UH-MMM at eHealth-KD Challenge 2021

Loraine Monteagudo-García, Amanda Marrero-Santos, Manuel Santiago Fernández-Arias, and Hian Cañizares-Díaz

Faculty of Math and Computer Science, University of Habana, La Habana, Cuba

Abstract. This paper explains the solution presented by the UH-MMM group to the eHealth-KD challenge at IberLEF 2021. Two main subtasks for knowledge discovery were defined: entity recognition and relationship extraction. The evaluation of the task is divided into three scenarios: one corresponding to the detection of entities, one corresponding to the detection of relations between such pair of entities, and the third one corresponding to the extraction of both entities and relationships. For both subtasks, our proposal makes use of BiLSTM as contextual encoders and Dense layers as the tag decoder architecture of the model. In the challenge, the system ranked fifth in the main scenario, fourth in the scenario evaluating the first task, and fifth in the last scenario. The score obtained in the relationship extraction task shows that the proposed approach needs to be further explored.

Keywords: eHealth · Knowledge Discovery · Natural Language Processing · Machine Learning · Deep Learning · Named Entity Recognition · Relation Extraction

1 Introduction

This paper explains the solution presented by the UH-MMM team in the eHealth-KD challenge at IberLEF 2021 [4]. The challenge proposes modeling of the human language in which electronic health documents could be machine-readable from a semantic point of view. It is divided into two tasks: A for entity recognition and B for the extraction of the semantic relationships between pairs of such entities. The evaluation was also divided into 3 scenarios: key-phrase identification and classification to evaluate task A, relation extraction to evaluate task B, and full knowledge extraction to evaluate both tasks. eHealthKD 2021's edition includes one significant addition concerning previous editions: a small selection of sentences from different domains and languages (i.e., English) to encourage cross-domain and multi-lingual approaches.

Our solution for both tasks is based on Recurrent Neural Networks (RNN) or, more precisely, Bidirectional Long Short Term Memory (BiLSTM) as contextual encoders and Dense layers as the tag decoder architecture of the model. This

IberLEF 2021, September 2021, Málaga, Spain.

architecture is chosen because of the sequential structure of the input its widely used in the literature for addressing the Named Entity Recognition (NER) problem. The system makes use of POS-tag (Part-of-Speech tag) information, dependency relations, char-level representations as well as contextual embeddings. The Relation Extraction (RE) task is addressed in a pairwise-query fashion, encoding the information about the sentence and the given pair of entities using syntactic structures derived from the dependency parse tree. In addition, a special type of relation was used to encode the relationship between non-related pairs of entities.

The rest of the paper is organized as follows. Section 2 explains in detail the proposed model. The results of the model in the several scenarios evaluated during the eHealth-KD 2021 event are presented in Section 3. In section 4 some insights derived from the performance of each one of our runs were discussed. Finally, the conclusions and some future work recommendations are shown in Section 5.

2 System Description

The proposed solution solves both tasks separately and sequentially. Thus, independent models with different architectures and features were trained to solve the NER and RE problems. The main distinction between the two architectures raises from the type of problem they solve. The first task is posed as a tag prediction problem that takes the raw text of a sentence as input and outputs two independent tag sequences: one in the BILOUV tag system for entity prediction and another with the tags corresponding to each entity type (*Concept*, *Action*, *Reference*, *Predicate*). The BILOUV tag scheme classification corresponds to **B**egin, for the start of an entity; **I**nner, for the token in the middle; **L**ast for the ending token; **U**nit, to represent single token entities; **O**ther to represent tokens that do not belong to any entity, and the o**V**erlapping tag is used to deal with tokens that belong to multiple entities. On the other hand, the second task is addressed as a series of pairwise queries among the entities present in the target sentence, oriented towards identifying the relevant relations between the previously extracted entities.

Taking into account the multilingual characteristics of the task, the feature extraction process of the syntactic features is handled in two phases. In the first one, the input sentence is classified by its language using a FastText pre-trained model for language identification [3][2]. Afterward, in the second phase, two different models of Spacy (https://spacy.io/) were used depending on the sentence's language (`es_core_news_sm` for Spanish and `en_core_web_sm` for English). These models were used to extract features like the POS tag, the dependency parse tree, and the dependency tag.

2.1 The Entity Recognition Model

The Entity Recognition Model task is to identify and classify key phrases in biomedical texts. Key phrases are considered to be all entities (single word or multi-word) that represent semantically relevant elements in a sentence.

Four potential classes are corresponding to each entity type:

- *Concept*: is any element of the sentence that has a semantic meaning of its own.
- *Action*: represents a concept that describes a transformation or modification of the state of one or more concepts present in the sentence.
- *Predicate*: represents a concept that describes the subset of elements of a domain that meets a certain condition.
- *Reference*: they allude to concepts that exist (in the corpus) but are not defined in the context (in the sentence).

The NER model receives as input the sentence as a sequence of words. For each word, the features described in the next subsection are extracted and vectorized. The output of the model consists of two independent tag sequences: the BILOUV tag system for entity prediction and another with the tags corresponding to entity types for classification purposes.

Input handling. Given the input sentence as raw text, some preprocessing is done to obtain a useful structure. Since the model makes use of word-piece information, the target sentence is tokenized first. To obtain a representation of the sentence, the model makes use of the following feature for each word:

- **Dependency tag:** Dependency relationship between the head token and its child token.
- **POS tag:** Part-of-Speech tag of the token.
- **Lemma:** The base form of the token, with no inflectional suffixes.
- **Character Representation:** Encodes each character of the token, assigning an integer value according to its index in a vocabulary obtained in the train set. Padding is done at the end to ensure all words have the same number of characters
- **Word embedding of the token:** we consider 3 alternative words embeddings models:
  - BERT [1]: contextual embeddings with no further hyper tuning. BERT multilingual base model (cased) was used. The BERT model provides its tokenizer but its incompatibility with the rest of the implemented system made it necessary to make several modifications to it. We decide to not use its tokenization algorithm and use the output of the tokens produced for Spacy instead. Therefore, the encoder provided by BERT was used directly.

- FastText Spanish Medical Embeddings [6]: these embeddings were generated from Spanish corpora that include: (a) the full-text in Spanish available in SciELO.org (until December/2018), (b) all articles from the following Wikipedia categories: Pharmacology, Pharmacy, Medicine and Biology (during December/2018) and (c) the concatenation of the previous two corpora. Furthermore, for each of these datasets, two different models were trained using CBOW (*continuous bag-of-words*) and SkipGram representation, and each of these architectures was developed with cased and uncased words. Therefore, there was a total of 12 pre-trained embeddings. All these models were tested for this task, and SciELO SkipGram Uncased gave the best results.
- Character embeddings trained in the training set using as input the character representation feature.

Architecture. In the first instance, we use character-level information to capture morphological dependencies on the token. Having this information, every single word's vector can be formed, even if it is out-of-vocabulary words. This component takes as input a character representation consisting of a sequence of characters encoded as numbers. The character representation is passed as input to an Embedding Layer which output is processed by an LSTM layer.

Then the syntactic features (the dependency tag, the POS tag, and the lemma) are vectorized. These features together with the previously computed character level representation and optionally one of the word embeddings pre-trained models are concatenated for each token in the input sequence. These vectors are processed by two sequential Bi-LSTM layers to produce a sequence of vectors that encode the tokens in the input sentence.

The output of the last Bi-LSTM layer is passed as input of two Dense layers. The first Dense layer produces a sequence in the BILOUV tag scheme. The second Dense layer generates a tag for each entity type: *Concept*, *Action*, *Predicate*, *Reference*.

The learning of the model is done with 10 epochs in the training dataset, which has 1500 sentences. The final model had a total of 1,073,767 trainable parameters. *Adam* optimization is used with the default learning rate of 0.001. The loss function used is categorical cross-entropy as with most multi-class classification problems. The first LSTM processing character input had 20 units and a recurrent dropout of 0.5. The two Bi-LSTMs had 32 units and a recurrent dropout of 0.1 and 0.2 respectively. Both Dense layers had a *Softmax* activation function.

A summary of the NER model architecture is provided in Figure 1.

Output handling. The sequence of BILOUV tags and entity types produced by the two Dense layers is processed to get the list of entities expected as output for Task A. There is an important challenge in this process: tokens belonging to an entity are not necessarily continuous in the sentence. Taking this into account, the process of decoding is handled in two phases, based on the methodology

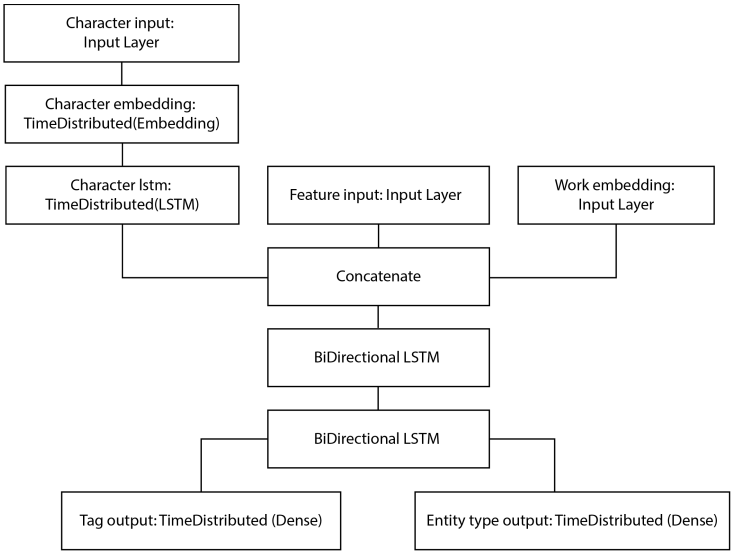

Fig. 1. NER model architecture

described by UH-MAJA-KD in the previous edition of the challenge [5]. First, two classes of discontinuous entities are extracted, one corresponding to entities that share their initial tokens and the other referring to those that share their final tokens. Entities matching to the former class are extracted using the regular expression $(VO^*)^+((I|O)^*L)^+$ and $(B(I|O)^*)^+(O^*V)^+$ expression relates to the latter. Afterward, the second phase starts assuming all the remaining entities appear as continuous sequences of tokens. To extract continuous entities, an iterative process is carried on over the tag sequence produced by the model assuming that the maximum overlapping depth is 2.

2.2 The Relation Extraction Model

The goal of this subtask is to discover semantic relationships between the entities detected and labeled in each sentence. In addition, every semantic relation has a source and target entity, therefore, the relation is directed, that is, the involved entities must match the correct direction.

To solve this task, all pairs of entities occurring in the same sentence are presented to the model. The absence of relations between a pair of entities is modeled with an additional relation type. Therefore, we used a multi-class approach that enabled us to predict whether a candidate pair is related to some of the relation classes available. One of the problems with this approach was that the negative instances (the absence of relation type) substantially exceed the positive ones leading to skewed class distribution. To mitigate the unbalance of the obtained dataset, we optionally employed a class-oriented weighting scheme

and reduced the negative sampling during the training phase. This way, the model gets to "pay more attention" to samples from an under-represented class.

Input handling. For the RE classifier, the following features were used for both the source and target entities presented to the model:

- **Entity type:** entity type of the key phrase according to the label it was assigned in the previous entity recognition task.
- **Dependency tag:** dependency relationship between the head token and its child token.
- **POS tag:** Part-of-Speech tag of the token.
- **Word embedding of the token:** The same two first alternatives of the previous model, consisting of pre-trained word embedding models were tested:
  - BERT [1]: BERT multilingual base model (cased) with no further hyper tuning was used.
  - FastText Spanish Medical Embeddings [6]: as in the previous task, all different models were tested.

For multi-word entities, the Lowest Common Ancestor (LCA) of the tokens in the dependency parse tree was used as the representative token of the entity, and only its syntactic features were processed.

In addition, to determine a possible relation between two entities, the system presented uses as input structures derived from the dependency parse tree associated with the target sentence, to obtain information from both the sentence and the entity pair:

- **Length of the path:** the distance between the source and target entity in the dependency parse tree.
- **Dependency path representation:** the path in the dependency parse tree is computed. Then, every dependency label is assigned an integer value. To ensure that all paths have the same number of nodes padding is added at the end.

Architecture. The syntactic features (dependency tag, POS tag, length of the dependency path) and the entity type are vectorized. These features together with one of the word embeddings of each pair of entities and the dependency path representation are concatenated. The vectors are then processed by a Bi-LSTM layer to encode the tokens and produce intermediate representations that capture dependencies between pairs of entities.

The resulting vector of the Bi-LSTM layer is processed by a final linear Dense layer, that produces as outputs the most probable type of relation between the involved entities.

A summary of the RE model architecture is shown in Figure 2.

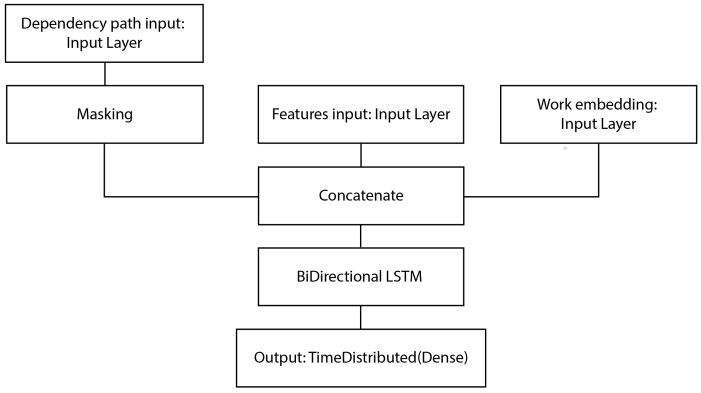

Fig. 2. RE model architecture

Like the NER model, the learning is done with 10 epochs in the training dataset, which has 1500 sentences. The final model had a total of 1,889,934 trainable params. *Adam* optimization is used with the default learning rate of 0.001. The loss function used was categorical cross-entropy. The Bi-LSTMs layer had 32 units and a recurrent dropout of 0.1. The Dense layer used *Softmax* as the activation function.

2.3 System Training

The training collection provided in the challenge was used to train the models. The development collection was used as an offline test set to evaluate our models and for fine-tuning.

Both models were implemented using Python programming language, with TensorFlow (v2.3.0) as the deep learning neural network library. BERT contextual embeddings were obtained from the `bert-base-multilingual-cased` pre-trained model using `torch` (v1.8.1) and `transformers` (v4.5.0) libraries. FastText (v0.9.2) Python library was used to load the language identification pre-trained model and the FastText Spanish Medical Embeddings pre-trained models. The tokenization of sentences and the extraction of syntactic and semantic features was done using spaCy (v3.0.5).

The training process was done on a personal computer with the following stats: 8 core Intel(R) Core(TM) i5-8250U CPU at a frequency of 1.60GHz, with a memory of 8.00GB with no GPU available for CUDA. The total training time for the entity model took about 5 minutes, while the relation model was close to 30 minutes.

3 Results

The evaluation in both tasks was carried out using the annotated corpus proposed in the challenge. The results were measured with a standard F1 measure as described in the challenge overview.

Table 1 presents the official results of the competition, given by the evaluation of scenario 1. As it can be seen, with an overall F1 score of 0.338 our system was ranked as fifth-best.

Team	F1	Precision	Recall
Vicomtech	0.53106	0.54075	0.53464
PUCRJ-PUCPR-UFMG	0.52835	0.56849	0.50276
IXA	0.49886	0.46457	0.53863
uhKD4	0.42264	0.48529	0.37431
UH-MMM	**0.33865**	**0.29163**	**0.40374**
Codestrange	0.23201	0.33703	0.17689
baseline	0.23201	0.33703	0.17689
JAD	0.10949	0.23441	0.07143

Table 1: Results of the Main Scenario evaluating task A and B

Tables 2 and 3 show the results of scenarios 2 and 3, where Task A and B were evaluated independently. Our system was able to reach the fourth on the task A evaluation scenario and, although achieved the fifth place on scenario 3, presented way lower results than the fourth place.

Team	F1	Precision	Recall
PUCRJ-PUCPR-UFMG	0.70601	0.71491	0.69733
Vicomtech	0.68413	0.69987	0.74706
IXA	0.65333	0.61372	0.6984
UH-MMM	0.60769	0.54604	0.68503
uhKD4	0.52728	0.51751	0.53743
Yunnan-Deep	0.33406	0.52036	0.24599
baseline	0.30602	0.35034	0.27166
JAD	0.2625	0.31579	0.2246
Yunnan-1	0.17322	0.27107	0.12727
Codestrange	0.08019	0.415	0.04439

Table 2: Results of Scenario 2 evaluating task A

Team	F1	Precision	Recall
IXA	0.4304	0.45357	0.40948
Vicomtech	0.37191	0.54186	0.28311

Team	F1	Precision	Recall
uhKD4	0.31771	0.55623	0.22236
PUCRJ-PUCPR-UFMG	0.26324	0.36659	0.20535
UH-MMM	**0.05384**	**0.07727**	**0.04131**
Codestrange	0.03275	0.4375	0.01701
baseline	0.03275	0.4375	0.01701
JAD	0.00722	0.375	0.00365

Table 3: Results of Scenario 3 evaluating task B

4 Discussion

Several models were trained in the training collection and tested in the development collection. For each task, different word embeddings pre-trained models were used: BERT multilingual model and FastText Medical Word Embedding for Spanish. The multi-lingual approach of the challenge made it very inefficient to use language-specific embeddings, thus no increase in overall F1 was seen using the FastText embeddings in the first task. The use of BERT didn't improve either the performance obtained. As a result, for this task, we only used the character-level information computed. The final submission of our system didn't use any of the pre-trained models proposed. The results obtained testings these embeddings in the development set can be seen in Table 4.

Embeddings	Recall	Precision	F1
SciELO SkipGram Cased	0.7239	0.4671	0.5678
SciELO SkipGram Uncased	0.7086	0.5055	0.5903
SciELO CBOW Cased	0.6932	0.4149	0.5191
SciELO CBOW Uncased	0.6861	0.3981	0.5038
SciELO+Wiki SkipGram Cased	0.6987	0.4387	0.5390
SciELO+Wiki SkipGram Uncased	0.702	0.4292	0.5327
SciELO+Wiki CBOW Uncased	0.6937	0.4013	0.5084
Wiki SkipGram Cased	0.6932	0.4224	0.5249
Wiki SkipGram Uncased	0.6937	0.4661	0.5576
Wiki CBOW Cased	0.7014	0.4062	0.5145
Wiki CBOW Uncased	0.7091	0.3956	0.5079
BERT	0.6114	0.4736	0.5338
No Embedding	0.6806	0.5268	0.5939

Table 4: Results of the NER model using different embeddings in the development collection.

In the second task, the FastText and BERT embeddings were also tested. However, the incorporation of the BERT model made it unable to complete in

time the run in scenario 3. With one of FastText's embeddings, we obtained a slight improvement in this task despite the language constraints, therefore, this was the embedding used in the final submission. The results testing these embeddings are shown in Table 5. 5.

Embeddings	Recall	Precision	F1
Scielo SkipGram Cased	0.0376	0.04161	0.03951
Scielo SkipGram Uncased	0.02703	0.05011	0.03511
Scielo CBOW Cased	0.0.02233	0.06859	0.03369
Scielo CBOW Uncased	0.03173	0.04584	0.03750
Scielo+Wiki SkipGram Cased	0.02115	0.0602	0.0313
Scielo+Wiki SkipGram Uncased	0.04465	0.07211	0.05515
Scielo+Wiki CBOW Uncased	0.0188	0.04051	0.02568
Wiki SkipGram Cased	0.03055	0.0552	0.03933
Wiki SkipGram Uncased	0.05875	0.04822	0.05297
Wiki CBOW Cased	0.0329	0.07254	0.04526
Wiki CBOW Uncased	0.02585	0.06111	0.03633
No Embedding	0.05288	0.05325	0.05307

Table 5: Results of the RE model using different embeddings in the development collection.

To tackle the class unbalanced problem encountered in task B we tested two techniques: class weighting and reduce negative sampling. These two techniques improved the system performance, however, we think the poor results obtained in this task show this problem wasn't completely solved. In addition, another of the reasons for these results can be the lack of more contextual features.

Finally, regarding the training process, it is worth noting the fact that the training time of the RE model is significantly longer than the NER model. This is somewhat expected since our approach for task B takes more training examples, defining as training instances each pair of entities in the sentence. In addition, the computation of LCA and the path between the pair of entities is very time-consuming. The long training time of this model was one of the reasons why deeper and more complex architectures weren't tested.

5 Conclusions

In this paper, we have described the main characteristics of the model that was developed for the UH-MMM team's submission to IberLEF's 2021 eHealth Knowledge Discovery shared task, where two main NLP tasks were defined: entity recognition and relationship extraction. Three evaluation scenarios involving the combination of these tasks were also developed.

Our proposal follows a deep learning approach for both tasks. It is focused on the use of a BiLSTM+Dense neural network where different word embeddings

are combined as input to the architecture. For Task A, this neural network was trained by using the annotated dataset provided by the organization, it was then tokenized and tagged using the BILOUV scheme. Syntactic and character-based features were used. Task B was addressed in a pairwise-query fashion, encoding information about the involved pair of entities using linguistic and syntactic features derived from the dependency parse tree, and employing a BiLSTM+Dense model. This system obtained a competitive performance on Scenario 2, where it was located in fourth place. However, our proposal revealed a weakness for the relationship extraction task, obtaining fifth place with a big difference concerning the fourth place. We need to analyze in detail if the problem lies in the class unbalanced problem or the lack of more contextual features.

It is proposed as future work to study the performance of the model using more contextual and semantic features as input of the neural network, as well as the use of other types of word embeddings. Furthermore, we will try to improve the relation extraction task by implementing another neural network that captures in a better way the relationship between concepts.

References

1. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. vol. abs/1810.04805. CoRR (2018), http://arxiv.org/abs/1810.04805
2. Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., Mikolov, T.: Fasttext.zip: Compressing text classification models (2016)
3. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification (2016)
4. Piad-Morffis, A., Gutiérrez, Y., Estevez-Velarde, S., Almeida-Cruz, Y., Muñoz, R., Montoyo, A.: Overview of the eHealth Knowledge Discovery Challenge at IberLEF 2021. Procesamiento del Lenguaje Natural **67**(0) (2021)
5. Rodríguez-Pérez, A., Quevedo-Caballero, E., Mederos-Alvarado, J., Cruz-Linares, R., Consuegra-Ayala, J.P.: UH-MAJA-KD at eHealth-KD Challenge 2020: Deep Learning Models for Knowledge Discovery in Spanish eHealth Documents (2020)
6. Soares, F., Villegas, M., Gonzalez-Agirre, A., Krallinger, M., Armengol-Estapé, J.: Medical word embeddings for Spanish: Development and evaluation. In: Proceedings of the 2nd Clinical Natural Language Processing Workshop. pp. 124–133. Association for Computational Linguistics, Minneapolis, Minnesota, USA (Jun 2019). https://doi.org/10.18653/v1/W19-1916, https://www.aclweb.org/anthology/W19-1916