Basic CRF approach to DIANN 2018 shared task

Pol Alvarez Vecino and Lluís Padró

TALP Research Center
Universitat Politècnica de Catalunya
pol.avms@gmail.com, padro@cs.upc.edu

Abstract. This paper describes the UPC_2 system participation in DIANN (*Disability annotation on documents from the biomedical domain*) shared task, framed in the IBEREVAL 2018 evaluation workshop[1]. The system tackles the detection of disabilities using a CRF to perform IOB Named Entity Recognition (NER). Regarding the detection of negated disabilities, the out-of-the-box NegEx rule-based system is used.

Keywords: Medical Named Entity Recognition · CRF · Disabilities · Negation detection

1 Introduction

This paper presents a simple approach to the Disability detection shared task DIANN proposed in the framework of IBEREVAL 2018 [2].

The task consists of identifying disabilities in biomedical research articles. The documents are abstracts or short descriptions, typically a few hundred words long, and use standard grammar and orthographic conventions. The goal is to detect where a disability is described or attributed to a patient. Thus, disability mentions that are negated or discarded in the text should be marked as "negated". The task requires the participation on Spanish, and English is optional.

We approach the task in two sequential stages: Disability recognition and negation detection. The former is addressed with a classical NER approach: A CRF [3] performing IOB annotation. The later is solved using an out-of-the-box rule-based system: NegEx[1], which has been adapted for Spanish.

2 Disability recognition approach

The training data format is short files with XML tags indicating disabilities and negation expressions. Our approach was to transform the input into a list of words and add to them the part of speech (PoS) and IOB information. The result elements were tuples of $(word, POS, IOB - tag)$. Only the disabilities

[1] http://nlp.uned.es/diann

were considered when building the IOB information. The negation expressions were not used because they were predicted by another module which does not require IOB annotations.

The PoS tagging was done using NLTK [4]. The IOB-tagging was performed manually using the entities inside the `<dis> ... </dis>` XML tags.

2.1 Basic Model

The basic model is a Conditional Random Field (CRF) applied to the IOB-tagged dataset. The implementation was built using NLTK's basic CRFTagger which is a module used for POS tagging that uses CRFSuite[2].

The model uses a *predefined entities list* extracted from the training set which contains an entity per line. It also uses an *acronyms list* which is built filtering all the single-word entities with all letters in uppercase.

2.2 Features

The features used are grouped by similarity in order to ease the evaluation of their utility. The following list describes them:

1. ***word*, *pos*, *lemma*, *all-caps*, *strange-cap*, *contains-dash*, *contains-dot***
   current word, its lemma, POS, whether all the word's letter are uppercase, whether the word contains uppercase letters while the first is lowercase, and if it contains a dash or dot.
2. ***inside-entities*, *is-acronym***
   boolean indicating whether the word is found in the predefined entities list, and in the acronyms list.
3. ***position-X*, *total-position-X***
   position-X is true if the word is found at the position X inside an entity of the predefined entities list; *total-position-X* is the number of entities in the list in which the word appears.
4. ***prev-X-word*, *prev-X-pos*, *prev-X-lemma***
   the word appearing X positions to the left of the current word, its POS, and its lemma.
5. ***next-X-word*, *next-X-pos*, *next-X-inside-ente***
   the word appearing X positions to the right of the current word, its POS, and if it is inside the entities list.
6. ***next1-word*, *next1-pos***
   concatenation of the current and next words, and their part of speech.
7. ***prev1-word*, *prev1-pos*, *prev1-lemma***
   concatenation of the previous and current words, their part of speech, and their lemma.
8. ***next2-word*, *next2-pos***
   concatenation of the two next words, and their part of speech.

[2] `https://pypi.python.org/pypi/python-crfsuite`

9. *prev2-word, prev2-pos*
concatenation of the two previous words, and the concatenation of their part of speech.

The number of words used in the features *next-X-* and *prev-X-* are tunable parameters. In the final execution, the number of preceding words considered was three and, for the next words, the number was two.

2.3 Training

The final features were chosen using 10-fold cross-validation. For each fold, the model builds the entities and acronyms list (using only the nine training chunks) and trains the model to predict the remaining validation chunk. After averaging all the folds F1-score results, the features corresponding to the best average (for the two languages) were used to train a model using all the training dataset.

For each group of features described in the previous section, the whole group was deactivated to check if they affected the precision.

Initially, the groups 4-9 contained all the features of groups 1-3 applied to their elements (i.e. the features applied to the *prev-X-* word, or concatenating them in the case of *next1-feature*. Experiments were performed deactivating one group at a time and checking the impact on performance. This allowed us to remove non-useful feature groups, leaving only those groups with actual contribution to the task. Once the useful feature groups were chosen, a more fine-grained inspection was carried to remove useless features inside each group, resulting in the final feature groups reported above.

3 Negation detection approach

The negations were predicted using an out-of-the-box NegEx implementation[3]. After tagging the entities, each sentence and the entity it contains are passed to NegEx which marks if the entity is negated and which is the set of words negating it (if no entity is present the sentence is not fed to NegEx). We detected that almost all the correct negations were close to the entity so the negation expressions that were more than three words away of the entity were discarded.

4 Experiments and Results

The experiments performed were to predict the whole training dataset using ten-fold cross-validation. The best model was then used to annotate the test set. Table 1 shows the results of some experiments varying the used feature set. Using all the features gives the best results. All results are computed using the evaluation tool provided by DIANN organizers[4].

[3] https://github.com/mongoose54/negex
[4] https://github.com/diannibereval2018/evaluation

Table 2 reports the final scores obtained on the official test set. The fields evaluated here are: *disability*, refers to all disabilities annotation both included or not in a negation; *negated disability*, considers all the negation-related annotations (disability, negation trigger, and scope); and *non-negated disability + negated disability* which evaluates jointly the annotation of disabilities and negation (negated disability are considered correct if both negation and disability are correct). For all categories, both partial and exact results are provided.

	Spanish			English		
	Precision	Recall	F1 score	Precision	Recall	F1 score
Group disabled: 2						
Negation	0.50	0.55	0.52	0.46	0.35	0.40
Disability	0.72	0.63	**0.68**	0.72	0.58	0.64
Group disabled: 3						
Negation	0.50	0.50	0.50	**0.48**	0.35	0.41
Disability	0.73	0.51	0.60	**0.75**	0.56	0.64
Group disabled: 4						
Negation	0.51	**0.58**	**0.54**	**0.48**	0.40	**0.44**
Disability	**0.74**	0.59	0.65	0.74	0.65	0.70
Group disabled: 5						
Negation	0.51	0.55	0.53	**0.48**	0.38	0.42
Disability	0.71	0.59	0.64	**0.75**	0.65	0.69
Group disabled: 6						
Negation	0.49	0.53	0.51	**0.48**	0.38	0.42
Disability	0.72	0.59	0.65	0.73	0.63	0.68
Group disabled: 7						
Negation	0.50	0.55	0.52	0.47	0.35	0.40
Disability	0.72	0.60	0.65	0.73	0.64	0.68
Group disabled: 8						
Negation	0.49	0.53	0.51	0.47	0.35	0.40
Disability	0.72	**0.69**	0.65	**0.75**	0.65	0.70
Group disabled: 9						
Negation	0.47	0.50	0.48	**0.48**	0.38	0.42
Disability	0.71	0.59	0.64	0.74	0.65	0.69
Group disabled: None						
Negation	**0.52**	0.55	0.53	0.47	**0.41**	0.43
Disability	**0.74**	0.62	**0.68**	**0.75**	**0.67**	**0.71**

Table 1. Results of cross-validation experiments deactivating one feature group at a time

5 Conclusions

We have presented a simple CRF approach to disability detection in medical texts. The systems produces average results, ranking in the middle of the table

	Exact Match			Partial Match		
	Precision	Recall	F1 score	Precision	Recall	F1 score
English						
Disability	0.756	0.560	0.643	0.822	0.588	0.686
Negated Disability	0.647	0.478	0.550	0.941	0.696	0.800
Non-negated + Negated Disability	0.724	0.519	0.604	0.822	0.588	0.686
Spanish						
Disability	0.732	0.502	0.596	0.828	0.568	0.674
Negated Disability	0.737	0.636	0.683	0.895	0.773	0.829
Non-negated + Negated Disability	0.710	0.480	0.573	0.819	0.555	0.661

Table 2. Final testing results with the full-featured model.

for most metrics. We consider that the presented approach has improvement margin, since the used features are a basic set, and could be extended with more advanced semantic information such as word embeddings.

Acknowledgements

This research has been partially funded by Spanish Government through GraphMed project TIN2016-77820-C3-3-R.

References

1. Chapman, W., Bridewell, W., Hanbury, P., F. Cooper, G., Buchanan, B.: A simple algorithm for identifying negated findings and diseases in discharge summaries **34**, 301–310 (11 2001)
2. Fabregat, H., Martinez-Romo, J., Araujo, L.: Overview of the diann task: Disability annotation at ibereval 2018. In: Proceedings of the Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018) (2018)
3. Lafferty, J., McCallum, A., Pereira, F.C.N.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. ICML '01 Proceedings of the Eighteenth International Conference on Machine Learning **8**(June), 282–289 (jun 2001). https://doi.org/10.1038/nprot.2006.61
4. Loper, E., Bird, S.: Nltk: The natural language toolkit. In: Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1. pp. 63–70. ETMTNLP '02, Association for Computational Linguistics, Stroudsburg, PA, USA (2002). https://doi.org/10.3115/1118108.1118117, https://doi.org/10.3115/1118108.1118117