Enhancing Entity Linking by Combining NER Models

Julien Plu[1], Giuseppe Rizzo[2], Raphaël Troncy[1]

[1] EURECOM, Sophia Antipolis, France
julien.plu|raphael.troncy@eurecom.fr
[2] ISMB, Turin, Italy
giuseppe.rizzo@ismb.it

Abstract. Numerous entity linking systems are addressing the entity recognition problem by using off-the-shelf NER systems. It is, however, a difficult task to select which specific model to use for these systems, since it requires to judge the level of similarity between the datasets which have been used to train models and the dataset at hand to be processed in which we aim to properly recognize entities. In this paper, we present the newest version of ADEL, our adaptive entity recognition and linking framework, where we experiment with an hybrid approach mixing a model combination method to improve the recognition level and to increase the efficiency of the linking step by applying a filter over the types. We obtain promising results when performing a 4-fold cross validation experiment on the OKE 2016 challenge training dataset. We also demonstrate that we achieve better results that in our previous participation on the OKE 2015 test set.

Keywords: Entity Recognition, Entity Linking, Entity Filtering, Model Combination, OKE Challenge, ADEL

1 Introduction

The 2016 Open Knowledge Extraction challenge (OKE2016) aims to: i) identify entities in a sentence, ii) link such entities, when it is possible, to a DBpedia resource and iii) assign a type to these entities selected from a set of given types. In this paper, we present our participation to this challenge using a newer version of the ADEL framework that extends our approach presented last year at OKE2015 [9]. A first improvement concerns the framework architecture which has became more modular: external NLP systems are used via a REST API and the knowledge base index uses more sophisticated tools (Elastic and Couchbase) while being built from additional data format (e.g. TSV). A second improvement concerns the way our Stanford-based named entity recognition module can be used, in particular, using multiple models.

This paper mainly focuses on entity recognition, which refers to jointly performing the appropriate extraction and the typing of mentions. *Extraction* is the task of spotting mentions that can be entities in the text while *Typing* refers to the task of assigning them a proper type. Linking is the last step of our approach, and it refers to the disambiguation of mentions in a targeted knowledge base. It is also often composed of two subtasks: generating candidates and ranking them accordingly to various scoring functions. Following the challenge requirements, we make use of the 2015-04 snapshot of DBpedia as the targeted knowledge base.

The remainder of this paper is structured as follows. We first describe some recent related work, emphasizing the usage of external NLP systems when performing entity recognition (Section 2). Next, we detail the newest architecture of ADEL (Section 3). We propose two experiment settings among many variants in order to highlight the importance of a pre-processing step (Section 4). We detail our preliminary results on the OKE2016 challenge training dataset using a 4-fold cross-validation setup and we also measure the improvements of ADEL on the 2015 test set (Section 5). Finally, we conclude and outline some future work in Section 6.

2 Related Work

Several entity linking systems use an external named entity recognition tool such as Stanford NER [2] or the Apache OpenNLP Name Finder[3]. For example, the popular AIDA[4] system makes use of Stanford NER trained on the CoNLL2003 dataset [4]. In [6], the authors also use Stanford NER but without saying which specific model is being used. In [4], the authors use Stanford NER in a similar way than in [5]. Finally, the FOX tool proposes an ensemble learning method over Stanford NER and other NER classifiers (such as OpenNLP, Illinois Named Entity Tagger and Ottawa Baseline Information Extraction). The authors use a model trained again on the CoNLL2003 dataset for each sub-classifier [10]. In terms of architecture, while all these systems use external NER systems, they integrate them only internally, using the provided Java API directly in their source code. This kind of integration makes difficult the possibility of re-configuring those external NLP systems, or switching between different ones.

We first hypothesize that the CoNLL2003-based model for recognizing entities for a type of text than differs from a newswire article will not be optimal [8]. Therefore, we will propose an architecture that enables to use multiple models. We will also promote a flexible way of interacting with NER components via a standard API.

Several entity linking systems advocate a so-called *E2E* (End-to-End) approach. This method uses only a semantic network of an entity catalogue to extract mentions from the text and to generate candidate links. The limitation with this method is therefore its ability to extract emerging entities, since entities that are not present in the catalogue will not be extracted and disambiguated. Our hybrid approach overcomes this problem since we do not only use a catalogue of entities but also a POS and a NER tagger for extracting entities, thus mixing semantic and NLP-based methods. Let's take an example coming from the OKE2016 dataset with the sentence: *James Tobin married Elizabeth Fay Ringo, a former M.I.T. student of Paul Samuelson, on September 14, 1946.* In this sentence, the entities to be extracted are: *James Tobin*, *Elizabeth Fay Ringo*, *M.I.T.*, *student* and *Paul Samuelson*. Most of those entities can be extracted and linked to DBpedia via an E2E approach except *Elizabeth Fay Ringo* which does not exist in DBpedia yet. TagME [1] is a popular system implementing an *E2E* approach that provides a public API[5]. TagME will not effectively extract the entity *Elizabeth Fay Ringo* from this sentence, contrarily to our ADEL framework.

[3] https://opennlp.apache.org/
[4] https://github.com/yago-naga/aida
[5] http://tagme.di.unipi.it

3 Architecture

The goal of our system is to link all the mentions occurring in a text to their counterparts in a knowledge base. Emerging entities, *i.e.* entities that are not present in a knowledge base, will be linked to *NIL*. ADEL comes with a brand new architecture compared to the version we have proposed in the previous edition of this challenge [9]. This architecture is composed of multiple modules spread into two main parts (Figure 1). The first part (*Entity Extraction*) contains the modules *Extractors* and *Overlap Resolution*. The second part (*Entity Linking*) contains the modules *Indexing*, *Candidate Generation*, *NIL Clustering* and *Linkers*.

3.1 Entity Extraction

In this section, we describe how we extract mentions from texts that are likely to be selected as entities with the *Extractor Module*. After having identified candidate mentions, we resolve their potential overlaps using the *Overlap Resolution Module*.

Extractors Module. We make use of three kinds of extractors: *i)* Dictionary, *ii)* POS Tagger and *iii)* NER. Each of these extractors run in parallel. At this stage, an entity dictionary reinforces the extraction by bringing a robust spotting for well-known proper nouns or mentions that are too difficult to be extracted for the other extractors (e.g. *Role-type* mentions). The two other extractors use an external NLP system based on Stanford CoreNLP [7] and particularly the POS [12] and NER taggers.

We have developed a generic *NLP System REST API* wrapper to use the Stanford CoreNLP system. This wrapper has been designed while keeping in mind the core idea of ADEL, namely *adaptivity*. Hence, this module gives the possibility to use any other NLP system such as the ones used in [10] or even systems tailored for other languages than English. The REST API provides annotations in the NIF format [3]. Therefore, by using this module, one can switch from one NLP system to another one without changing anything in the code or can combine different systems. This module enables as well to save computing time since all models being used are loaded only once at startup. A configuration file enables to parametrize how to use Stanford CoreNLP. During our tests on the OKE2016 dataset, we used the *english-bidirectional-distsim* model that provides a better accuracy but for a higher computing time for the POS tagger[6].

[6] `http://nlp.stanford.edu/software/pos-tagger-faq.shtml#h`

Contrarily to [4], [6], [10] and [4], we use a model combination method that consists of combining different CRF models as described in the Algorithm 1.

Algorithm 1: Combining multiple CRF models

```
Result: Annotated tokens
Input  : $(Txt, M)$ with $Txt$ the text to be annotated and $M$ a list of CRF models
Output: $A = List(\{token, label\})$ a list of tuples $\{token, label\}$
1  begin
2      $finalTuples \leftarrow EmptyList()$;
3      foreach $model$ in $M$ do
           /* $tmpTuples$ contains the tuples $\{token, label\}$ got
              from $model$                                          */
4          $tmpTuples \leftarrow$apply $model$ over $Txt$;
5          foreach $\{token, label\}$ in $tmpTuples$ do
6              if $token$ from $\{token, label\}$ not in $finalTuples$ then
7                  add $\{token, label\}$ in $finalTuples$;
8              end
9          end
10     end
11 end
```

The Algorithm 1 shows that the order in which the models are applied is important. Hence, if a token is badly labeled by the first model, the second model cannot correct it even if it would have given the correct label in the first place. This algorithm in Stanford NER is called *NER Classifier Combiner*. An implementation of the Stanford CoreNLP as provided by our *NLP System REST API* is available on Github[7].

Overlap Resolution Module. The extractors can extract mentions that have a partial or a full overlap with others. To resolve this ambiguity, we implement an *overlap resolution* module that takes the output of each component of the extractors module and gives one output without overlaps. The logic of this module is the following: given two overlapping mentions, *e.g.* `States of America` from Stanford NER and `United States` from Stanford POS tagger, we only take the union of the two phrases. We obtain the mention `United States of America` and the type provided by Stanford NER is selected.

3.2 Entity Linking

In this section, we describe how we disambiguate candidate entities coming from the extraction step (Section 3.1). First, we create an index over a targeted knowledge base, *e.g.* the April 2015 DBpedia snapshot, using the *Indexing Module*. This index is used to select possible candidates with the *Candidate Generation Module*. If no candidates are provided, this entity is passed to the *NIL Clustering Module*, while if candidates are retrieved, they are given to the *Linkers Module*.

Indexing Module. Previously, we were using an index stored in Lucene. We have, however, observed unexpected behavior from Lucene such as not retrieving resources

[7] `https://github.com/jplu/stanfordNLPRESTAPI`

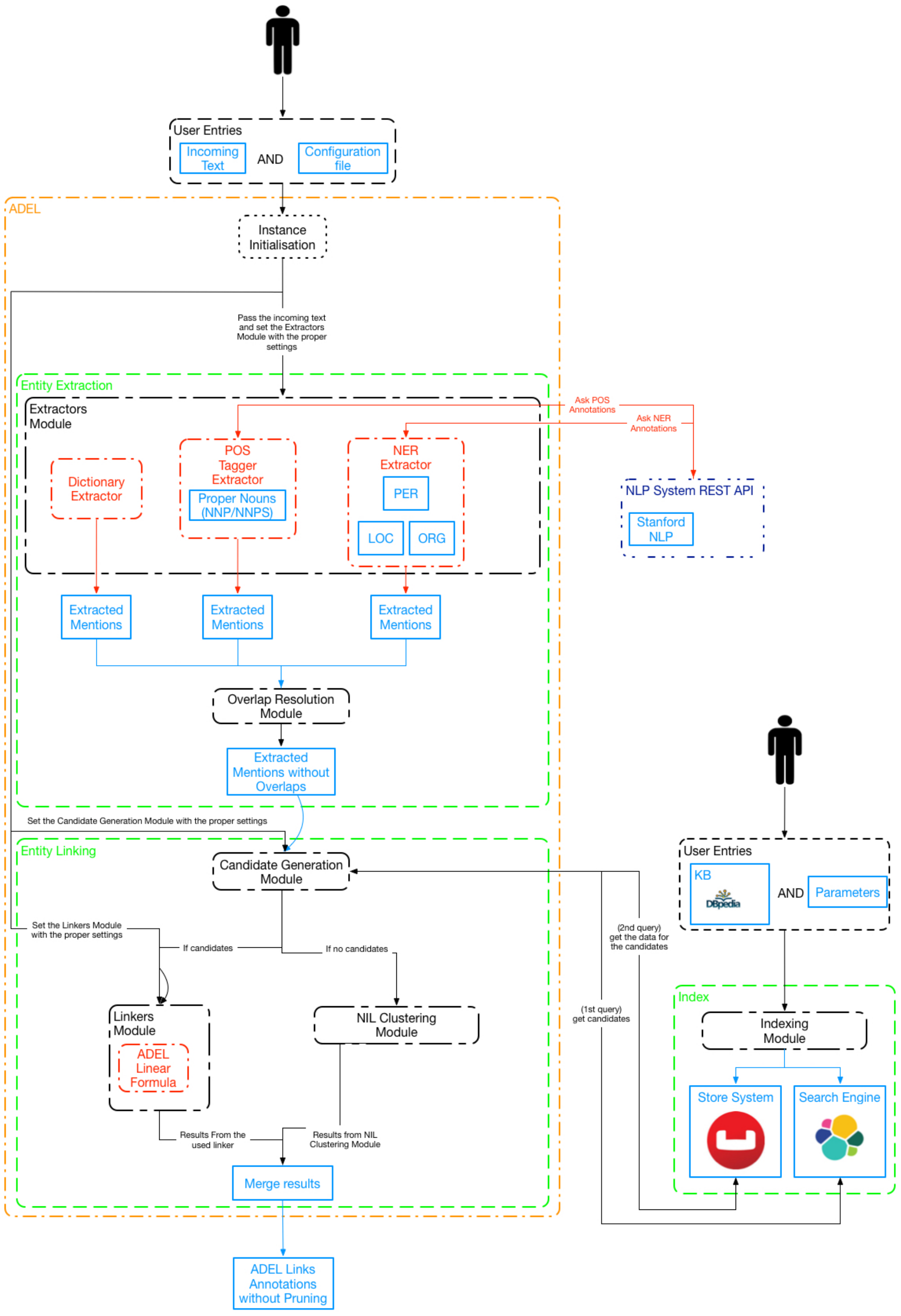

Fig. 1: ADEL new architecture

that match partially a query even when not bounding the number of results. The index is now built using *Elastic* as a search engine and *Couchbase* as data storage. First, we query *Elastic* to get the possible candidates. Second, we query *Couchbase* to get the data associated with these possible candidates. The index is built on top of both DBpedia2015-04[8] and a dump of the Wikipedia articles[9] dated from February 2015.

Candidate Generation Module. This module is querying *Elastic* and *Couchbase* to get possible candidates for the entities coming from the extraction module (Section 3.1). If this module gets candidates for an entity, they are given to the *Linkers Module*; if not, they are given to the *NIL Clustering Module*.

NIL Clustering Module. We propose to group the *NIL* entities (emerging entities) that may identify the same real-world thing. The role of this module is to attach the same *NIL* value within and across documents. For example, if we take two different documents that share the same emerging entity, this entity will be linked to the same *NIL* value. We can then imagine different *NIL* values, such as *NIL_1*, *NIL_2*, and so on so forth.

Linkers Module. This module implements an empirically assessed function that ranks all possible candidates given by the *Candidate Generation Module*:

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

The function $r(l)$ is using the Levenshtein distance L between the mention m and the title, the max distance between the mention m and every element (title) in the set of Wikipedia redirect pages R and the max distance between the mention m and every element (title) in the set of Wikipedia disambiguation pages D, weighted by the PageRank PR, for every entity candidate l. The weights a, b and c are a convex combination that must satisfy: $a + b + c = 1$ and $a > b > c > 0$. 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 which is itself more important than the distance measure with a disambiguation page.

4 Experiment Settings and Pre-Processing

In this section, we aim to demonstrate the added-value of combining NER models for improving the named entity recognition performance. We have set up two distinct experiments, using either a single CRF model (Section 4.1) or multiple ones (Section 4.2), and that also highlight the importance of doing a proper data pre-processing and training. We performed a lightweight error analysis in the Section 4.3 that justifies this pre-processing step.

4.1 Experiment 1: no Role, one single CRF model

In the first experiment, we *i)* pre-process the training set by removing all the occurrences of the Role type, and *ii)* train a single CRF model with Stanford NER using the OKE

[8] `http://wiki.dbpedia.org/services-resources/datasets/datasets2015-04`

[9] `https://dumps.wikimedia.org/enwiki/`

2016 training set, that will be used via the *CRF Classifier* feature. We discard the Role type on purpose as explained in the Section 4.3.

4.2 Experiment 2: no Role, multiple CRF models

In the second experiment, we perform the same pre-processing step as in the first experiment, but we make use of the *NER Classifier Combiner* feature instead of *CRF Classifier*. This feature allows to combine multiple CRF models to annotate a text where combining *model1* and *model2* means that *model1* will first be applied, followed by *model2*. Two combination options are available: if the option *ner.combinationMode* is set to *NORMAL* (the default option), any label applied by *model1* cannot be applied by subsequent models. For instance, if *model1* applies the LOCATION tag, no other model's LOCATION tag will be used; if *ner.combinationMode* is set to *HIGH_RECALL*, this limitation will be deactivated.

In our experiments, we use the *HIGH_RECALL* combination mode with two CRF models: *english.all.3class.distsim.crf.ser.gz* trained over the PERSON, LOCATION and ORGANIZATION types from the CoNLL 2003 [11], MUC6, MUC7 and ACE 2002 datasets which is provided by default in the Stanford NER package, and a specific model trained from the OKE2016 dataset.

In order to illustrate this model combination functionality, let's take the following sentence from the OKE2016 dataset: *Martin Luther King then began doctoral studies in systematic theology at Boston University and received his Ph.D. degree on June 5, 1955, with a dissertation on "A Comparison of the Conceptions of God in the Thinking of Paul Tillich and Henry Nelson Wieman"*. First, the *english.all.3class.distsim.crf.ser.gz* model is applied, yielding to the extraction of the candidate entities *Martin Luther King (PERSON)*, *Boston University (LOCATION)*, *Paul Tillich (PERSON)* and *Henry Nelson Wieman (PERSON)*. Next, the OKE2016 model is applied. The combination mode prevents to re-label those candidate entities, but the second model will extract *God (PERSON)*. The entities coming from the two models are then merged without possible conflicts.

4.3 Error Analysis

We have performed three other variants from the previous two experiments as follows:

1. keep the Role type annotations from the training set and use either one CRF model or multiple ones;
2. vary the experiment 2 setup using the *NORMAL* combination mode;
3. vary the experiment 2 setup using all models provided by default in the Stanford NER package.

These variants consistently yield to a drop of performance, in terms of computing time or in terms of recognition results.

The first variant provides worst extraction and recognition results. This can be explained by the lack of sufficient occurrences of the Role type in the training dataset which is too small. Consequently, the results of the first variant have a high precision, but a lower recall for the Role recognition. The second variant provides a performance drop in terms of computing time, without giving more annotations compared to the *HIGH_RECALL* mode. This can be explained by the fact that there is no overlap between the types provided by the OKE model and the ones from the *english.all.3class.distsim.crf.ser.gz* model. Actually, the *NER Classifier Combiner* feature is case sensitive which means that the PERSON type (from CoNLL) is different from the Person type (from OKE). The third variant provides also a drop of performance in terms of computing time, since we obtain the same results while performing much larger computation.

Consequently, we pre-process the training dataset by discarding the Role type when training the NER module and we rely on a dictionary for extracting the Role type entities. This dictionary is built by using a list of the names of all the jobs existing in Wikipedia. A comparison of the pure linguistic approach (Stanford NER) and our system in terms of recognition is shown in Section 5.3 to demonstrate the advantage of using multiple extractors.

5 Results

In this section, we provide preliminary results of our ADEL framework on a 4-fold cross validation experiment using the 2016 OKE challenge training dataset. We use two different scorers: *conlleval*[10] and *neleval*[11]. We did not use the official scorer of the challenge (GERBIL [13]) since it cannot provide breakdown figures per entity type, like the conlleval scorer, or per sub-task (extraction, recognition, linking), like the neleval scorer. We have computed the NER results using the conlleval scorer to get the breakdown results per entity type (Person, Role, Organization and Place). We have computed the NEL results using the neleval scorer to get the breakdown results per sub-task. The differences in terms of figures between the conlleval and the neleval scorers can be explained by the fact that conlleval does not count the entities without a type (entities coming from the POS extractor and linked to NIL), which consider them either as false negative or true negative. These entities are counted by default as false positive in recognition for the neleval scorer.

5.1 Statistics of the Oracle

The training dataset provided by the OKE2016 challenge organizers is composed of a set of 196 annotated sentences using the NIF ontology[12]. The average length of the sentences is 155 chars. In total, the dataset contains 1043 mentions corresponding to 719 distinct entities that belong to one of the four types: `dul:Place`, `dul:Person`, `dul:Organization` and `dul:Role`. 565 entities (78.58%) are linked within DBpedia, while 153 (21.28%) are not. The breakdown of those annotations per type is provided in Table 1.

We applied a 4-fold cross validation on the training set. In each fold of the cross validation, a train and a test sets are generated and respectively used for building the supervised learning models and for benchmarking the output of the model with the expected results of the test set.

[10] `http://bulba.sdsu.edu/~malouf/ling681/conlleval`
[11] `https://github.com/wikilinks/neleval`
[12] `http://persistence.uni-leipzig.org/nlp2rdf/ontologies/nif-core#`

type	nb mentions	nb entities	nb mentions disambiguated (%)	nb entities disambiguated (%)
dul:Place	182	145	171 (93.96%)	134 (92.41%)
dul:Person	458	253	350 (76.42%)	164 (64.82%)
dul:Organization	237	212	198 (83.54%)	177 (83.49%)
dul:Role	166	109	145 (87.35%)	90 (82.57%)
Total	1043	719	864 (82.84%)	565 (78.58%)

Table 1: Statistics of the OKE 2016 training dataset

5.2 Results of Experiment 1 and Experiment 2 on the Training Set

Table 2-a shows the results of the NER extractor following the Experiment 1 setup (Section 4.1) computed with the conlleval scorer. We observe a higher recognition score for the type *Person* than for the two other types. This can be explained by the fact that there are twice more entities of type *Person* than *Place* and *Organization* in the training dataset.

Type	Precision	Recall	F-measure
Organization	67.05	58.58	62.42
Person	88.09	85.73	86.87
Place	69.75	68.69	69.16
Total	78.86	74.75	76.75

(a)

Type	Precision	Recall	F-measure
Organization	88.35	80.10	83.97
Person	92.11	93.50	92.73
Place	78.03	78.58	77.83
Total	88.23	85.37	86.75

(b)

Table 2: Results of the Experiment 1 (a) and Experiment 2 (b)

Table 2-b shows the results of the NER extractor following the Experiment 2 setup (Section 4.2) computed with the conlleval scorer. We observe a significant improvement in terms of recognition compared to the previous experiment. Nevertheless, the results for the type *Place* is still lower than for the other types. This can be explained by the fact that the datasets used to train the *english.all.3class.distsim.crf.ser.gz* model and the dataset from OKE2016 do not share the same definition of what is a *Place*. For example, the mention *Poughkeepsie, New York* is a single entity in the OKE2016 dataset, but correspond to two entities in the datasets used to train the *english.all.3class.distsim.crf.ser.gz* model.

5.3 NER Results of ADEL

As we have described in the section 3.1, ADEL makes use of multiple extractors (dictionary-based, POS and NER based) followed by an overlap resolution module. This hybrid approach provides the final results presented in the Table 3. We observe a general improvement and a particular high recognition of the type *Role* due to the specialized dictionary extractor. The results for the type *Person* are a little bit lower than in Experiment 2. This is due to some false positive extracted by the POS tagger extractor.

Type	Precision	Recall	F-measure
Organization	85.90	82.72	84.22
Person	91.27	93.27	92.10
Place	77.13	81.42	78.82
Role	95.23	98.65	96.84
Total	87.85	88.91	88.36

Table 3: Final ADEL results at the recognition stage

5.4 NEL Results of ADEL

We use the *neleval* scorer for computing results at the linking stage. More precisely, we consider the *strong_mention_match*, *strong_typed_mention_match* and *strong_link_match* scores. The first score corresponds to a strict mention extraction. The second score corresponds to a strict mention extraction with the good type. The third score corresponds to a strict mention extraction with the good link. Considering that ADEL performs relatively well for extracting and recognizing entities, we assume that the candidate links that do not have a type corresponding to the one assigned by ADEL are likely to not be good candidates and should therefore be filtered out. We applied this simple heuristic yielding to improved results (Table 4).

Level	Precision	Recall	F-measure	Precision	Recall	F-measure
strong_mention_match	81.0	88.7	84.7	81.0	88.7	84.7
strong_typed_mention_match	78.1	85.4	81.6	78.1	85.4	81.6
strong_link_match	45.2	57.4	50.5	57.4	55.7	56.5
	no filter used			a type filter is being used		

Table 4: ADEL results at the linking stage: a) without using a type filter and b) using a type filter

We finally compare the previous version of ADEL (v1) used in 2015 with the newer version of ADEL (v2) presented in this paper. For both versions, we use the OKE 2015 training set for training the NER extractors and we use the OKE 2015 test set for evaluation (Table 5). We observe a relative gain of 18.75% which is largely due to the novel model combination feature detailed in this paper.

	ADEL-v1			*ADEL-v2*		
Level	**Precision**	**Recall**	**F-measure**	**Precision**	**Recall**	**F-measure**
strong_mention_match	78.2	65.4	71.2	85.1	89.7	87.3
strong_typed_mention_match	65.8	54.8	59.8	75.3	59.0	66.2
strong_link_match	49.4	46.6	48	85.4	42.7	57.0

Table 5: Comparison between ADEL-v1 and ADEL-v2 over the OKE 2015 test set

6 Conclusion and Future Work

In this paper, we have showed the benefit of combining different models to improve the entity recognition, and use it as a filter to also improve the linking. We can also say that the provided dataset is not fairly distributed in terms of type accordingly to the Table 1. The type distribution problem is revealed by the difficulties to properly recognize the *Place* or *Role* types by using only the OKE2016 dataset. This dataset also contains and links co-references and anaphora. A co-reference denotes a situation where two or more expressions refer to the same entity in a same text, for example *Look at that man over there; he is wearing a funny hat*. An anaphora is when pronouns are used to link to an antecedent but this antecedent does not refer to any entity in the same text, for example *No man said he was hungry*. Then, we can say that anaphora is the generic term and co-reference is a specific kind of anaphora. Our system does not take into account this syntactic particularity, a possible future work would be to include a module to extract and link such syntactic particularity.

Acknowledgments

This work was partially supported by the innovation activity 3cixty (14523) of EIT Digital and by the European Union's H2020 Framework Programme via the FREME Project (644771).

References

1. P. Ferragina and U. Scaiella. TAGME: on-the-fly annotation of short text fragments (by wikipedia entities). In *19th ACM Conference on Information and Knowledge Management (CIKM)*, 2010.
2. J. Finkel, T. Grenager, and C. Manning. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *43rd Annual Meeting on Association for Computational Linguistics*, 2005.
3. S. Hellmann, J. Lehmann, S. Auer, and M. Brümmer. Integrating NLP Using Linked Data. In *12th International Semantic Web Conference (ISWC)*, 2013.
4. J. Hoffart, Y. Altun, and G. Weikum. Discovering Emerging Entities with Ambiguous Names. In *23rd World Wide Web Conference (WWW)*, 2014.
5. 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 *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2011.

6. X. Ling, S. Singh, and D. S. Weld. Design Challenges for Entity Linking. *Transactions of the Association for Computational Linguistics TACL*, 2015.
7. C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, 2014.
8. J. Plu. Knowledge Extraction in Web Media: At The Frontier of NLP, Machine Learning and Semantics. In *25th World Wide Web Conference (WWW), PhD Symposium*, 2016.
9. J. Plu, G. Rizzo, and R. Troncy. A Hybrid Approach for Entity Recognition and Linking. In *12th European Semantic Web Conference (ESWC), Open Knowledge Extraction Challenge*, 2015.
10. R. Speck and A.-C. Ngonga Ngomo. Ensemble Learning for Named Entity Recognition. In *13th International Semantic Web Conference (ISWC)*, 2014.
11. E. F. Tjong Kim Sang and F. D. Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *17th Conference on Computational Natural Language Learning (CoNLL)*, 2003.
12. K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich Part-of-speech Tagging with a Cyclic Dependency Network. In *Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, 2003.
13. R. Usbeck, M. Röder, A.-C. Ngonga Ngomo, C. Baron, A. Both, M. Brümmer, D. Ceccarelli, M. Cornolti, D. Cherix, B. Eickmann, P. Ferragina, C. Lemke, A. Moro, R. Navigli, F. Piccinno, G. Rizzo, H. Sack, R. Speck, R. Troncy, J. Waitelonis, and L. Wesemann. GERBIL: General Entity Annotator Benchmarking Framework. In *24th World Wide Web Conference (WWW)*, 2015.