Retrieval based approach over embeddings of Open Data samples for TAC RUFES 2020

Jaime Collado-Montañez
CEATIC
Universidad de Jaén
23071 - Jaén (Spain)
jcollado@ujaen.es

Arturo Montejo-Ráez
CEATIC
Universidad de Jaén
23071 - Jaén (Spain)
amontejo@ujaen.es

Miguel Á. García-Cumbreras
CEATIC
Universidad de Jaén
23071 - Jaén (Spain)
magc@ujaen.es

Abstract

This paper describes the approach of the SINAI team to fine-grained NER on TAC RUFES 2020 task. The proposed system enriches the ontology of entities with examples collected from different knowledge bases like Yago, WikiData and DBPedia. For each entity category, a template is used to embed the examples and generate ELMo vectors. K-nearest neighbor algorithm is used to match a candidate vector against all example vectors, so the most frequent category is chosen as final label for the candidate term. This approach is a semi-supervised approach relying on data augmentation via knowledge databases. Our results show that this can be a promising solution for certain categories as being applied to other fine-grained NER problems.

1 Introduction

The objective of TAC RUFES (Recognizing Ultra Fine-grained Entities) is to identify predefined entity types in any language and to classify or group mentions of the same entity. TAC RUFES continues, somehow, with the challenges proposed in previous editions of TAC KBP [Getman et al., 2018]. This RUFES task challenges systems to recognize name, nominal, and pronominal mentions of entities in news articles, from a newly developed ontology with about 200 fine-grained entity types to be discovered in a corpus of news texts. As described in the task, given an input document, a system is required to automatically identify an entity as a cluster of name, nominal, and/or pronominal mentions, and classify the entity into one or more of the types defined in the ontology.

This paper presents the submissions of the SINAI team to the RUFES Task of the TAC 2020 evaluation. In our first participation the highlights of our work are i) the data augmentation (a pre-trained FastText model [Bojanowski et al., 2017] and different knowledge bases like Yago[1] [Suchanek et al., 2008], WikiData[2] [Vrandečić and Krötzsch, 2014] and DBPedia[3]) [Auer et al., 2007], ii) the selection of candidates and classification (using ELMo embeddings [Peters et al., 2018] and a K-nearest neighbours (KNN) classifier), and iii) the calculation and adjustment of a confidence level.

In the following, we present our system overview in Section 2. Section 2.3 shows some preliminary runs with the sample annotations set in order to obtain the confidence level of a mention to belong to the predicted entity type. Next, we provide detailed descriptions of the training in Section 3. Section 4 shows the official evaluation scores of our submissions. Finally, Section 5 presents our conclusions and further work.

2 System overview

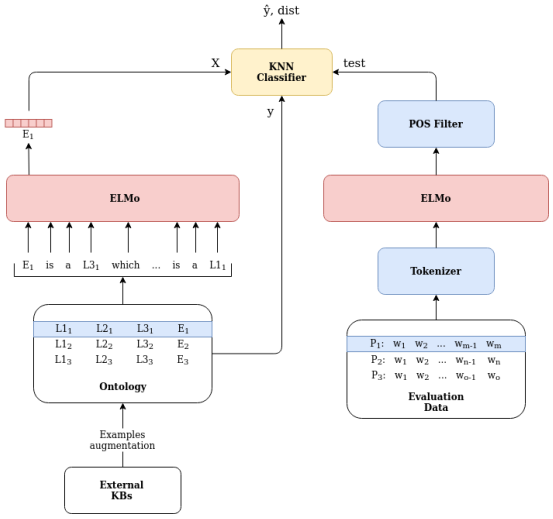

Figure 1: system architecture

Our approach can be splitted in two main parts as shown in Figure 1: in the left one we collect additional examples E of every fine-grained entity type $L1.L2.L3$ (type.subtype.subsubtype) from different external knowledge bases that will be used to build context-based sentences. Such sentences are then embedded, and the resulting vectors for the example tokens are used to train a KNN classifier that will predict entity types given a word embedding.

[1] available at https://yago-knowledge.org/
[2] available at https://www.wikidata.org/
[3] available at https://es.dbpedia.org/

In the right branch, we first tokenize each paragraph $P : \{w_1, w_2, ...w_n\}$ in the texts and then vectorize every word on it. After that, we filter out every word that is not tagged as noun, proper noun or pronoun by a part-of-speech (POS) tagger. Finally, we use the trained KNN classifier to predict entity types for each embedding from the words that have passed the filter.

We're using the same ELMo's [Peters et al., 2018] pre-trained model[4] [Che et al., 2018] from the English CoNLL17 corpus [Fares et al., 2017] for both the synthetic sentences and the paragraphs from the development texts. We are also using Spacy [Honnibal et al., 2020] for the POS tagging filtering and Gensim [Řehůřek and Sojka, 2010] for extending the number of examples with the most similar words from FastText's pre-trained word vectors [Mikolov et al., 2018].

2.1 Data augmentation with knowledge bases

Since this approach relies heavily on the ability of the example embeddings to define the category to which they belong, we have put great emphasis on increasing the number of examples present in the ontology. This way we can make an ELMo model, which is context-aware, to learn from every entity level. To do so, we have firstly searched for words similar to these examples by computing the cosine similarity measure between their weight vectors and the vectors of each key in a pre-trained FastText [Bojanowski et al., 2017] model. Secondly, for those entity types with few or no examples in the ontology, we have collected examples of them from different knowledge bases such as Yago, DBPedia or WikiData.

Each category and each knowledge base defined a different search strategy. In some cases, just using the "description" or "label" properties was enough (mostly for Yago related queries), but the complexity and variability of the ontology behind DBPedia and, to a lesser extent, Wikidata, made difficult to build a general tool for enriching the RUFES ontology in a more automatic way. Thus, we went label by label, querying the SPARQL endpoints of these three resources to find examples.

Once we have enough examples, for each one of them we have built a synthetic sentence that allows us to calculate their contextual embedding. To do this, in order to make the resulting embeddings more accurate, we have translated every category name into natural language (e.g., APP for application, FAC for facility, ConsumerGoods for consumer goods, ...). After this, we build each sentence according to the following pattern: <Example> is a <Level n> [which is a <Level n-1> [which is a <Level n-2>]], e.g., "Facebook is a social media which is a communication software which is an application".

[4]http://vectors.nlpl.eu/repository/

We then calculate ELMo embeddings [Peters et al., 2018] for each sentence and keep the one corresponding to the <Example>, that will be the one we use to train a KNN model with which we'll later classify each mention.

2.2 Selection of candidates and classification

For the evaluation data, we used each paragraph as a context to extract the ELMo [Peters et al., 2018] embedding of each word and then filtered out all those that, after a POS tagging, were not labeled as nouns, proper nouns or pronouns. Once the embeddings of all potential mentions in the texts are obtained, we label each one with the KNN classifier that we previously trained with the ontology examples.

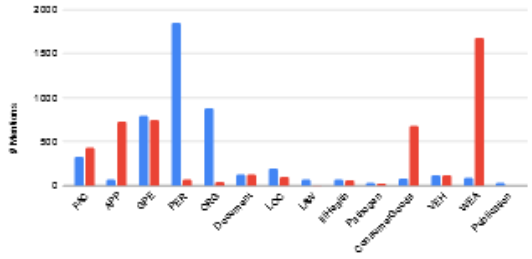

Figure 2: Mention count before boosting underrepresented entity types

PER and ORG entity types seem to be particularly relevant in journalistic texts according to the set of sample annotations, in which they represent more than half of the annotated mentions: 39% for the former and 19% for the latter. For this reason, and since our system had particular difficulty in detecting such categories as shown in the figure 2, we have used Spacy's [Honnibal et al., 2020] entity recognizer so that if it recognizes any of them in addition to other hard to find entities such as LAW or LOC, the corresponding mention will be tagged with a different KNN classifier trained with examples from just that same category.

This has proved to be helpful for classifying types of entities that our system struggled to predict previously as seen in figure 3. PER mentions have been raised from 76 to 406 and ORG ones went from 42 to 456 for the sample annotations set.

On the other hand, WEA, APP and ConsumerGoods entity types were being overpredicted by our system. In order to solve this, we increased the distance resulting from the KNN classifier whenever it predicted one of these categories. Then, we discarded mentions whose predicted distance was greater than an established threshold. Sample annotations prediction results are shown in figure 4.

Finally, we performed a co-reference search so that each word referring to a previous one inherits both its entity ID and type.

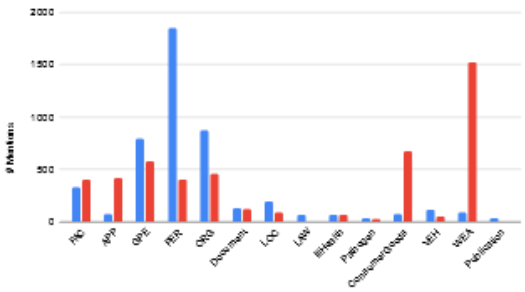

Figure 3: Mention count after boosting underrepresented entity types

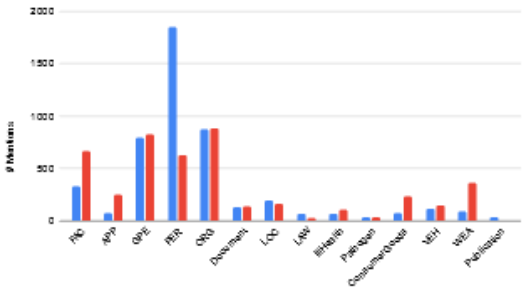

Figure 4: Mention count after both boosting underrepresented and penalizing overrepresented entity types

2.3 Computing a confidence level

In order to calculate the confidence level we first calculate the distance between each mention embedding and its nearest word vector in the ontology examples. This generates a vector of thousands of distances. Next, we scale each distance down to a range [0, 1] by dividing by the maximum distance found. The final confidence value for each mention will be 1 - the scaled distance, which could be considered has the probability of the example (thus, its represented class) to be associated with the candidate term. Equation 1 illustrates how this confidence value is computed.

$$p(t) = 1 - \frac{min_i d(t, e_i)}{max_i d(t, e_i)} \quad (1)$$

Where $p(t)$ is the probability of the candidate term t and $d(t, e_i)$ is the Euclidean distance of candidate term t to example e_i.

3 Training and adjustment

As we already mentioned, we're using 5 different KNN models: 4 of them are trained with just one of the entities our system wasn't good at detecting (i.e. PER, ORG, LAW and LOC) and the last, which is a general one, is trained with all the example embeddings from the ontology. In this approach we perform a NER analysis in such a way that, in case it detects one of those 4 categories for a mention, we'll predict all its three levels with the corresponding KNN model.

threshold	fscore
0.8	0.0362
0.9	0.0434
1.0	**0.0446**
1.1	0.0394
1.2	0.0349

Table 1: Performance of different thresholds on the sample annotations set

precis	recall	fscore	measure
0.479	0.482	0.481	strong mention match
0.142	0.143	0.142	strong typed mention match
0.356	0.358	0.357	mention ceaf
0.125	0.125	0.125	typed mention ceaf
0.263	0.407	0.319	entity ceaf

Table 2: Complete first results

As described in Section 2, we have also established a threshold to discard all mentions that passed the POS filtering but weren't close enough to any example embedding in the ontology. We tested different values: the average distance (i.e. removing the half worst mentions), and some values around it (80%, 90%, 110% and 120%), where the one performing the best was the first as seen in Table 1.

4 Results and discussion

RUFES evaluation was divided into two different phases. Both are evaluated in the same way, but the second is after a manual feedback over a subset of documents, so systems can re-adjust their parameters. Our results for the first phase are shown in Table 2. Our results are quite low, and we clearly fail in the type identification. Our results after attempting to tweak the system with feedback information are even worse (see Table 3). A detailed revisions and an error analysis have to undergo in order to fully understand the weaknesses of our approach.

If we compare these results with the average F-score of the rest of participants, which is 0.805 for strong mention match, we must admit that our system is not a valid approach to fine-grain entity recognition. Regarding type-based scores, we get best results on

precis	recall	fscore	measure
0.388	0.477	0.428	strong mention match
0.174	0.214	0.192	strong typed mention match
0.238	0.292	0.262	mention ceaf
0.133	0.164	0.147	typed mention ceaf
0.200	0.263	0.227	entity ceaf

Table 3: Complete feedback results

Mention type	fscore
NAM	0.416
NAM-NOM	0.486
NAM-PRO	0.415
NOM	0.501
NOM-PRO	0.488
PRO	0.408

Table 4: F-score values by mention type on first submission

NOM and NOM-PRO types, with values of 0.501 and 0.488 respectively for F-score in first evaluation phase, as detailed in Table 4.

We realized that the idea of generating vectors for proper names with ELMo was naïve, as most of those names were unknown by the model. A more varied definition of templates or the extraction of excerpts where the examples found appear in news could improve the quality of the final embeddings vector for that example.

5 Conclusions and future work

In this paper we have presented the design and implementation of SINAI TAC RUFES 2020 system. We explored a non-supervised method relying on data augmentation by samples retrieval from knowledge bases. RUFES ontolofy was enriched by populating it with many samples for each entity type by querying know open knowledge bases like Wikidata, Yago and DBpedia. This examples were used to "attract" candidates terms to their represented entity types.

Our results and their analysis show us that our approach, although far from the median scores, could be worth exploring, but that it needs a major revision in key aspects like ELMo vectors computation (improving samples contexts) and, maybe, a learned threshold for each possible label.

Additional analysis of the errors has been initiated now that ground-truth values have been released. We hope that our approach, maybe with more advanced models, like those in the BERT family, based ones [Devlin et al., 2018] could lead to better performance.

Acknowledgments

This study is partially funded by the Spanish Government under the LIVING-LANG project (RTI2018-094653-B-C21) and the Interreg V-A España-Portugal project (0551_PSL_6_E) from the Fundación CSIC.

References

[Auer et al., 2007] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

[Bojanowski et al., 2017] Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

[Che et al., 2018] Che, W., Liu, Y., Wang, Y., Zheng, B., and Liu, T. (2018). Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64, Brussels, Belgium. Association for Computational Linguistics.

[Devlin et al., 2018] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[Fares et al., 2017] Fares, M., Kutuzov, A., Oepen, S., and Velldal, E. (2017). Word vectors, reuse, and replicability: Towards a community repository of large-text resources. In *Proceedings of the 21st Nordic Conference on Computational Linguistics*, pages 271–276, Gothenburg, Sweden. Association for Computational Linguistics.

[Getman et al., 2018] Getman, J., Ellis, J., Strassel, S., Song, Z., and Tracey, J. (2018). Laying the groundwork for knowledge base population: Nine years of linguistic resources for tac kbp. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

[Honnibal et al., 2020] Honnibal, M., Montani, I., Van Landeghem, S., and Boyd, A. (2020). spaCy: Industrial-strength Natural Language Processing in Python.

[Mikolov et al., 2018] Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., and Joulin, A. (2018). Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

[Peters et al., 2018] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations.

[Řehůřek and Sojka, 2010] Řehůřek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. `http://is.muni.cz/publication/884893/en`.

[Suchanek et al., 2008] Suchanek, F. M., Kasneci, G., and Weikum, G. (2008). Yago: A large ontology from wikipedia and wordnet. *Journal of Web Semantics*, 6(3):203–217.

[Vrandečić and Krötzsch, 2014] Vrandečić, D. and Krötzsch, M. (2014). Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.