Named Entity Recognition for LivingNER-Species based on BERT and Span Detection

Song Han[1], Haiyan Ding[1,*]

[1]*Information Science and Engineering, Yunnan University, Yunnan 650091, P.R.China*

Abstract

Recently, organisms/species have been attracted to NLP studies, especially for non-English content. So, we need high-quality guidelines and corpora to determine where species or humans are in medical text. Task 1 of LivingNER (Species mention entity recognition) gives a plain text clinical case report document collection, participants must return the exact character offsets of all species mentions, both human and non-human. So, this paper tries to figure out a satisfactory way by using a model based on BERT and span detection.

Keywords

NLP, non-English, BERT, span

1. Introduction

Currently, organisms/species have scarcely featured in NLP studies, particularly non-English content. High-quality guidelines and corpora providing granular text-bound semantic annotations are needed to leverage mentions of living creatures in biomedical texts.

The annotation of species of living organisms is critical to scientific disciplines like medicine, biology, ecology/biodiversity, nutrition, and agriculture. And scientists have figured out LINNAEUS and the SPECIES tool to detect the difference of species mentions [1, 2, 3]. However, it is very difficult to adapt to non-English languages due to its special rule-based systems and pattern matching. What's more, there are not enough resources and evaluation scenarios in other languages[4]. And researchers have met a lot of challenges such as name changes, homonymy with commonly used words, abbreviations, incorrect cases or misspelled names, and so on. So, in task 1 of LivingNER Shared Task (IberLEF2022), this paper conducts more in-depth research about this task to find out which parts of the text are human or species or neither them. To carry out this task, this paper utilizes an entity recognition approach based on BERT and span detection[5, 6, 7, 8].

This paper is organized as follows. In the second section, this paper introduces previous research on how to figure out challenges about named entity recognition. The third section talks about how to preprocess raw data from this task. And in the third section, this paper

IberLEF 2022, September 2022, A Coruña, Spain.

2452106319@qq.com (S. Han); teidhy@163.com (H. Ding)
https://github.com/songhan123123/ner (S. Han)

introduces the datasets for task 1, instruction of model and experiment results. Conclusions are drawn in section 4.

2. Related Works

At the beginning of research about Named entity recognition, people used to utilize a single model or relationship classification, such as RNN[9], transformer[10] and so on. With the development of research of NER, researchers have proposed more and more combining models to get better performances. For example, the first approach is a combination of Bi-LSTM and CRF[11, 12], and the other is a BILOU-based combination of a bidirectional LSTM and a CNN to finish the extraction of the feature representation of the input sentence at a high level[13]. Researchers about NER studies have not only proposed new models but also have found many amazing ways to promote the development of NER research. Xiaoya Li and his team have proposed a new approach to the MRC Framework for Named Entity Recognition[13]. They found that when they have added a query of prior knowledge to a sentence, we can get a better performance in most NER classic datasets.

Recently, a classic model based on Transformers' pre-training in the vast majority of natural language processing tasks has very eye-catching performances. Transformers mainly use an attention mechanism, which can calculate complex information without using RNN[14]. In addition, the pre-training model of Transformers can greatly save the time and training data required for training. Fine-tuning has greatly improved the efficiency of model training. When released in 2018, BERT, as the most SOTA language model, performed well among many tasks. In the current situation of various more advanced models, many researchers can still use BERT as the baseline model of their work[14]. Based on BERT, Markus Eberts and Adrian Ulges further proposed a span-based combined entity and relationship extraction model based on the Transformers pre-training model. The span-based physical recognition method proposed by the author can identify the overlapping entity. The traditional method is classifying each word into a class using BIO or other labeling methods. Every word should be classified as the start of entity or end of an entity or non-entity, so the total number of categories is 2*n+1 (n is the number of categories of the entity), so it is a span-based method that trains entity and relationship category simultaneously. The idea is to form all the spans in a sentence and obtain the SPAN's characteristics, classify the physical category of the SPAN, and then combine the two entity combinations to form a representation of the relationship and classify the relationship. [15].

3. Experiment

3.1. Datasets

Task 1: Livingner corpus includes 2,000 clinical cases in more than 10 medical fields. These areas are marked with species mentioned, using the NCBI classification method for mapping and used in Spanish medical documents[2, 4]. It is composed of 1,000 clinical case reports extracted from various medical majors. The test set consists of 500 clinical case reports. Training sets and test sets include COVID, oncology, infectious diseases, tropical medicine, urology, pediatrics,

etc. And the training sets and test sets of task 1 are distributed in the two watchmaking separate (TSV)files, respectively. There is one line for each annotation, including the following columns:

- `filename`: Document Name
- `mark`: The identifier refers to the mark
- `label`: type
- `off0`: The starting position of the entity in the documentation
- `off1`: The end of the entity in the documentation

The annotation of task 1 is shown in Figure 1:

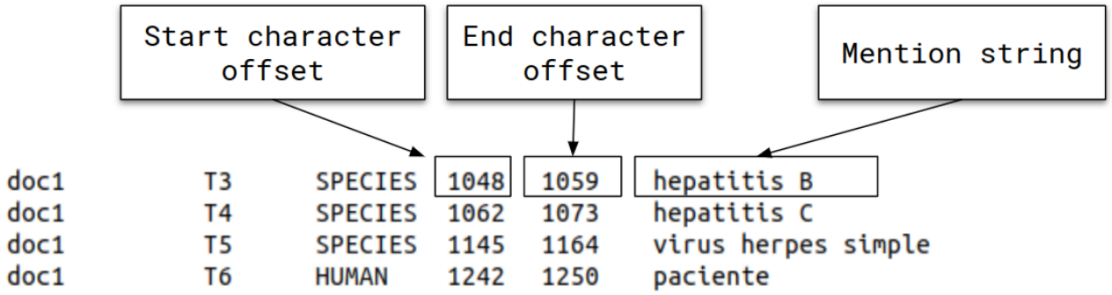

Figure 1: Annotation example of LivingNER.

3.2. Data Processing

The training set consists of a text file that contains 1,000 clinical case reports and a TSV file marked with 1,000 text files. Similarly, the test set is marked and the TSV file is composed. The purpose of the data pre-processing is to remove the text file from the file and divide it. Because the length of the maximum training set sequence is 128, and a large part of each line in the original text file has exceeded 128. Without losing order information, each row's text needs to be divided into smaller sequences, so this experiment uses a comma, ending, and alternating the lines to divide the data of each row into a smaller sequence without affecting the sentence integrity. The next step is to convert each sentence that is divided into a token. Since the location of each word of the token and the atomic sentence may be too different to restore the starting position and end position of the original word so the corresponding token is needed to restore to the original word position. This experiment uses this here to save each word into a list, and then all the words in one list sentence. The list of compositions is stored in a list so that when the training position is trained, and the list location where the token is located can be restored to the original real entity. After completing the above work, excluding physical sentences are deleted. Before generating test files, this experiment needs to store each sentence divided into a dictionary, because it needs physical file information and location information when generating prediction documents. Algorithm 1 gives the famework of training or validation set preprocessing. And Algorithm 2 gives the framework of test set preprocessing.

Finally, Algorithm 3 gives the framework of prediction results processing.

Algorithm 1 Framework of training or validation set preprocessing.

Input: 1000 text files and training_entities_subtask1.tsv for training set or 500 text files and validation_entities_subtask1.tsv for validation set;

Output: training set or validation set;

1: Reading the text in each file line by line, traversing all files;
2: Dividing each line of text according to punctuation, limit the sentence length to at least 16 or more, and store the divided sentences of each line in the list;
3: Reading training_entities_subtask1.tsv or validation_entities_subtask1.tsv line by line, remove the tabs and store it in a list;
4: Searching the entities in the list obtained in step 3 in the list in step 2, and store the text of each sentence containing the entity in the list in step 2, the entity, and the position of the entity in the sentence into a dictionary;
5: Storing the dictionary obtained in step 3 into another list;
6: Saving all the dictionaries in the list into a file to get a training set or a validation set;
return training set or validation set;

Algorithm 2 Framework of test set preprocessing.

Input: 13467 text files;

Output: test set;

1: Reading the text in each file line by line, traversing all files;
2: Dividing each line of text according to punctuation, limit the sentence length to at least 16 or more, and store the divided sentences in each line into a list;
3: Traversing the list obtained in step 2, traverse each element in the list and store it in a dictionary, and add the id of the text, the file name, and the position of the element in the file to each dictionary;
4: Storing all the dictionaries obtained in step 3 into a list;
5: Saving all the dictionaries in the list into a file to get the test set;
return test set;

3.3. Model Introduction

Since the span-based model proposed by Markus Eberts and Adrian Ulges achieved the best performance on several datasets, this paper will also focus on the span-based model[15]. In this model, the model passes a layer of dropout layer after the BERT pre-training model to prevent the model from overfitting and improve its generalization ability of the model. The pre-trained model is a BERT model trained on a large Spanish corpus. BETO is similar in size to BERT and is trained using full word masking techniques[16]. And then each word is classified through a classifier to determine the start of the entity, the specific position of the location and the category of the entity, the result after the classifier, and the dropout layer's output are spliced. Then a linear layer and layer norm operation are performed to ensure the stability of the data feature distribution. Finally, a linear layer is used to get the end position of the entity. The structure of the model is shown in Figure 2. Algorithm 4 gives the framework for calculating the end and start of an entity by the model:

Algorithm 3 Framework of prediction results processing.

Input: The dictionary predicted by the model, the dictionary contains ID, file name, entity type, entity, the start position and end position of the entity, and the serial number of the entity in the modified file;

Output: TSV file containing prediction results;

1: Storing all predicted dictionaries in a list;
2: Traversing the list and convert all dictionaries into the same format as training_entities_subtask1.tsv or validation_entities_subtask1.tsv. Each piece of data contains the following fields: filename (the name of the file where the entity is located), mark (the serial number of the entity in the file), label (the category of the entity), off0 (the starting position of the entity in the file), off1 (the ending position of the entity in the file), span (entity), save the above fields of each data into a list;
3: Storing all the lists obtained in step 2 into a list;
4: Traversing the list and write the data to the TSV file;
return TSV file containing prediction results;

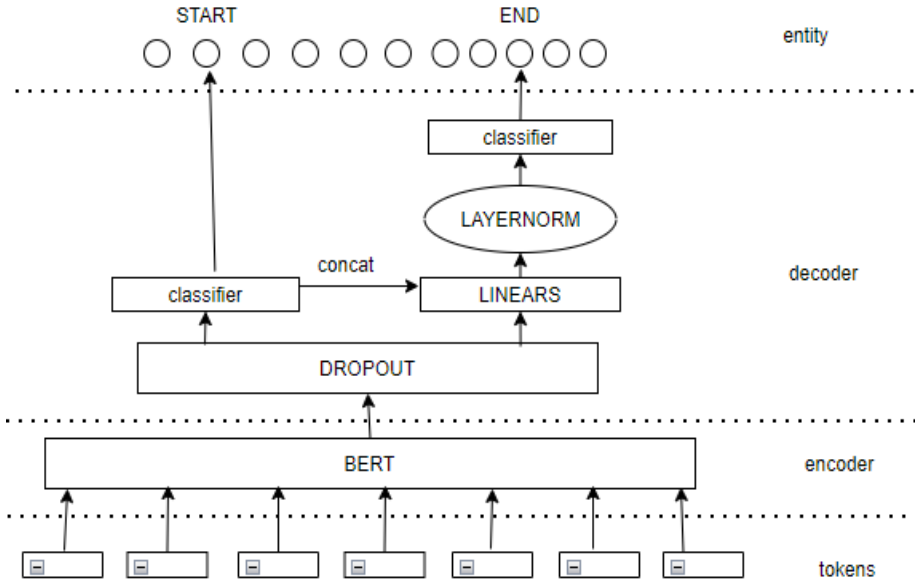

Figure 2: BERT network structure diagram based on span

3.4. Experiment Descriptions

The main hyperparameters of the model are shown in Table 1.

where "loss_type" represents the type of loss function of the model, "lsr" represents LabelSmoothingCrossEntropy() and "focal' means FocalLoss()[17], 'ce' stands for CrossEntropyLoss(). 'train_max_seq_length' represents the maximum sequence length of the training, 'eval_max_seq_length' represents the maximum sequence length of the validation, 'batch_size'

Algorithm 4 Framework of Calculating the end and start of an entity by the model.

Input: input_ids;attention_mask;token_type_ids;num_classes;
Output: start,end;

$outputs \leftarrow BERT(input\_ids, attention\_mask, token\_type\_ids)$;
$sequence\_output \leftarrow outputs[0]$;
$sequence\_output \leftarrow dropout(sequence\_output)$;
$start\_logits \leftarrow Linear(sequence\_output, num\_classes)$;
$label\_logits \leftarrow softmax(start\_logits, -1)$;
$end\_logits \leftarrow Linear(concat([sequence\_output, label\_logits], dim = -1))$;
$end\_logits \leftarrow LayerNorm(sequence\_output)$;
$end\_logits \leftarrow Tanh(end\_logits)$;
$end\_logits \leftarrow Linear(sequence\_output, num\_classes)$;
$start \leftarrow argmax(start\_logits, -1).numpy()[0][1:-1]$;
$end \leftarrow argmax(end\_logits, -1).numpy()[0][1:-1]$;
return start, end;

Table 1
Main hyperparameters configuration of Model

Hyperparameters	value
loss_type	ce
train_max_seq_length	128
eval_max_seq_length	512
batch_size	24
learning_rate	2e-4
num_train_epochs	4
seed	42

represents the number of batch size of the training and verification process on each GPU, and the 'learning_rate' indicates the learning rate of the model training process, 'num_train_epochs' represents the epoch number of the training. The experiment mainly explored the effect of 'loss_type', 'batch_size', 'learning_rate', and 'num_train_epochs' on the model effect. So l firstly explore the impact of 'loss_type' on the model. Based on the experiment's performance, this experiment chose CrossEntropyLoss() as the model loss function. The result is the best when 'learning_rate' is 2e-4, so this experiment chooses 2e-4 as the model's learning_rate. The optimal batch size is 24. Next, this experiment found this model converges at the 4th epoch, so 4 is chosen as the 'num_train_epochs', and the performance of the validation sets is shown in Table 2 after training 4 epochs.

Table 2
Eval results.

	LivingNER NER			NER only SPECIES			NER only HUMAN		
	acc	R	F1	acc	R	F1	acc	R	F1
Eval results	0.8326	0.4685	0.5996	0.8506	0.7239	0.7822	0.7643	0.1888	0.3027

3.5. Prediction Results

Significantly, 'acc' is the abbreviation of accuracy in table 2. After training the model according to the hyperparameters in Table 1, the model got the first prediction result, due to failure to deal with the heading containing only two words as a single line during the preprocessing. The previous method of dividing the sentence by punctuation failed to separate the title from the content, resulting in entities can not be found in the file, so a large part of the entities failing to get all, which results in unsatisfactory results for the first prediction, the prediction score of the first prediction file is shown in Table 3.

Table 3
The first prediction results.

	LivingNER NER			NER only SPECIES			NER only HUMAN		
	P	R	F1	P	R	F1	P	R	F1
first-prediction results	0.526	0.0396	0.0736	0.0563	0.0019	0.0036	0.6951	0.0905	0.1601

And then prediction file is submitted with the same hyperparameters for the second time and gets another prediction score, as shown in Table 4.

Table 4
The post-workshop.

	LivingNER NER			NER only SPECIES			NER only HUMAN		
	P	R	F1	P	R	F1	P	R	F1
post-workshop	0.5399	0.1965	0.2881	0.1111	0.0202	0.0342	0.725	0.4343	0.5432

As can be seen from the data in Tables 3 and 4, the accuracy of the two prediction files is about the same, but their recall scores are very different. The reason why the first prediction score is lower than the second prediction score may be that the first submission of the prediction file predicts that the prediction is correct for the entity is less than the second. Therefore, the recall score also increased after the increase of the correct entity predicted in the second prediction file, however, the result of the second prediction is still not satisfactory, the main reason may be that the division of sentences in the pre-processing process does not retain the information of the sentence and the entity very well, and later this experiment modify the pre-processing process of the data to make the sentence division more efficient, another possibility The reason is that this experiment did not handle the data imbalance well, and according to the prediction results, the score of the species is much lower than that of the entity of the human class, which may be due to the imbalance of data, so this experiment introduced focalloss in the follow-up experiment Trying to solve the data imbalance problem, the effect improved, and after 4 epochs, the F1 score increased by 0.002, With an F1 score of 0.599 on the validation set, this experiment tries to do experiments to find out the hyperparameters that perform better so that the data imbalance problem can be better solved using focalloss[17] At the same time, this experiment also try to continue to improve the pre-processing process of the data, dividing the sentences into longer lengths, and found that the effect was slightly improved, and the F1 score increased by 0.001 , reaching 0.600, which also verifies conjecture, the failure to pre-process well and the

failure to solve the data imbalance problem well is the shortcomings of this experiment, and also points the way for improving related work in the future.

4. Conclusion

This paper introduces the submission of the Task 1 task of LivingNER Shared Task (IberLEF2022), mainly using the combination of span and BERT to carry out the named entity recognition task of task 1, and the hyperparameters of the model are tuned. And the prediction score was also displayed, with an accuracy of 53.99%, and the prediction score was analyzed, mainly analyzing the possible deficiencies that the approach had in this task and trying to follow up the work by verifying the conjecture and pointing the way to get better performance in later work.

References

[1] M. Gerner, G. Nenadic, C. M. Bergman, Linnaeus: a species name identification system for biomedical literature, BMC bioinformatics 11 (2010) 1–17.

[2] S. Federhen, The ncbi taxonomy database, Nucleic acids research 40 (2012) D136–D143.

[3] E. Pafilis, S. P. Frankild, L. Fanini, S. Faulwetter, C. Pavloudi, A. Vasileiadou, C. Arvanitidis, L. J. Jensen, The species and organisms resources for fast and accurate identification of taxonomic names in text, PloS one 8 (2013) e65390.

[4] C. L. Schoch, S. Ciufo, M. Domrachev, C. L. Hotton, S. Kannan, R. Khovanskaya, D. Leipe, R. Mcveigh, K. O'Neill, B. Robbertse, et al., Ncbi taxonomy: a comprehensive update on curation, resources and tools, Database 2020 (2020).

[5] A. Miranda-Escalada, E. Farré, M. Krallinger, Named entity recognition, concept normalization and clinical coding: Overview of the cantemist track for cancer text mining in spanish, corpus, guidelines, methods and results., IberLEF@ SEPLN (2020) 303–323.

[6] S. Lima-López, E. Farré-Maduell, A. Miranda-Escalada, V. Brivá-Iglesias, M. Krallinger, Nlp applied to occupational health: Meddoprof shared task at iberlef 2021 on automatic recognition, classification and normalization of professions and occupations from medical texts, Procesamiento del Lenguaje Natural 67 (2021) 243–256.

[7] A. J. Yepes, A. Albahem, K. Verspoor, Using discourse structure to differentiate focus entities from background entities in scientific literature, in: Proceedings of the The 19th Annual Workshop of the Australasian Language Technology Association, 2021, pp. 174–178.

[8] S. Pyysalo, T. Ohta, R. Rak, D. Sullivan, C. Mao, C. Wang, B. Sobral, J. Tsujii, S. Ananiadou, Overview of the infectious diseases (id) task of bionlp shared task 2011, in: Proceedings of BioNLP Shared Task 2011 Workshop, 2011, pp. 26–35.

[9] W. Zaremba, I. Sutskever, O. Vinyals, Recurrent neural network regularization, arXiv preprint arXiv:1409.2329 (2014).

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[11] Z. Huang, W. Xu, K. Yu, Bidirectional lstm-crf models for sequence tagging, arXiv preprint arXiv:1508.01991 (2015).
[12] Q. Liao, J. Wang, J. Yang, X. Zhang, Ynu-hpcc at ijcnlp-2017 task 1: Chinese grammatical error diagnosis using a bi-directional lstm-crf model, in: Proceedings of the IJCNLP 2017, Shared Tasks, 2017, pp. 73–77.
[13] J. P. Chiu, E. Nichols, Named entity recognition with bidirectional lstm-cnns, Transactions of the association for computational linguistics 4 (2016) 357–370.
[14] J. D. M.-W. C. Kenton, L. K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of NAACL-HLT, 2019, pp. 4171–4186.
[15] M. Eberts, A. Ulges, Span-based joint entity and relation extraction with transformer pre-training, in: ECAI 2020, IOS Press, 2020, pp. 2006–2013.
[16] J. Canete, G. Chaperon, R. Fuentes, J.-H. Ho, H. Kang, J. Pérez, Spanish pre-trained bert model and evaluation data, Pml4dc at iclr 2020 (2020) 1–10.
[17] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2980–2988.