to create a new mention and we remove the two others. We obtain the mention *United States of America* and the type provided by the NER module is selected. The mechanism is the same if one mention is included in another one. For example, if *United States* and *United States of America* are extracted, the later will be kept. Finally, there are cases of ambiguity. For example, with the text *Yesterday I went to Los Angeles, California*, *Los Angeles*, *California* and *Los Angeles, California* are three different valid extractions. We have decided to systematically keep the longest mention, in this case *Los Angeles, California*.

Once we have independent mentions, the candidate generation module searches the index to retrieve as many candidates as possible. This means that for each mention, we have potentially numerous candidates, while many of them have to be filtered out because they are most likely not related to the context. We use a filter module that creates a graph with all the candidates of each mention and find the densest graph between all of these candidates, similarly to [6]. Our approach is, however, slightly different: we use the feature number 13 (quotes) described in the Table 3 and not BabelNet in order to build the graph. The edges of the graph are weighted according to the number of occurrence of the link between each candidates. For example, given the Wikipedia article describing the Eiffel Tower, if there is one outbound link to Paris in Texas and three to Paris in France, both candidates (Paris in Texas and Paris in France) will be kept. However, the weight of Paris in France will be higher than the one of Paris in Texas. In case all candidates of a mention do not have any relation with any other candidate of the other mentions, all its candidates are kept.

Once we have reduced the number of candidates, we use a ranking module in order to score each of those candidates and to pick up the best one. This module is using a rank function $r(l)$ to compute, for each candidate, a score based on a string similarity measure between the extracted mention and the title of the candidate, the set of redirect and the set of disambiguation pages associated to this candidate and its PageRank.

$$r(l) = (a \cdot L(m, title) + b \cdot max(L(m, R)) + c \cdot max(L(m, D))) \cdot PR(l) \tag{1}$$

where the weights a, b and c have to follow the following hypothesis: $a + b + c = 1$ and $a > b > c$. We take the assumption that the string distance measure between a mention and a title is more important than the distance measure with a redirect page and is itself more important than the distance measure with a disambiguation page.

The last module is the pruning one, which is used to detect and remove the false positive annotations[10] in order to improve the precision of the system. We use a machine learning approach, with the algorithm k-NN. We have tried four different algorithms (Random Forest, Naive Bayes, SVM and k-NN), and have empirically assessed that k-NN generally provides the best results. To train this algorithm and getting a model, we use ten features, most of them are listed in Table 3: the extracted mention and the title, type, PageRank, HITS, inLinks, outLinks, length, redirectsNumber and $r(l)$ of the entity. The training method uses a four steps approach: 1) run our system on a training set; 2) classify entities as true or false according to the entities in the Gold Standard of the training set and the ones provided by the results of our system; 3) create a file with the features of each of these entities and their true / false classification; 4) train k-NN with this file that contains the features to get a model. Once the model has been created, we let k-NN classify each annotation provided by our system to true or false.

4 Evaluation

Our hybrid approach has been benchmarked against the test dataset of the #Micropost2014 NEEL challenge and the test dataset of the OKE2015 challenge. An example of #Micropost2014 and OKE2015 datasets are provided in Table 4.

[10] We consider an *annotation* as the tuple (mention, entity)

4.1 Experimental Settings

We configured two different pipelines, one for each dataset. For OKE2015, we used the following modules: three extractors (POS for proper nouns, NER without date and gazetteer), the index lookup, the filtering and the scoring. For #Micropost2014, we used six extractors (POS for proper nouns, POS for numbers, NER with only dates, NER without dates, gazetteer and Twitter account dereferencing), the index lookup, the filtering and the scoring. We have run these two pipelines with and without the pruning module in order to assess its performance. This shows the advantages of our hybrid approach of being able to adapt an entity linking process by adding or removing modules (methods) and combining them.

dataset	text	links
#Microposts2014	**Murdoch** unable to answer who the top legal officer at **News International** was. (via: http://mm4a.org/q3rx06) #p2 **#notw #hackgate**	db:Rupert_Murdoch db:News_UK db:News_of_the_World db:News_International_phone_hacking_scandal
OKE2015	The **man** served on various committees and spent time at the **Royal Aircraft Establishment** in **Farnborough**, where **he** made a major contribution to the design of the Mark XIV bomb sight which allowed bombs to be released without a level bombing run beforehand.	NIL db:Royal_Aircraft_Establishment db:Farnborough,_Hampshire NIL

Table 4. Examples from OKE2015 and #Micropost 2014 test datasets

4.2 Results

Breakdown figures for the #Micropost2014 NEEL challenge with and without the pruning reported in Table 5 have been computed with the official scorer of the challenge[11]. In the breakdown figures, the results at the recognition level is not presented since typing entities was not required by the challenge. Breakdown figures for the OKE2015 challenge with and without the pruning reported in Table 6 have been computed with the neleval scorer[12].

The Table 7 shows the performance of our approach in comparison to other systems, using the F1-measure at the final linking stage. Results for #Micropost2014 for TagME (more precisely DataTXT), AIDA, E2E and UTwente are coming from the official results of the challenge, while Babelfy and DBpedia Spotlight have not been tested as they are not made for processing tweets. We refer the reader to [1] for the complete results of

[11] https://github.com/giusepperizzo/neeleval

[12] https://github.com/wikilinks/neleval

	Without Pruning			With Pruning		
Task	P	R	F_1	P	R	F_1
Extraction	69.17	72.51	70.80	70	41.62	52.2
Linking	47.39	45.23	46.29	48.21	26.74	34.4

Table 5. Breakdown figures on the #Micropost2014 challenge test set with and without the pruning.

	Without Pruning			With Pruning		
Task	P	R	F_1	P	R	F_1
Extraction	78.2	65.4	71.2	83.8	9.3	16.8
Recognition	65.8	54.8	59.8	75.7	8.4	15.1
Linking	49.4	46.6	48	57.9	6.2	11.1

Table 6. Breakdown figures on the OKE challenge training set with and without the pruning.

all systems having participated in the 2014 NEEL challenge while we only reproduce in Table 7 the best performing systems. For the OKE2015 challenge, the results have been computed with the neleval scorer.

	Our Approach	DBPedia Spotlight	TagME (DataTXT)	Babelfy	WAT	AIDA	E2E	UTwente
#Microposts2014	46.29	*N/A*	49.9	*N/A*	*N/A*	45.37	**70.06**	54.93
OKE2015	**48**	39.8	37.9	42	*N/A*	44.6	*N/A*	*N/A*

Table 7. F1-measure results at the linking stage on both the #Microposts2014 NEEL challenge and OKE2015 challenge test datasets

The results that are reported for DBpedia Spotlight, TagME, AIDA and Babelfy for OKE2015 have been obtained using their respective APIs with the best tested settings. For DBpedia Spotlight, those settings are: *confidence=0.3* and *support=20*. For TagME, those settings are: *include_all_spots=yes* and *epsilon=0.5*. For AIDA, those settings are: *technique=GRAPH*, *algorithm=COCKTAIL_PARTY_SIZE_CONSTRAINED*, *alpha=0.6* and *coherence=0.9*. For Babelfy, those settings are: *lang=en*, *annType=NAMED_ENTITIES*, *annRes=WIKI*, *match=EXACT_MATCHING*, *dens=true* and *th=0.4*. WAT is not publicly available and could not be tested with the OKE challenge dataset.

Our approach outperforms those systems when analyzing formal texts. E2E and UTwente still perform significantly better than our approach when processing microposts while we achieve similar results than TagME (DataTXT).

5 Error Analysis

For #Micropost2014, an error analysis shows that our approach misses 490 entities and finds 578 additional entities on a total of 1460 entities contained in the test dataset. Amongst the most common errors, our approach has problems with entities which do not contain tokens tagged as proper nouns but as noun (e.g. phone hacking). There is also a problem with the possessive endings (e.g *Beyonce's* for *Beyonce*). Syntactically speaking, a noun phrase may contain a noun phrase, which provides duplicates. For example, in the sentence *She gets into the action, overlapping Lloyd*, the noun phrase *the action, overlapping Llyod* can be split in two noun phrases, which are *the action* and *overlapping Lloyd*. The extraction may lead to these three cases.

For OKE2015, an error analysis shows that our approach misses 207 entities and find 167 additional entities on a total of 594 entities contained in the test dataset. The most common error for this challenge is that our approach does not resolve the co-references while the test dataset contained a lot of them. The training set was not big enough to properly train Stanford NER. This is why the correct mention is often extracted but is given a wrong type. The test dataset contained errors in the gold standard, some of them have been fixed by us, but some others need a closer work with the organizers to be resolved.

We observe that there is still a large margin of progress to reduce the performance drop between the results at the recognition or extraction stage and the final results at the linking stage. This drop can be explained by the score function and the candidates filter used at the linking stage. For the pruning, we can see that it increases a bit the precision but at the cost of significantly decreasing the recall. It overall performs poorly as it removes too many (correct) mentions to get good results at the linking stage. This idea has been inspired by the WAT system [7]. However, some features we choose differ from the ones used in WAT resulting in this serious performance drop. We stay positive on the fact that a pruning step can typically help increasing the precision when a real high recall at the recognition or extraction level is obtained.

6 Conclusion and Future Work

The results are encouraging and show that this hybrid approach which exploits both linguistic and semantic features can achieved the expected behavior. As future work, we plan to improve the linking stage by making more use of graph-based algorithms with more accurate ranking functions and filtering. Another future work is to further develop our pruning strategy by reviewing the list of feature used to show the full potential of our hybrid approach. We finally aim to optimize the creation of the index by parallelizing the process and to multiply the number of knowledge bases on which entities can be disambiguated against.

Acknowledgments

This work was partially supported by the innovation activity 3cixty (14523) of EIT Digital (`https://www.eitdigital.eu`).

References

1. A. E. Cano, G. Rizzo, A. Varga, M. Rowe, Stankovic Milan, and A.-S. Dadzie. Making Sense of Microposts (#Microposts2014) Named Entity Extraction & Linking Challenge. In *4th International Workshop on Making Sense of Microposts*, Seoul, South Korea, 2014.
2. L. Chenliang, W. Jianshu, H. Qi, Y. Yuxia, D. Anwitaman, S. Aixin, and L. Bu-Sung. TwiNER: Named Entity Recognition in Targeted Twitter Stream. In *35th International Conference on Research and Development in Information Retrieval (SIGIR)*, 2012.

3. J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust Disambiguation of Named Entities in Text. In *8th Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 782–792, Stroudsburg, PA, USA, 2011.
4. P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. DBpedia Spotlight: Shedding Light on the Web of Documents. In *7th International Conference on Semantic Systems (I-Semantics)*, pages 1–8, 2011.
5. C. Ming-Wei, H. Bo-June, M. Hao, L. Ricky, and W. Kuansan. E2E: An End-to-End Entity Linking System for Short and Noisy Text. In *Making Sense of Microposts (# Microposts2014)*, 2014.
6. A. Moro, A. Raganato, and R. Navigli. Entity Linking meets Word Sense Disambiguation: a Unified Approach. *TACL*, 2:231–244, 2014.
7. F. Piccinno and P. Ferragina. From TagME to WAT: a new entity annotator. In *1st ACM International Workshop on Entity Recognition & Disambiguation (ERD)*, pages 55–62, Gold Coast, Australia, 2014.
8. U. Scaiella, M. Barbera, S. Parmesan, G. Prestia, E. Del Tessandoro, and M. Verì. DataTXT at# Microposts2014 Challenge. In *Making Sense of Microposts (# Microposts2014)*, 2014.
9. M. A. Yosef, J. Hoffart, Y. Ibrahim, A. Boldyrev, and G. Weikum. Adapting AIDA for Tweets. In *Making Sense of Microposts (# Microposts2014)*, 2014.