A BERT-Based Framework for Chinese Medical Entity Type Inference

Zhe Wu, Shen Ge, Xian Wu

Tencent Medical AI Lab
`kerrzwu, shenge, kevinxwu@tencent.com`

Abstract. Entity Type Inference (ETI) targets to assigns pre-defined labels to named entities. To encourage the development of Chinese ETI techniques in medical field, the 2020 Chinese Conference on Knowledge Graph and Semantic System organized the COVID-19 Knowledge Graph Entity Type Inference Challenge, in which participates are required to assign one of eight types to each entity given the name as well as related data from web corpus. In this challenge, we propose a BERT based solution which consists of a BERT model as the encoder, a multi-layer perceptron as the classifier and finally a post processor. On the official test set, our approach achieves an F1-score of 0.99525.

Keywords: Entity Type Inference · Chinese Medical Text · BERT.

1 Introduction

Knowledge graph (KG) is a widely adopted form of knowledge representation. In the medical domain, the KG uses nodes to represent medical concepts, like diseases, symptoms, medicines, body parts, etc, and use edges to connect related nodes, like the typical symptoms of a disease, the co-occurring relations between diseases, etc. Such a medical KG, once built, can benefit multiple tasks, like inferring diseases according to given symptoms, identifying medicines with similar functions. Building a KG from scratch consists of multiple tasks, including named entity recognition, relation identification, entity disambiguation. Among which, one critical step is Entity Type Inference (ETI).

For each item identified from unstructured text, ETI assigns one or few predefined type(s) for it. However, limited by the insufficiency of verified dataset and the inconsistency of type assigning criterions, such process is especially challenging. For example, in medical field, since there are so many intricate terms in different branches and specializations, it is quite difficult to build a fully-covered dataset which includes all possible terms. Meanwhile, currently the medical ETI tasks still heavily rely on the doctor's manual labels, which are labor-intensive and therefore could not be widely used.

To encourage the development of Chinese ETI in medical field, the 2020 Chinese Conference on Knowledge Grapha and Semantic System (CCKS-2020) organized the COVID-19 Knowledge Graph Entity Type Inference Challenge. In this challenge, the participates are supposed to assign one of 7 pre-defined types (i.e. virus, bacteria, diseases, drugs, medical specialties, examination subjects and symptoms) to medical related entities and assign a special type named “NoneType” to all other entities.

In this paper, we model the ETI task as a text classification problem. Text classification is a fundamental task in many Natural Language Processing applications, such as search, document classification and recommendation system. The first step of text classification is to convert the target entities to numerical feature vectors. However, different from English where the word boundaries are rather clear, to build feature vectors for Chinese entities is much more difficult since there are no delimiters between the words. To solve such problem, people either choose to build feature vectors for individual characters, or add an extra word segmentation step before classification.

Several methods [10, 3, 11] have been proposed for English text classification task and achieved good performance. Among them, the neural network (NN) based methods have achieved the highest performance and reduced a lot of manual works in feature engineering. Using huge chunk of unlabeled text as the pre-training data, these NN-based methods (e.g., Bidirectional Encoder Representations from Transformers, BERT [2]) could be utilized as the feature extractor (encoder) in many downstream tasks, achieving even better performance with futher parameter fine-tuning by a small chunk of labeled data.

In this paper, we build a framework for Entity Type Inference. This framework consists of an encoder to convert the target entities to numerical feature vectors and a classifier to infer the types based on these feature vectors. For the encoder, to enrich the information, we combine each entity with some related context and then feed the combined results into a BERT [2] based encoder to generate a fixed-size feature representation. All of these context are collected from open web corpus. To address the problem of training data insufficiency, we also use this web corpus to perform data augmentation. For the classifier, we use a multi-layer perceptron (MLP) to predict the probability scores for all types and a post-processor with some heuristic rules to generate the final prediction. On the official test set of CCKS-2020, our system achieve an F1-score of 0.99525.

2 Method

In this section, we first illustrate the structure of our framework. Then, we describe the encoder of our framework which converts the input text to a nunber of numerical feature vectors. Finally, we describe the classifier of our framework which consists of a 2-layer MLP and a post-processor with some heuristic rules.

2.1 Framework

The framework of our method is depicted in Fig.1. Since the input entity is a short phrase or just a single word which lacks of concrete information, we first use it as a query to pull some relevant descriptions about this entity from open web corpus, and then combine them together as the new input. For an input entity with n characters, denoted as $X = (x_1, x_2, ..., x_n)$, we first use X as a query to pull a descriptive context from a pre-built web corpus. Next, we use the obtained context with m characters $T = (t_1, t_2, ..., t_m)$ as the auxiliary information for the entity and combine X and T together to a vector like

$$X_T = ([\text{CLS}], x_1, x_2, ..., x_n, [\text{SEP}], t_1, t_2, ..., t_m, [\text{SEP}]) \tag{1}$$

as the input sentence for the encoder, where [CLS] and [SEP] denote the special tokens in BERT, meaning “class” and “separator”, respectively.

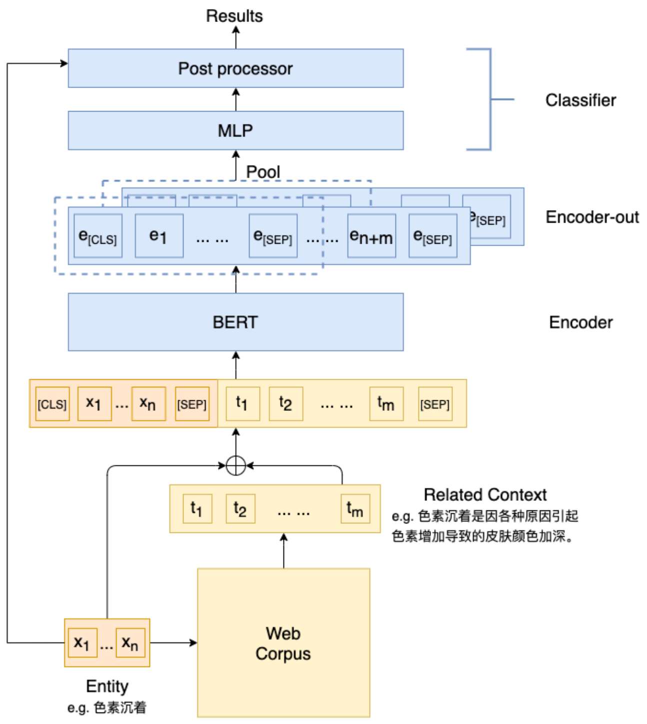

Fig. 1. Our framework for Chinese medical ETI

To avoid the possible errors by the introduction of word segmentation, we use the character-based encoder instead of word-based encoder in our approach, which has also shown better performance in some similar tasks in Chinese [4]. For the input sentence X_T, the encoder first tokenize X_T using a given vocabulary where each character is represented by a unique integer. And then the encoder zero-pads the tokenized sentence to a fix-sized vector. Finally, the encoder converts the obtained fix-sized vector to an array of numerical feature vectors $E = (e_{[\text{CLS}]}, e_1, ..., e_n, e_{[\text{SEP}]}, e_{n+1}, ..., e_{n+m}, e_{[\text{SEP}]})$ containing all semantic information in entity X and its context T.

For the output feature vector of the encoder, we use a 2 layer MLP to generate the probabilities for all pre-defined types. These probabilities would be further adjusted by a post processor to generate the final prediction. The post processor is designed to inject more prior human knowledge into the final prediction including expert knowledge, lexical pattern, web vocabulary and etc.

2.2 BERT Encoder

BERT is a language model based on multi-layer bidirectional Transformer [8] and trained on two different tasks: Masked Language Model (MLM) and Next Sentence Prediction (NSP), which is different from other language models such as ELMo [6] and OpenAI GPT [7]. BERT makes full use of huge chunk amount of unlabeled text data, and the learned feature representation could be further used for other downstream NLP tasks, i.e. Named Entity Recognition, Text Classification and Question Answering. Specifically, by using a pre-trained BERT as the feature extractor, we just need a small-scale of labeled data to fine-tune the model parameters for specific downstream tasks where the achieved performances are often better than methods with hand-crafted word vectors.

There are two implementations with different model sizes in the origin paper of BERT [2]. For our task, we use the larger one which has 24 Transformer blocks, 768-d hidden size and 16 self-attention heads. Considering that the deeper blocks may be more correlated to the pre-train tasks MLM and NSP, we drop the output of the last Transformer block and just take the outputs of the second and third layers to the last blocks.

For each output of these two blocks, we first use a pooling layer to further integrate the information from different channels and than concatenate the two results together as the output of the encoder. For example, for an input vector $X_T = ([\text{CLS}], x_1, x_2, ..., x_n, [\text{SEP}], t_1, t_2, ..., t_m, [\text{SEP}])$ as we mentioned above, the encoder first tokenize and zero-pad it to a vector X_{in} with fixed size $L_{in} = 512$. Next, we use X_{in} as the input of the BERT model and take the output E_1 and E_2 which are the outputs of the second and third to the last Transformer blocks, respectively. After that, we concatenate E_1 and E_2 together using 3 different pooling strategies, i.e. Max, Min and Average, to extract different aspects of features from the word embedding vectors corresponding to entity part $([\text{CLS}], x_1, ..., x_n, [\text{SEP}])$. Finally, we concatenate the outputs of the 3 pooling methods and obtained the final feature representation E_{out}.

2.3 Classifier

For the feature representation generated by the encoder, we use a 2-layer MLP as the classifier to predict the probabilities for all pre-defined types. And these probabilities are further revised by some common heuristic rules based on web dictionary and lexical analysis. The details of the classifier are shown in Fig.2.

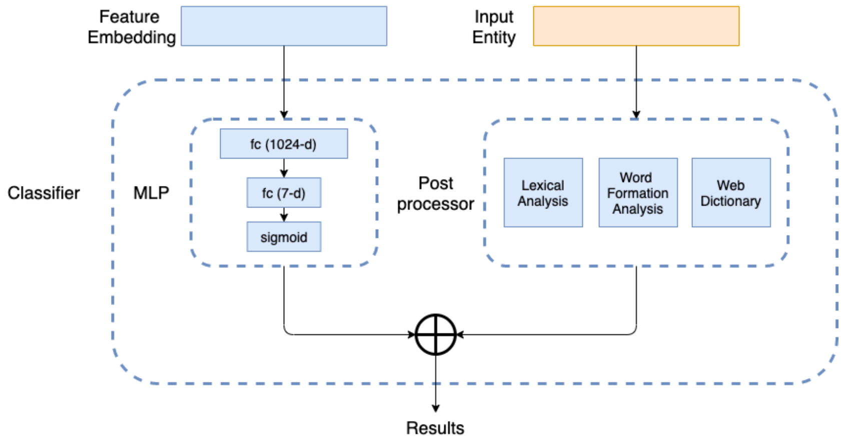

Fig. 2. Details of the classifier

3 Experiments

3.1 Dataset

The dataset of CCKS-2020 ETI Challenge provided by the organizer consists of 3 subsets: training set, validation set and test set, containing 5000, 19999 and 14321 entities respectively, and only the entities in the training set are given their corresponding types, i.e. virus, bacteria, diseases, drug, medical specialties, examination subject, symptoms and NoneType. We noticed that some validation and test entities are already occurred in the training dataset.

Beside these dataset, the organizer also provided a web corpus for target entities. This official web corpus are collected from BaiduPedia, HudongPedia, WikiPedia and A+ MedicalPedia. However, this corpus is difficult to use since it format is hard to parse, and moreover some entities are not included. As a result, we use the entity names as the search terms in some search engines and use the obtained results as the supplementary contexts, making sure that each entity has at least one related description text. We refer this web corpus as the extended web corpus.

In our experiments, we randomly selected 80% entities in the training set to train our encoder and MLP, where the rest of entities are used to tune the hyper-parameters.

3.2 Data Augmentation

Since the provided official training set just contains 5000 entities, it may not be so sufficient to train a model with massive parameters, e.g., BERT-large with over 340M parameters [2]. To avoid overfitting, we use the information in the extended web corpus to augment the dataset. Specifically, for a specific entity X, we assign it with different description texts obtained from the extended web corpus to form different input samples $X_{d_1}, X_{d_2}, ..., X_{d_k}$. At the training stage, these samples would be seen as independent individuals for the loss backward propagation. At the inference stage, all corresponding output probabilities $O_{d_1}, O_{d_2}, ..., O_{d_k}$ which belong to the same entity X would be gathered, and their average value $\overline{O}$ would be used as the output of the entity X. Since we use binary classifiers for each one of 7 pre-defined types, if all probabilistic values for an input entity are less than 0.5, the special NoneType would be assigned to this entity.

3.3 Implementation

In our experiments, we use PyTorch as the backend and implement the Transformers based on the library of HuggingFace [9]. The pre-training method of our BERT model is a RoBERTa [5] model initialized with the weights pre-tranined on Chinese text data [1]. Moreover, we used 2 different ways to fine-tune the encoder and the MLP: using entity name as input, and using entity name with the description from the extended web corpus.

The experimental settings we used for our best encoder and MLP model are described as follows. During the training stage, for the BERT encoder and the MLP, we update the parameters of them with same optimizer but different learning rates. For the BERT encoder which has been pre-trained using a large amount of unlabeled text data, we use a small learning rate of 1e-5 to fine-tune its parameters. On the contrary, we use a relatively higher learning rate of 1e-3 for the MLP which was initialized randomly. Our models are trained on Tesla V100 with an effective batch size of 32. Finally, we use some heuristic rules in the post processor to adjust the ensemble predictions of 14 average voting models as the output.

3.4 Evaluation

The evaluation metric used in this task is F1-score. Denoting two entity-type pair set A and G in which contains all entities and their corresponding type names. The precision, recall and F1-score are calculated as follows.

$$P = \frac{|A| \cap |G|}{|A|} \qquad R = \frac{|A| \cap |G|}{|G|} \qquad F1 = \frac{2PR}{P+R} \tag{2}$$

3.5 Experimental Results

The results of all our experiments on the official validation set and test set are shown in Tab. 1 and Tab. 2. As the results depicted, the methods using the extended web corpus achieve better performance than just using entity names. The main reason is that the entity names are usually short phrases containing rather limited information. Adding related descriptions for them would in fact greatly enrich the context information and make fully use of the representation ability of BERT. In addition, our data augmentation strategy infuse various information for a single entity, which reduces the negative effect brought by the noise in the web corpus.

Table 1. Performance on the official validation set

Method	F1-score
BERT+MLP	0.95735
BERT+MLP+The Extended Web Corpus	0.97570

Table 2. Performance on the official test set

Method	F1-score
BERT+MLP	0.96599
BERT+MLP+The Extended Web Corpus	0.98129
BERT+MLP+The Extended Web Corpus+Post	0.99525

4 Conclusion

In this paper, we present a framework for Chinese entity type inference in medical field. This framework uses a BERT-based encoder to learn Chinese text feature representation and uses the extended web corpus to enrich the context information for entities. The experimental results show the effectiveness of our framework. On the official test set, our method achieved an F1-score of 0.99525.

References

1. Cui, Y., Che, W., Liu, T., Qin, B., Wang, S., Hu, G.: Revisiting pre-trained models for chinese natural language processing. In: Findings of EMNLP. Association for Computational Linguistics (2020)
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
3. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759 (2016)
4. Li, H., Hagiwara, M., Li, Q., Ji, H.: Comparison of the impact of word segmentation on name tagging for chinese and japanese. In: LREC. pp. 2532–2536 (2014)
5. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
6. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. arXiv preprint arXiv:1802.05365 (2018)
7. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding by generative pre-training (2018)
8. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. pp. 5998–6008 (2017)
9. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al.: Huggingface's transformers: State-of-the-art natural language processing. ArXiv pp. arXiv–1910 (2019)

10. Zhang, X., LeCun, Y.: Text understanding from scratch. arXiv preprint arXiv:1502.01710 (2015)
11. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in neural information processing systems. pp. 649–657 (2015)