Semi-supervised learning for disabilities detection on English and Spanish biomedical text

Salvador Medina[1,2], Jordi Turmo[1,2], Henry Loharja[2], and Lluís Padró[1,2]

[1] Talp Research Center
[2] Universitat Politcnica de Catalunya
http://www.talp.upc.edu/

Abstract. This paper describes the disability detection model approaches presented by UPC's *TALP_3* team for the DIANN 2018 shared task. The best of those approaches was ranked in 3rd place for exact-matching of disability detection. The models combine a semi-supervised learning model using CRFs and LSTM with word embedding features with a supervised CRF model for the detection of disabilities and negations respectively.

Keywords: disabilities detection · biomedical abstracts · semi-supervised learning.

1 Introduction

This paper describes the approaches built by one of the UPC teams (TALP_3) to participate in the DIANN 2018 task [4]. The task consisted in automatically recognizing disabilities occurring biomedical domain. Two different subtasks were proposed: dealing with Spanish text and dealing with English text, both being abstracts from biomedical journals.

The paper is organized as follows. Sections 2 and 3 describe the approaches used to learn the disabilities detection model and the negation detection model respectively. The results achieved by our methods are presented and briefly analyzed in Section 4. Finally, Section 5 concludes.

2 Learning of the disabilities recognition model

Our system tackles the disability recognition task as a sequence tagging problem, mapping each word in them to their corresponding BIO-Tag. We apply two alternative sequence tagging models: either learning Conditional Random Field (CRF) probabilistic graphical models or recurrent artificial neural networks using Bidirectional Long Short-Time Memory Network (BiLSTM) memory units and a final CRF layer.

Due to the relatively small size of the provided training corpus, the two proposed models are prone to severe over-fitting issues in completely supervised learning scenario. In order to prevent this issue and add new patterns not included in the original training set, we applied self-learning to unlabeled abstracts.

2.1 Semi-supervised learning method

Our system iteratively uses self-learning to add new examples to the training set from an unlabeled corpus. The unlabeled corpus is built by scrapping articles' abstracts from *ScienceDirect*[3] and *Tesis Doctorals en Xarxa*[4], two websites that contain PhD theses and articles from science journals. We use the disabilities' phrases found in the training set as search terms and limit to 2000 results, removing duplicates. With this, we could retrieve 41049 abstracts for English and 38632 for Spanish. We then divide it into 7 batches of 5000 abstracts each, which are applied to the respective iteration of the self-learning algorithm. Our particular implementation takes the training set, a minimum confidence threshold and the batches as input and proceeds as described in Algorithm 1.

Algorithm 1 Pseudo-code of the implemented self-learning algorithm. C_{min} is the confidence threshold, It_{max} is the maximum iteration and B_i represents the *i-th* unlabeled batch.

```
$i \leftarrow 0$
$m \leftarrow fit\_model(X_{train}, Y_{train}, 0)$
$evaluations \leftarrow \emptyset$
while $i < It_{max} \wedge \neg converges(evaluations)$ do
    $Y_{b_i} \leftarrow run(m, B_i)$
    $X_{selected}, Y_{selected} \leftarrow select\_examples(B_i, Y_{b_i}, k, C_{min})$
    $X_{train} \leftarrow X_{train} \cup X_{selected}$
    $Y_{train} \leftarrow Y_{train} \cup Y_{selected}$
    $m \leftarrow fit\_model(X_{train}, Y_{train}, i)$
    $evaluations \leftarrow evaluations \cup evaluate(m, X_{validation}, Y_{validation})$
    $i \leftarrow i + 1$
end while
return m
```

2.2 Word-Embedding models

Word-Embedding models are used in both the CRF and the BiLSTM-CRF models. For the first model, we group all words into 1024 clusters using *k-means* and apply them as binary input features. For the second, the full feature vector is fed to the input layer. We considered the *word embedding* models listed below.

- G-en: General-purpose English word-embedding model of 300 dimensions, trained using *GoogleNews*'s articles[5].

[3] https://www.sciencedirect.com/

[4] https://www.tesisenred.net/

[5] Downloaded from https://code.google.com/archive/p/word2vec/

- G-es: General-purpose Spanish word-embedding model of 300 dimensions, trained from multiple sources[6].
- S-en and S-es: English and Spanish context-specific word-embedding models of 30 dimensions, trained from the unsupervised corpus fetched from *ScienceDirect* using Pennington's *GloVe* algorithm [8].

2.3 Conditional Random Field tagger

Our first sequence tagger consists on a linear-chain CRF tagger with binary input features, using the implementation provided by *Python CRFSuite*[7]. In order to compute the confidence of tagged sequences, required for self-learning, we compute the probability $P(y|x;w)$ of the assigned labels y respect to the input x and feature functions w as defined in Equation 1.

$$P(y|x;w) = \frac{\exp(\sum_i \sum_j w_j f_j(y_{i-1}, y_i, x, i))}{\sum_{y' \in Y} \exp(\sum_i \sum_j w_j f_j(y'_{i-1}, y'_i, x, i))} \tag{1}$$

Input Features For each token of the input sentences, we use a combination of the features listed below, in a window of up to 7 tokens (3 before and 3 after). If the window is within the beginning or the end of the document, the special features *Begin of Sentence (BOS)* and *End of Sentence (EOS)* are applied. Sentences are tokenized and analyzed using *FreeLing*[7], a multi-lingual natural language processing tool.

- **Word capitalization**, either *all lowercase*, *all uppercase*, *first uppercase* or *combined.*
- Whether or not the token contains **numerical characters**.
- **Prefixes and suffixes** of length 3 and 4, padded when necessary.
- **Part of Speech**, determined by *FreeLing*'s PoS tagger.
- **Lemma**, determined by *FreeLing*'s lemmatizer.
- **Word embedding** cluster.
- **Token**, just used in the first iteration of the self-learning algorithm.

2.4 Bilinear Long Short-Time Memory model

The BiLSTM-CRF model is implemented using *Python's Keras* library with *TensorFlow* backend[8]. LSTM layers for both directions use the standard LSTM layer provided by *Keras*, whereas for the output CRF layer we use the implementation in the *Keras-Contrib* extension library[9]. A dropout factor of 0.5 is applied to the output layer for regularization.

[6] Downloaded from `http://crscardellino.me/SBWCE/`[2]

[7] Python CRFSuite - Python bindings to CRFSuite [6]

[8] Keras: The Python Deep Learning library[3]

[9] keras-contrib : Keras community contributions - GitHub

We tune the network to estimate the probability of each tag assignment by using the *Adam* optimizer with categorical cross-entropy loss function. The probability of the output sequence $P(Y^T)$ is usually computed as the product of conditional probabilities at each time step $P(y_t|Y^{t-1})$. However, this is not practical in our case, as the probability vanishes fast for long sentences, which would potentially prioritize shorter ones. To prevent this, we opted for defining the confidence as the minimum output probability for all time-steps $min\{P(y_t|Y^{t-1})\}$.

Input Features In this second model we only consider word-embeddings as the input features. For out-of-dictionary words, the average vector of all words in the training corpus is applied. When both the general-purpose word embedding model and the context-specific word embedding model are used, both feature vectors are concatenated.

3 Learning of the negation detection model

The approach we use for the negation detection is based on the work presented by Agarwal and Yu [1]: a CRF-based negation detection. That work uses a tool named ABNER [9] which is a software tool for molecular biology text analysis especially in named entity recognition. At ABNER's core is a statistical machine learning system using linear-chain CRFs with a variety of orthographic and contextual features. The tool includes a Java API allowing users to incorporate ABNER into their systems, as well as training and using models for other data. This Java API are what we mainly used in our approach for negation detection by adapting a point of view of named entity recognition.

3.1 Data Preprocessing

The main task in DIANN is detecting disabilities whilst detecting negation is only focused on those that are related to the negated disabilities. This characteristic of the task results in a very few number of negation occurrences being annotated inside the training data and so it is insufficient to use only this dataset for training a negation detection model using the approach we use. As a way to tackle this issue we decided to include another datasets, Bioscope [10] for English and IULA corpus [5] for Spanish, to enrich the training dataset we have for DIANN task especially for negation. By using this method, we obtain two datasets for training:

1. English training dataset: sentences with negation annotated in both English training data from DIANN and abstracts from Bioscope.
2. Spanish training dataset: sentences with negation annotated in both Spanish training data from DIANN and from IULA corpus.

We pre-processed both corpora by enriching the raw data with BIO tags in order to be used as input for training. Each token results tagged with "—B-S" if it is in the beginning of negation scope, ”—I-S” if it is inside the negation scope, or ”—O” if it is outside of the scope. In order to capture the information of the negation cue, we append suffix ”C” to the tag if the token is part of a negation cue. An example of sentence in this format is:

```
Five|O hundred|O twenty-five|O infants|O without|B-SC risk|I-S
                          factors|I-S
```

3.2 Training The Model for Negation Detection

Our goal with negation detection is to classify whether each word inside a sentence is part of negation (scope or cue) or not. Using this understanding, we can use ABNER, a CRF-based NER tool, as a platform for negation detection by adapting it to reach that goal. We give three kind of classification for each word which we observe: Scope, Cue, or Out. A word classified as Out is not part of either a negation scope or a cue. Figure 3.2 shows the flow of our negation detection approach.

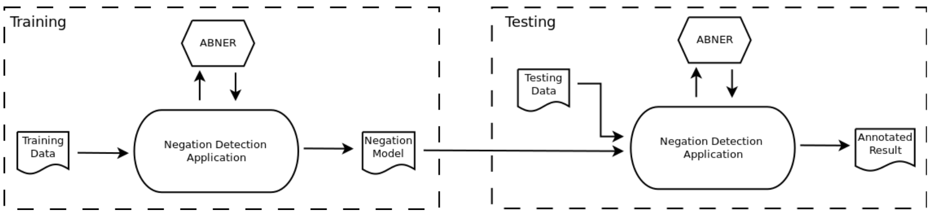

Fig. 1. Flow that describes negation detection approach.

We used the CRF-based system in ABNER to train a negation detection model by using the training data we prepared before, as described in Section 3.1. The training process uses an orthographic feature set which by default is the one used in ABNER. The simplest and most clear feature set is the vocabulary from the training data. Generalizations over how the words are written (capitalization, affixes, etc.) are also relevant. The current approach includes training vocabulary, 17 orthographic features based on regular expressions (e.g., Alphanumeric, HasDash, HasDigit) as well as prefixes and suffixes in the character length ranged from three to four. As an example, the word ’without’ has two prefix features: Prefix3=’wit’ and Prefix4=’with’ as well as two suffix features: suffix3=’out’ and suffix4=’hout’. To model localization context in a simple way, neighboring words in the window [-1,1] are also added as features. For example, the middle token in the sequence with no symptoms has features Word=’no’, Neighbor=’with’, and Neighbor= ’symptoms’. Words are also assigned with a generalized word class in which capital letters are replaced by ’A’, lowercase

Table 1. Results for the Spanish subtask achieved by the UPC_3 team

	Approach	Exact			Partial		
Spanish disability		P	R	F1	P	R	F1
	CRF1	0,814	0,594	0,687	0,898	0,655	0,758
	CRF2	0,807	0,603	0,69	0,889	0,664	0,76
	LSTM	0,67	0,603	0,634	0,743	0,668	0,703
Spanish neg_disability							
	CRF1	0,647	0,5	0,564	0,941	0,727	0,821
	CRF2	0,647	0,5	0,564	0,941	0,727	0,821
	LSTM	0,688	0,5	0,579	1	0,727	0,842
Spanish all							
	CRF1	0,779	0,555	0,648	0,89	0,633	0,74
	CRF2	0,772	0,563	0,652	0,88	0,642	0,742
	LSTM	0,64	0,559	0,597	0,735	0,642	0,685
	IXA	0,746	0,795	0,77	0,82	0,873	0,846

Table 2. Results for the English subtask achieved by the UPC_3 team

	Approach	Exact			Partial		
English disability		P	R	F1	P	R	F1
	CRF1	0,799	0,605	0,689	0,875	0,663	0,754
	CRF2	0,795	0,605	0,687	0,865	0,658	0,748
	LSTM	0,655	0,617	0,636	0,742	0,7	0,72
English neg_disability							
	CRF1	0,773	0,739	0,756	0,955	0,913	0,933
	CRF2	0,773	0,739	0,756	0,955	0,913	0,933
	LSTM	0,696	0,696	0,696	0,913	0,913	0,913
English all							
	CRF1	0.772	0.584	0.665	0.87	0.658	0.749
	CRF2	0.768	0.584	0.664	0.859	0.654	0.743
	LSTM	0.626	0.593	0.609	0.735	0.695	0.715
	IXA	0.746	0.811	0.777	0.841	0.914	0.876

letters by ’a’, digits by ’0’, and all other characters by ”. There is a similar ”brief word class” feature which collapses consecutive identical character types into one. For example, the words ”EX3” and ”SHA1” are given the features WC=AA0 and BWC=AAA0, respectively, while ”N-folds” and ”T-cells” both are assigned WC=A_aaaaa and BWC=A_a.

After applying the resulting negation detection model to the test, we do some post-processing to change the BIO format of the result into the required format.

4 Results

We performed three executions for each language (Spanish and English). Each execution combines the semi-supervised method described in 2.1, one tagger model from those presented in Sections 2.3 and 2.4 (CRF and BiLSTM) and one or two of the word embeddings described in Section 2.2 (G-es, S-es, G-en, S-en). Concretely, the approaches were:

- CRF1: combination of CRF and specific word embeddings (S-es or S-en).
- CRF2: CRF combined with both specific and general word embeddings (S-es+G-es or S-en+G-en).
- BiLSTM: BiLSTM combined with the specific and general word embeddings.

Tables 1 and 2 report the results achieved by each of our three approaches for Spanish and English subtasks in terms of Precision, Recall and F1-score. Each table provides the results for the detection of all the disabilities, without considering scopes and cues (*[LANG] disability*), for the negated disabilities, considering the tuple [scope, cue, disability] (*[LANG] neg_disability*) and for both together (*[LANG] all*). With the aim of providing a better comparison, the results from the best participant in DIANN (IXA group) is included in *[LANG] all*.

As far as *[LANG] all* and *[LANG] disability* results are concerned, our best approach was CRF2 for Spanish and CRF1 for English, although for the later, the differences with CRF2 does not seem statistically significant. Hence, the use of both the specific and the general word embeddings seems like a better choice for detecting disabilities and consequently for the full task of detecting disabilities together with their scopes and cues when negated.

On the one hand, however, our results for the detection of disabilities are around 10 points lower than those achieved by the best approach presented in DIANN for Spanish, and from 13 (Exact) to 14 (Partial) for English. This leads to a similar behaviour of our approaches for the whole DIANN task. The hypothesis for these differences is that the resulting taggers are biased to get better precision than recall, hence penalizing the overall F_1 score.

On the other hand, F_1 scores in the whole DIANN task shows interesting results. For Spanish, CRF2 was ranked the 3th for both exact and partial matching, although the differences with the 2nd place are not statistically significant. For English, CRF1 was also ranked the 3th for exact matching, but it fell to the 6th place for partial matching.

Regarding negated disabilities, the results are more difficult to analyze. Mainly because the number of negated disabilities is around 30-40, so for the test corpus as for the train corpus, which is not significant enough to conclude with a statistically representative comparison.

5 Conclusions

In this paper we have described our participation in DIANN 2018 task of disabilities detection for Spanish and English biomedical text. Our best approach to find exact matches consisted of a semi-supervised approach combining CRF, medical domain-specific word embeddings and context-independent word embeddings for both Spanish and English. This approach was ranked in 3th position in the official results, although far from the 1st ranked: from 11 to 13 points less in F_1 score.

Acknowledgments

This contribution has been partially funded by the Spanish Ministry of Economy (MINECO) and the European Union (TIN2016-77820-C3-3-R and AEI/FEDER,UE).

References

1. Agarwal, S., Yu, H.: Biomedical negation scope detection with conditional random fields **17**, 696–701 (11 2010)
2. Cardellino, C.: Spanish Billion Words Corpus and Embeddings (March 2016), `http://crscardellino.me/SBWCE/`
3. Chollet, F., et al.: Keras. `https://keras.io` (2015)
4. Fabregat, H., Martínez-Romo, J., L., A.: Overview of the diann task: Disability annotation task at ibereval 2018. In: Proceedings of the Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval'18) (2018)
5. Marimon, M., Vivaldi, J., Bel, N.: Annotation of negation in the iula spanish clinical record corpus. In: Proceedings of the Workshop Computational Semantics Beyond Events and Roles. pp. 43–52 (2017)
6. Okazaki, N.: Crfsuite: a fast implementation of conditional random fields (crfs) (2007), `http://www.chokkan.org/software/crfsuite/`
7. Padr, L., Stanilovsky, E.: Freeling 3.0: Towards wider multilinguality. In: Proceedings of the Language Resources and Evaluation Conference (LREC 2012). ELRA, Istanbul, Turkey (May 2012)
8. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP). pp. 1532–1543 (2014), `http://www.aclweb.org/anthology/D14-1162`
9. Settles, B.: Abner: an open source tool for automatically tagging genes, proteins and other entity names in text. Bioinformatics **21**(14), 3191–3192 (2005)
10. Vincze, V., Szarvas, G., Farkas, R., Móra, G., Csirik, J.: The bioscope corpus: biomedical texts annotated for uncertainty, negation and their scopes. BMC bioinformatics **9**(11), S9 (2008)