A Parallel-Attention Model for Tumor Named Entity Recognition in Spanish

Tong Wang[a], Yuanyu Zhang[a] and Yongbin Li[b]

[a]*School of Information Science and Engineering, Yunnan University, Kunming 650091, P.R.China*
[b]*School of Medical Information Engineering, Zunyi Medical University, Zunyi 563000, P.R.China*

Abstract
Named Entity Recognition is one of the subtasks of Natural Language Processing, which aims to locate and classify named entities in text into pre-defined categories. The CANTEMIST 2020 is a task for tumor named entity recognition and we propose a new model for this task. We use Recurrent Neural Networks and Convolutional Neural Networks to extract relevant text features. Then dynamic attention mechanism is used to merge features extracted from these two structures. In the final evaluation, we achieve an F1-score of 0.746.

Keywords
tumor named entities recognition, clinical records, Natural Language Processing, parallel-attention structure

1. Introduction

With the development of the Biomedical Named Entity Recognition (BNER) task, it has gradually become an important means for auxiliary treatment and diagnosis, and tumor named entity recognition is one subtask of BNER. Due to the protection of patient privacy and the confusion of clinical records, there are few data sets about tumor named entity recognition. The CANTEMIST 2020 [1] is a great opportunity to study tumor named entity recognition, and we participated in the CANTEMIST-NER subtask, which requests to find the mentioned tumor morphology in clinical records.

Currently, the main method for BNER tasks is to adopt BiLSTM-CRF [2] and various fine-tuning models for BERT [3]. To achieve more lightweight, our team proposed a parallel-attention model based on Recurrent Neural Networks (RNN) [4] and Convolutional Neural Networks (CNN) [5]. On the other hand, the tumor named entity recognition task has out-of-vocabular (OOV) problem. The clinical records contain a large amount of professional and unstructured texts, which cannot be represented by pre-training words. So we use character embedding to supplement word embedding to alleviate the impact of the OOV problem. Finally, our model achieves excellent performance.

Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020)
EMAIL: 1480467828@qq.com (T. Wang); 879411813@qq.com (Y. Zhang); 1957405@qq.com (Y. Li)
URL: https://github.com/18720936539 (T. Wang)
ORCID: 0000-0002-0877-7063 (T. Wang)

2. Related Work

Cancer is one of the diseases with the highest mortality rate, and the number of people suffering from cancer are increasing year by year. With the rapid development of natural language processing (NLP), it has become an essential means of early cancer diagnosis, which can effectively prevent the deterioration of the tumor. The BNER task is a prerequisite task for other NLP tasks on cancer, and its performance significantly affects the final diagnosis. Tumor named entity recognition is a subtask of BNER in the tumor field, and its experimental method is similar to the BNER task.

The current main methods for BNER are machine learning methods and deep learning methods. For example, Makino et al. [6] use SVM and Settles et al. [7] use CRF to process the BNER task. However, the feature engineering of machine learning methods relies on manual processing, and the generalization ability of the model is not enough, which leads to its effect only on specific tasks.

With the rapid development of Neural Networks (NN), people have gradually abandoned traditional methods to deal with BNER tasks and replaced them with NN methods. Sahu et al. [8] uses the RNN network to process the BNER task and achieve a significant improvement. Luo L et al. [9] uses An attention-based BiLSTM-CRF approach to solve the BNER task. These and many other experiments fully prove the effectiveness of RNN for BNER tasks. Later, Jacob Devlin et al. [3] creates BERT which achieves significant success on the NER task. Subsequently, Lee et al. [10] applies the BERT model to BNER and proposes Biobert, which achieves great success and proves the effectiveness of BERT for BNER task. But BERT generates a large number of parameters and training time. In this context, it becomes a meaningful work to implement a lightweight model with good performance based on RNN.

3. Our Model

Our model includes four parts: the embedding layer, the parallel layer, the attention structure, and the classification layer. In the embedding layer, we vectorize the text sequences by using word embedding and character embedding. The RNN and CNN modules are used to extract context information respectively in the parallel layer, and then we use the dynamic attention structure to merge these contextual information. Finally, the dense layer acts as a classifier to output the final prediction. The overall model structure is shown in Figure 1.

3.1. Embedding layer

In order to alleviate the OOV problem, our model combines word embedding and character embedding to obtain OOV information in the embedding layer. The character embedding is initialized by random feature vector, while word embedding is initialized by Spanish Medical Word Embeddings [11] that is train on SciELO and Wikipedia Health through fasText [12]

For the word embedding, we denote the input sentence as $s = [s_1, ..., s_i, ..., s_l]$, where s_i is the i-th token, as well as l is the number of token in the sentence. The word embedding w_i is expressed as follows:

$$w_i = wordembedding(s_i),$$

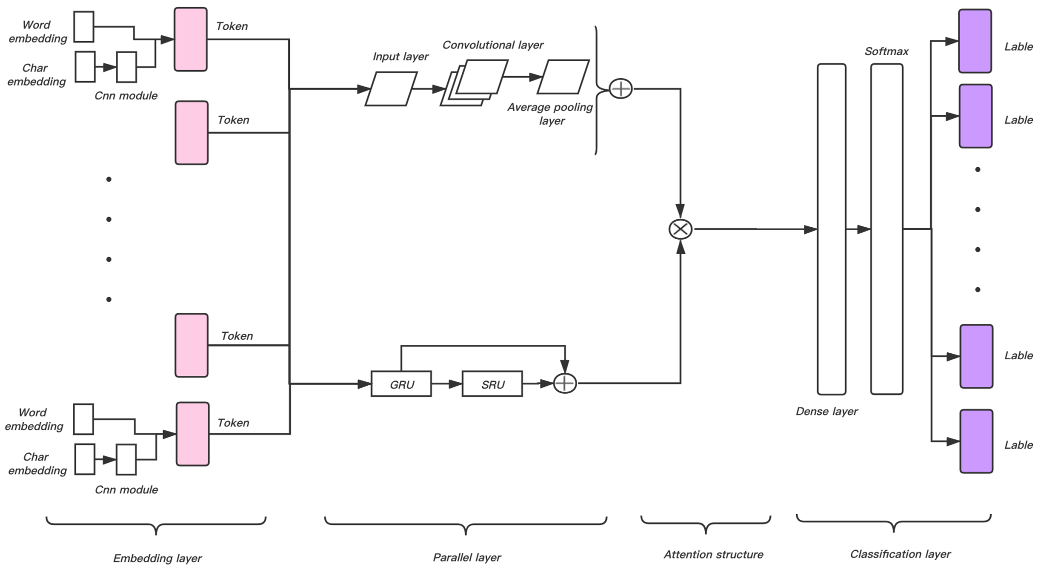

Figure 1: Our model structure

where $wordembedding()$ is the word embedding function. Then we use character embedding to supplement word embedding to obtain OOV information. The character embedding c_i is expressed as follows:

$$t_i = [c^i_1, ..., c^i_j, ..., c^i_n],$$

$$c_i = charembedding(t_i),$$

where $charembedding()$ is a character embedding function to randomly initialize the token t_i, n is the number of characters in the token t_i, and j represents the j-th character in the token t_i. Because the dimensions of word embedding and character embedding are inconsistent, they can not be merged directly. Here we use CNN to fine-tune the character embedding, and the specific step is as follows:

$$c^i_j = conv([c^i_{j-k/2}, .., c^i_j, ..., c^i_{j+(k/2)}]),$$

where k represents the size of the convolution kernel, and k is set to 3 in this paper. Finally, we concatenate word embedding w_i and character embedding c_i to obtain the final token features s_i.

3.2. Parallel layer

The parallel layer is aimed to extract the context-dependence information of the text sequence. It consists of an RNN module to obtain the long-distance dependence information and a CNN module to obtain the local dependence information.

RNN module: Whether in industry or academia, long short-term memory (LSTM) [13] has proven its excellent performance in NER, but LSTM adds a forget gate mechanism, which introduces more parameters and increases training costs. To make the model more lightweight while ensuring outstanding performance, we combine Gated Recurrent Unit (GRU) [14] and Simple Recurrent Unit (SRU) [15] in this model to replace the traditional LSTM. Firstly, the output sequence $s = [s_1, ..., s_n]$ of the embedding layer is used as the input of the GRU, and the output of the GRU can be expressed as follows:

$$z_i = \sigma(W_z \cdot [g_{i-1}, s_i]),$$

$$a_i = \sigma(W_r \cdot [g_{i-1}, s_i]),$$

$$\tilde{g_i} = tanh(W_g \cdot [a_i * g_{i-1}, s_i]),$$

$$g_i = (1 - z_i) * g_{i-1} * \tilde{g_i},$$

where W_z, W_r, and W_g are parameter matrix for calculating the update gate z_i, the reset gate r_i, and the new memory $\tilde{g_i}$, and then we can get the output $g = [g_1, ..., g_i, ...]$ of the GRU. In order to extract long-distance dependence information more comprehensively, the model stacks the GRU and the SRU. At the same time, considering the overfitting caused by ordinary stacking and the disappearance of SRU gradient, the output of GRU is added as a penalty to the output of SRU. After the addition operation, the output of the final RNN module is as follows:

$$f_i = \sigma(W_f g_i + V_f c_{i-1} + b_f),$$

$$c_i = f_i c_{i-1} + (1 - f_i)(W g_i),$$

$$b_i = \sigma(W_r g_i + V_r c_{i-1} + b_r),$$

$$s_i = b_i C_i + (1 - b_i) g_i,$$

$$r_i = s_i + g_i,$$

where W, W_f, W_r are parameter matrix, and V_f, V_r, b_f and b_r are parameter vector in SRU. $r = [r_1, ..., r_i, ...]$ is the final output of the entire RNN module.

CNN module: Because of the characteristics of convolution, CNN can not capture the long-distance dependence information in the text sequence. However, due to the window sliding mechanism in convolution operation, we can obtain obvious local features by controlling the size of convolution kernel. In the CNN module, it extracts local information between different distances by setting three convolutional layers with different kernel sizes. And through a lot of tuning experiments, we set the size of the three convolution kernels to 3, 5 and 7 respectively. Then we perform data compression through average pooling operation while reducing data redundancy. Finally, the results of three convolution layers are added to get the output c of CNN module

3.3. Attention structure

Similar to the self-attention mechanism [16], weighted attention can dynamically assign different weights to long-distance dependence information and local dependence information in the current sequence for data compression and enhancement of effective data. Firstly, we execute the activation function $tanh$ on the r output by the RNN module and the c output by the CNN module, and then add the two parts as the attention matrix a. The specific steps are as follows:

$$a_r = tanh(W_r r + b_r),$$

$$a_c = tanh(W_c c + b_c),$$

$$a = a_r + a_c,$$

where W is weight and b is bias term. Through the weight control of the activation function, the attention matrix of the current sequence can be obtained after addition operation. Taking into account the new noise brought by the addition operation, we execute the activation function $sigmoid$ on a, which can enhance the weight of the effective value:

$$a = sigmoid(Wa + b),$$

where W is weight and b is bias term. Then we can calculate the attention matrix a and use it to weight the output of the RNN and CNN modules. The specific method is as follows:

$$z = a \cdot c + (1 - a) \cdot r.$$

Through the attention structure, our model can assign different weights to the output of the RNN and CNN modules, and filter the useless information of these two modules at the same time, thereby effectively reducing the impact of redundant context-dependence information on the final result.

3.3.1. Classification layer

After encoding at the parallel layer and the attention structure, the model obtains a sequence of semantic features that contain context-dependence. Considering the over-fitting problem during training, only a single dense layer is used as the final classifier. Finally, through the activation function $softmax$ which computes the probability of a label for each tag, the final prediction is obtained.

4. Experiment and Result

4.1. Corpus

The CANTEMIST corpus consists of 3000 clinical cases, and the professional clinical coding experts annotate these clinical cases in Spanish with eCIE-O-3.1 codes using the BRAT annotation tool. These cases are distributed in plain text in UTF8 encoding, and each clinical case is stored as a file. The corpus is randomly divided into training set, development set and test set. There

Table 1
The number of records and sentences in each data set

	Train set	Dev1 set	Dev2 set	Test set	Total
Number of Records	501	250	250	300	1031
Number of Sentence	18842	9274	8478	10985	47579

Table 2
HyperParameter settings

HyperParameter	Value
character_embedding_size	200
character_CNN_kernel	3
word_embedding_size	200
GRU_units	200
SRU_unit	200
CNN_kernel	3,5,7
batch_size	16
epoch	5
dropout	0.1
learning_rate	0.0001

are two development set in this task, and we use them to adjust the hyperparameters of this model. The test set have 300 clinical records with standard annotations, which can be used to test the performance of the model. This task also introduces a background set without standard annotations to prevent the participating teams from making manual corrections. Finally we count the number of records and sentences to help understand the distribution of the data sets. The results are shown in Table 1.

4.2. Preprocessing and hyperParameter settings

In the preprocessing of this task, we perform basic sentence and word segmentation operations on the original text. Sentences in clinical texts are split by using specific punctuation (such as line breaks or periods) and words are split by using spaces. Then taking into account the character embedding in the model, the original text was not processed with stop words in the preprocessing, and we also remove punctuation marks, special symbols, and single-word sentences.

Finally, we use Keras and Tensorflow as the model backend to implement the model, and train and predict on NVIDIA GeForce GTX 2080Ti. The detailed hyperparameter settings of the experiment are shown in Table 2.

4.3. Comparative experiment and experiment result

To verify the stability of the model and the credibility of the result, we compare our model with the BERT, which is the most popular model for NER. The result shown in Table 3. It shows

Table 3
Results of comparative experiment

model	P	R	F1
Baseline	0.181	0.737	0.291
bert [3]	0.737	0.707	0.722
our model	0.757	0.736	0.746

Table 4
Comparison of different RNN modules and stacking methods

model	P	R	F1
GRU	0.735	0.731	0.733
SRU	0.725	0.716	0.720
LSTM	0.737	0.728	0.732
GRU+LSTM	0.750	0.741	0.745
GRU+SRU	0.757	0.736	0.746

that our model performance on this data sets is better than the BERT, and the F1-score exceeds 2%. Compared with the BERT, our model parameters are less and the training cost is lower. The comparative experiment proves that our model performs well on the tumor named entity recognition in Spanish.

5. Ablation Study

The ablation study illustrates the effectiveness of the RNN and CNN modules in this model.

For the RNN module, we mainly compare the performance of LSTM, GRU and SRU without stacking, and the impact of different stacking methods on model performance. Firstly, we use LSTM, GRU, and SRU as encoders to encode the embedding layer features without stacking. From experiments, we find that GRU and LSTM have similar performance on this task, and both F1-scores are 0.736. Consider the cost of training, GRU can effectively reduce single training time and the required parameter is also less than LSTM. Therefore, we use the GRU as the first encoder of the RNN module. For the second encoder of the model, the module of SRU and LSTM have similar performance when stacking GRU, with F1 scores of 0.746 and 0.745 respectively. Because SRU reduces a lot of parameters and training costs, so combination of SRU and GRU will be a more suitable choice for this task. The RNN module comparative experiment results are shown in Table 4.

For the CNN module, we want to prove whether the existence of the CNN module is necessary. Firstly, we use the RNN module directly stack with the embedding layer, and the F1-score of the model can only achieve 0.727. Then we use the CNN module to replace the embedding layer and find that the F1-score of the model achieve 0.746. This shows that the CNN module is helpful to improve model performance. The experiment results are shown in Table 5.

Table 5
The result of CNN module's impact on model performance

model	P	R	F1
RNN+embedding	0.736	0.718	0.727
RNN+CNN	0.757	0.736	0.746

6. Conclusions

In this paper, we verified the effectiveness of the parallel-attention model and the outstanding performance of RNN on tumor named entity recognition through CANTEMIST-NER task. Compared with BERT, which has higher training costs, over-reliance on corpus, and risk of overfitting, our model is more lightweight. This also shows that RNN is a more suitable choice for NER tasks.

In our model, we use the local dependency extraction ability of CNN to make up for the feature extraction ability of the RNN. At the same time, the long-distance dependence extraction ability of RNN is further strengthened by stacking the GRU and SRU. Finally, by adding a dynamic attention mechanism, our model can filter the redundant features extracted by the CNN and RNN. For the OOV problem, we combine character embedding and word embedding to alleviate this problem. And the comparative experiments and ablation study are also sufficient to show the outstanding performance of the model and the rationality of each module in this model.

For future work, our team hopes to compare the model with more advanced models and verify the effectiveness of the model on more data sets to improve generalization capabilities.

References

[1] 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, in: Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2020), CEUR Workshop Proceedings, 2020.

[2] H. Wei, M. Gao, A. Zhou, F. Chen, W. Qu, C. Wang, M. Lu, Named entity recognition from biomedical texts using a fusion attention-based bilstm-crf, IEEE Access 7 (2019) 73627–73636.

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

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

[5] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (1998) 2278–2324.

[6] T. Makino, Y. Ohta, J. Tsujii, et al., Tuning support vector machines for biomedical named entity recognition, in: Proceedings of the ACL-02 workshop on Natural language processing in the biomedical domain, 2002, pp. 1–8.

[7] B. Settles, Abner: an open source tool for automatically tagging genes, proteins and other entity names in text, Bioinformatics 21 (2005) 3191–3192.

[8] S. K. Sahu, A. Anand, Recurrent neural network models for disease name recognition using domain invariant features, arXiv preprint arXiv:1606.09371 (2016).

[9] L. Luo, Z. Yang, P. Yang, Y. Zhang, L. Wang, H. Lin, J. Wang, An attention-based bilstm-crf approach to document-level chemical named entity recognition, Bioinformatics 34 (2018) 1381–1388.

[10] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, J. Kang, Biobert: a pre-trained biomedical language representation model for biomedical text mining, Bioinformatics 36 (2020) 1234–1240.

[11] F. Soares, M. Villegas, A. Gonzalez-Agirre, M. Krallinger, J. Armengol-Estapé, Medical word embeddings for spanish: Development and evaluation, in: Proceedings of the 2nd Clinical Natural Language Processing Workshop, 2019, pp. 124–133.

[12] A. Joulin, E. Grave, P. Bojanowski, T. Mikolov, Bag of tricks for efficient text classification, arXiv preprint arXiv:1607.01759 (2016).

[13] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (1997) 1735–1780.

[14] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, arXiv preprint arXiv:1406.1078 (2014).

[15] T. Lei, Y. Zhang, Y. Artzi, Training rnns as fast as cnns (2018).

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, pp. 5998–6008. URL: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.