UNIXLONG at SemEval-2020 Task 6: A Joint Model for Definition Extraction

Shu-Yi Xie , Jian Ma , Hai-Qin Yang, Lian-Xin Jiang , Yang Mo , Jian-Ping Shen
Ping An Life Insurance, Lt
AI Department
ShenZhen, China
{XIESHUYI542,MAJIAN446,YANGHAIQIN260,JIANGLIANXIN769,
MOYANG853,SHENJIANPING324}@pingan.com.cn

Abstract

Definition Extraction is the task to automatically extract terms and their definitions from text. In recent years, it attracts wide interest from NLP researchers. This paper describes the unixlong team's system for the SemEval 2020 task6: DeftEval: Extracting term-definition pairs in free text. The goal of this task is to extract definition, word level BIO tags and relations. This task is challenging due to the free style of the text, especially the definitions of the terms range across several sentences and lack explicit verb phrases. We propose a joint model to train the tasks of definition extraction and the word level BIO tagging simultaneously. We design a creative format input of BERT to capture the location information between entity and its definition. Then we adjust the result of BERT with some rules. Finally, we apply TAG_ID, ROOT_ID, BIO tag to predict the relation and achieve macro-averaged F1 score 1.0 which rank first on the official test set in the relation extraction subtask.

1 Introduction

Definition extraction is a hot research topic in natural language processing (NLP) that aims to identify terms and their corresponding definitions in unstructured text. However, it is a challenging problem to extract term-definition pairs from free and semi-structured text, especially those whose term-definition pair span crosses a sentence boundary and those lacking explicit definition verb phrases.

SemEval is a yearly organized NLP competition. The SemEval 2020 - Task 6, "DeftEval: Extracting term-definition pairs in free text", aims to extract definition from the DEFT (Sasha et al., 2019) corpus, which contains roughly 7,000 sets of 3-sentence groupings extracted from textbooks of various topics from cnx.org. Each sentence set reflects a context window around which the author of the original text marked a bolded word to indicate a key term.

DeftEval consists of three tasks:

1) Subtask 1: Sentence Classification. Given a sentence, it is to classify whether the sentence contains a definition or not. This is a traditional definition extraction task.

2) Subtask 2: Sequence Labeling. It is to label each token with the BIO tag according to the corpus' tag specification.

3) Subtask 3: Relation Classification. Given the tagging results, it is to label the relations between each tag according to the corpus' relation specification.

This paper describes our system and participation in all subtasks of SemEval 2020 task6.

The challenges of DeftEval include:

1) The DEFT corpus contains various topics, such as biology, history, physics, psychology, economics, sociology, and government, etc. It is challenging and requires deep background knowledge to understand the professional definition from these topics.

2) The sentences in the DEFT corpus are extremely complicated. A term and its definition may come from several sentences. A model needs to capture long distance term relations.

To overcome these challenges, we incorporate several key technologies:

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 730–736
Barcelona, Spain (Online), December 12, 2020.

1) proposing a multi-task learning model to simultaneously solve subtask1 and subtask2, where a shared layer is designed and a multi-task loss function is adopted to train the model to capture common information.

2) applying the data augmentation technique to alleviate the imbalance class problem while utilizing rank average ensemble learning method on several state-of-the-art pre-trained models, including BERT (Devlin et al., 2018), Roberta (Liu et al., 2019), and ALBERT (Lan et al., 2019).

3) applying CRF to learn some constraints of BIO annotation.

4) inserting a [#] token around term and definition to emphasize their position importance in BERT and designing some rules to handle special cases in the relation classification subtask.

The rest of this paper is organized as follows: In section 2, we briefly introduce related work on definition extraction. In section 3 we detail our proposed system. In section 4, we analyze the results of our models. Finally, we conclude our work in section 5.

2 Related Work

One stream of work on definition extraction are the rule-based approaches. Some linguistic rules and patterns are well designed (Fahmi et al., 2006). Most patterns are relied on the fact that the definition can be captured by the common "definitor" verb phrases such as "means", "refers to", and "is". Although rule-based approaches exhibit high precision if matching the patterns, they suffer from the low recall issue. Another stream of work on definition extraction are the feature engineering approaches (Westerhout et al., 2009). They try to address the low recall issue by training the machine learning models on well-designed hand-made features. However, they usually yield poor performance when transferring to new domains.

Recently, pre-trained language models, such as BERT, XLNET (Yang et al., 2019), Roberta, have demonstrated their effectiveness in tackling NLP tasks. For example, BERT, Bidirectional Encoder Representations from Transformers, redefines the performance of 11 most common NLP tasks after fine-tuning. XLNET outperforms BERT on 20 tasks via the idea of auto-regressive models and bi-directional context modeling. Roberta refreshes many leaderboards through robustly optimizing BERT via the self-supervised learning scheme. The successfulness of pre-trained models on typical NLP tasks, such as text classification and NER, motivates us to apply them in tackling the tasks in the competition, including definition extraction, entity annotation, and relation extraction.

Multi-Task Learning (MTL) is an effective learning mechanism to boost the performance of each individual task by simultaneously learning several similar tasks together to leverage the knowledge among the tasks. MTL has been demonstrated its effectiveness on many NLP tasks (Zhang and Yang, 2017). Liu et al., 2019 presents a Multi-Task Deep Neural Network (MT-DNN) for learning representations across multiple natural language understanding (NLU) tasks and obtains new state-of-the-art results on ten NLU tasks. Veyseh et al. 2019 proposes a joint model for definition extraction to exploit the global structures of the input sentences as well as the semantic consistencies between the terms and the definitions. Wu and He, 2019 develop an approach for relation classification by enriching the pre-trained BERT model with entity information. In this competition, since the subtasks are interrelated, e.g., if the word level BIO tag predicted by subtask2 contains B-Definition, then the classification tag for subtask1 is 1. It naturally motivates us to explore a joint model to solve them in a whole.

3 System Overview

We elaborate the task and our system in this section.

3.1 Task and Data Description

The dataset is split into two subsets, train and dev, and written in a CONLL-like tab-deliniated format. Each line represents a token and its features. A single blank line indicates a sentence break; two blank lines indicates a new 3-sentence context window. All context windows begin with a sentence id followed by a period. These are treated as tokens in the data. Each token is represented by the following features: [TOKEN] [SOURCE] [START_CHAR] [END_CHAR] [TAG] [TAG_ID] [ROOT_ID] [RELATION]. Where: SOURCE is the source.txt file of the excerpt. START_CHAR/END_CHAR are char index boundaries of the token. TAG is the label of the token (O if not a B-[TAG] or I-[TAG]). TAG_ID is the ID associated with this TAG (0 if none). ROOT_ID is the ID associated with the root of this relation (-1 if no relation/O tag, 0 if root, and TAG_ID of root if not the root). RELATION is the relation tag of the token (0 if none).

In the definition extraction subtask, we predict the binary tag of each sentence. The tag is 1 if the sentence contains a definition and 0 for others. In the competition, the score is measured by the F1 score on the positive class. In the sequence labeling subtask, the evaluated TAG classes include Term, Alias-Term, Referential-Term, Definition, Referential-Definition, and Qualifier. In the relation extraction subtask, the evaluated relations include Direct-defines, Indirect-defines, Refers-to, AKA, and Qualifies. The evaluation metric is the macro-averaged F1 score.

3.2 Data Augmentation

The dataset provided in this competition is unbalanced with respect to tag schema. As reported in Table 1, some tags, e.g., Qualifier, Referential-Definition, contain only few training samples. To alleviate the data imbalance problem, we apply synonym substitution schemes based on word vectors, singular plural, and pronoun to increase the training data. We only replace some adjectives, verbs in the sentence to keep the number of words unchanged. Therefore, the tag corresponding to each word is also unchanged. Our evaluation shows that our proposal performs better on the minority classes after applying data augmentation.

tag	count	percentage
O	73389	40.51%
B-Definition	5852	3.23%
I-Definition	83305	45.98%
B-Term	6291	3.47%
I-Term	9051	5.00%
B-Qualifier	143	0.08%
I-Qualifier	927	0.51%
B-Referential-Definition	157	0.09%
I-Referential-Definition	328	0.18%
...		

Table 1: Tag Statistics in Training Data

3.3 Joint Model

We propose a joint model to solve the sentence definition extraction task (subtask1) and sequence labeling task (subtask2) simultaneously. These two tasks share related information, which can be complement each other.

Figure 1 outlines our proposed joint model framework. In the input layer, the input of the definition extraction task includes the definition label and the sequence labeling tags. In a similar way, the input of the sequence labeling task contains its own BIO tags and the definition label. Then, we encode the input to obtain the embedding vector corresponding to each token from the multiple transformer blocks. The shared layers can be from one of the pre-trained models, such as BERT, XLNET, ALBERT, Roberta and others. Finally, we send the output vectors from shared layers to the task specific network layer.

In the definition extraction task, we add a MLP layer after the shared layer model's output to predict the binary labels. We add a scaling factor on cross entropy as definition extraction task's loss function.

$$L_{cla} = -\frac{1}{r^2}\sum_i (y_i \log(y_i^{'}) + (1-y_i)\log(1-y_i^{'})), \tag{1}$$

where y_i is the gold label, $y_i^{'}$ is the predicted label, r is the scaling factor.

In sequence labeling task, we apply the shared layer to generate the probability vector of each word's BIO tags. Then we feed the probability vectors as the input of CRF model. Before our training, we

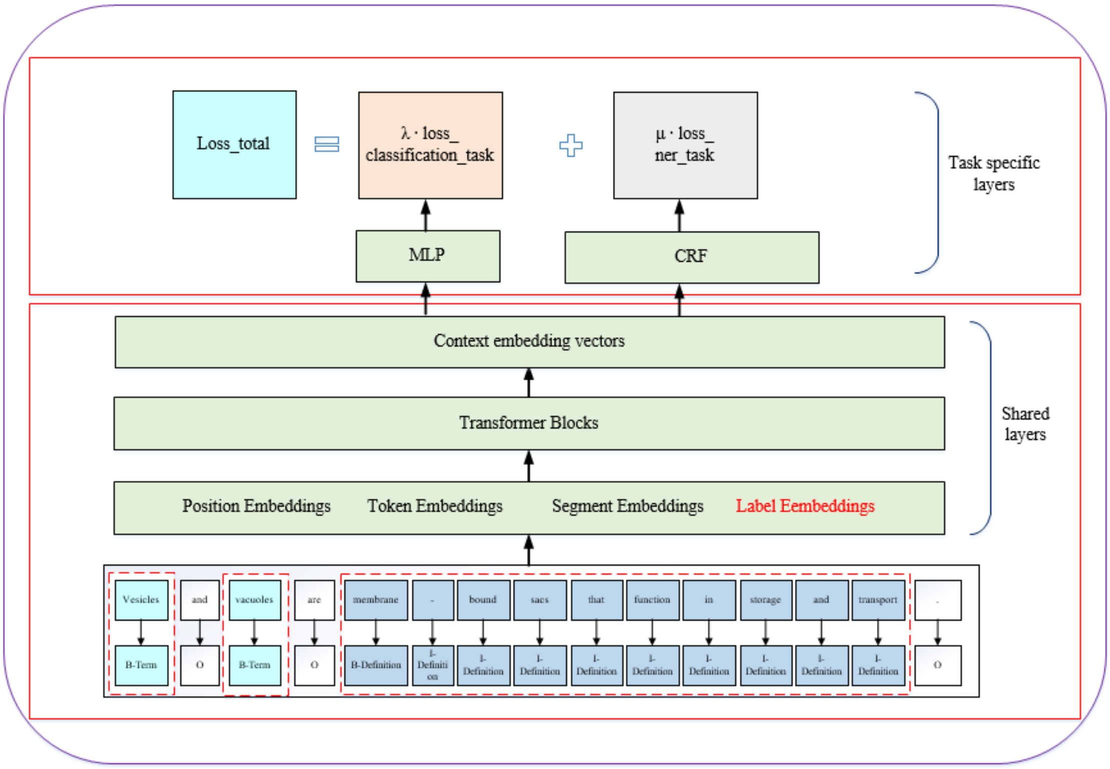

Figure 1: Joint Network for Definition and Entity Extraction

	B-Term	I-Term	B-Definition	I-Definition	O	...
B-Term	0.01	0.7	0.02	0	0.2	...
I-Term	0.02	0.5	0.03	0	0.2	...
B-Definition	0.01	0	0	0.7	0.25	...
I-Definition	0.02	0	0.01	0.6	0.3	...
...						...

Table 2: CRF Transition Matrix

initialize the CRF transition matrix according to the training data's BIO tag probability statistics as illustrated in table 2.

The transition matrix of the CRF model can be automatically learned during the training process. The CRF layer can learn many constraints, such as the first word of one sentence should be B-TAG or O TAG but can't be I-TAG, I-TAG can't immediately follow O-TAG and in "B-tag1 I-tag2 I-tag3…", where tag1, tag2, tag3 should have the same BIO tag.

The goal of this sequence labeling task is to assign BIO tag to each word. We define the prediction score to be

$$S(X, y_{tag}) = \sum_{i=0}^{n} P_{i,y_i} + \sum_{i=0}^{n} A_{y_{i-1},y_i} \tag{2}$$

where $X = (x_1, x_2, ..., x_n)$ is the output vector from the shared layer, $y_{tag} = (y_1, y_2, ..., y_n)$ is the predicted tag. P_{i,y_i} corresponds to the score of assigning the tag y_i of the i^{th} word in a sentence. The matrix A records the transition scores, e.g., A_{y_{i-1},y_i} represents the score of the transition from the tag y_{i-1} to the tag y_i in the CRF model. We normalize the prediction probability by

$$p(y_{tag}|X) = \frac{e^{S(X,y)}}{\sum_{y' \in Y_X} e^{S(X,y')}} \tag{3}$$

The loss function of the sequence labeling task is then defined as follows:

$$L_{tag} \quad = -\log(p(y_{tag}|X)) = \log(\sum_{y' \in Y_X} e^{S(X,y')}) - S(X, y_{tag}) \tag{4}$$

The key point is the design of loss function of the joint model. Traditional method usually sums different tasks' loss function directly. In this work, we consider the uncertainty of coefficient weight of the classification task and the sequence labeling task and define the total loss function as follows:

$$L_{total} = \frac{1}{\sigma^2} L_{cla} + \frac{1}{\tau^2} L_{tag} \tag{5}$$

where σ, τ are the trade-off factors to be tuned from the datasets.

3.4 relation extraction

In the relation prediction task (subtask3), we observe that one sentence may contain different BIO tag annotation for each word, which enhances the complication of the problem. To keep the solution simple, we apply BERT as our base model and add some rules to handle special cases to adjust the final results. Figure 2 shows the input format of our approach. The [CLS] token is added at the beginning of each sentence context window and a special token [#] is inserted at the beginning and the end of each BIO tag annotation sequence to make the BERT model capture the location information of BIO tags.

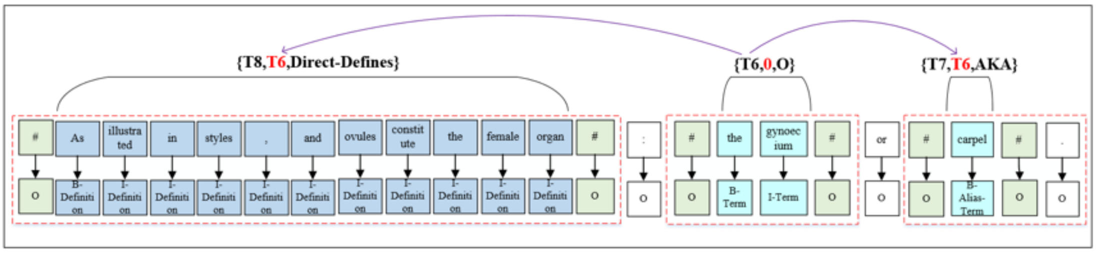

Figure 2: Input Format of BERT Model

For example, in Figure 2, the input is "[CLS] Sentence [#] BIO tag sequence [#] Sentence [#] BIO tag sequence [#] Sentence". There are 3 BIO tag sequences t1, t2 and t3. We define the representation of t1, t2 and t3 as follows:

$$H_1^{'} = W_1[tanh(\frac{1}{a-b+1}\sum_{t=b}^{a} H_t)] + b_1, \tag{6}$$

$$H_2^{'} = W_2[tanh(\frac{1}{i-j+1}\sum_{t=j}^{i} H_t)] + b_2, \tag{7}$$

$$H_3^{'} = W_3[tanh(\frac{1}{m-n+1}\sum_{t=n}^{m} H_t)] + b_3. \tag{8}$$

where vectors H_b to H_a are the final hidden state vectors from BERT for sequence t1, vectors H_j to H_i are the final hidden state vectors from BERT for sequence t2, and vectors H_n to H_m are the final hidden state vectors from BERT for sequence t3. We just add these vectors and average them. After that, we send the average vector to tanh activation function. We share the weight and bias parameters and these parameters are automatically learned during training process. That is, $W_1 = W_2 = W_3$ and $b_1 = b_2 = b_3$. Then, we obtain t1, t2 and t3's representation vectors and can calculate each other's cosine distance to test if it is bigger than a certain threshold. For those sequence pairs whose relation output is 1, we predict their ROOT_ID from the BIO tags, like TERM's ROOT_ID is 0, DEFINITION's ROOT_ID is the TERM in the sequence pairs' TAG_ID. In this example, t1 and t2 has relation, t1's ROOT_ID is T6 because t1 is a definition, t2 is the term to be defined, t2's TAG_ID is t1's ROOT_ID. Since the sentence context matters, one term's definition may not appear in the same sentence with this term, we take the sentence context window as our model's input.

There are some special cases that one sentence contains different BIO tags. Therefore, we add a certain strategy to adjust the model's prediction. For example, if there are Referential-Definition and Term tags in the sentence, and there is no referenced Definition tag appeared, the ROOT_ID of Referential-Definition is the TAG_ID of the nearest Term in the sentence. If there is a Referential-Term tag in the sentence, and there is a Term tag in front of this sentence, the ROOT_ID of Referential-Term is the TAG_ID of the previous Term nearest to it (Term usually appears in the previous sentence of Referential-Term). If the term and its definition tag do not exist in the same sentence, and the term tag exists in the previous sentence or the next sentence, the ROOT_ID of the definition in the sentence is the TAG_ID of the nearest (we calculate the distance from START_CHAR and END_CHAR information of the word) term in the previous sentence or the following sentence.

After we predict the ROOT_ID, we can decide the relation tag based on TAG_ID, ROOT_ID and BIO tag by the following rules:

1 If the BIO tag is Qualifier, then the relation tag is Supplements.

2 If the BIO tag is Alias-Term, then the relation tag is AKA.

3 If the BIO tag is Definition and its ROOT_ID's BIO tag is Referential-Term, the relation tag is Indirect-Defines; if its ROOT_ID's BIO tag is Term, then its relation tag is Direct-Defines; but if its ROOT_ID is 0, the relation tag should be 0.

4 If the BIO tag is Referential-Definition, if its ROOT_ID's BIO tag is Term, then its relation tag is Indirect-Defines; if its ROOT_ID's BIO tag is Definition, its relation tag is Refers-To.

5 If the BIO tag is Referential-Term and its ROOT_ID's BIO tag is Term, then its relation tag is Refers-To.

4 Experiment

We detail our experimental setup and present the results in this section. We first split the training dataset into 5 groups and set the dev dataset as our 6^{th} group. For each unique group, we take the group as a hold out or local test dataset and set the remaining groups as the training dataset. We then fit our model on the training set and evaluate it on the local test set. We choose different base models, such as BERT, XLNET, ALBERT, Roberta, XLMROBERTA. For different base models, we finetune the hyperparameters to fit a better result in the local test dataset. For example, in Roberta, the learning rate, the batch size, the training epochs are set to 3e-5, 32, and 5, respectively. Each basic model predicts 5 results on the 5 cross-validation dataset. And finally the 5 basic models get 25 results. For the joint model, we try to adjust the corresponding parameters to see the results in local test dataset.

subtask1 loss weight	0.74	0.85	0.82	0.83
subtask2 loss weight	0.26	0.15	0.18	0.17
subtask1 learning rate	5e-5	2e-5	1e-5	1e-5
subtask2 learning rate	5e-5	2e-5	2e-5	1e-5
Train subtask1 first	√	√		
Train subtask2 first			√	√
Warmup	0.1	0.15	0.15	0.15
subtask2 F1	0.650	0.662	0.669	0.687
subtask1 F1	0.765	0.781	0.790	0.800

Table 3: local results for joint model by choosing different training parameters

We conduct various experiment on choosing different dataset, base model and parameters and ensemble the 25 results by rank average method. Finally, we achieve score 0.8077 and 0.6869 on subtask1 and subtask2 official test dataset, respectively. For the relation prediction task, we predict the ROOT_ID with BERT and adjust some special cases with carefully designed rules. After combining BIO tag with TAG_ID and ROOT_ID, we decide the relation tag. Our best result achieves 1.0 F1 score, ranking the first in this subtask.

5 Conclusion

Definition Extraction is a challenging NLP task to extract definition from free text. In this work, we present our implementation to solve all subtasks of SemEval task6. We propose a joint model to solve definition extraction and sequence labeling subtasks simultaneously. We apply BERT to predict the relation classification task and optimize the model result with some man-made rules. In the future, we plan to explore end-to-end model for the relation extraction subtask. In addition, we can keep improving the joint model's prediction accuracy for subtask1 and subtask2, especially increasing the accuracy of those tags with few training samples.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. *Bert: Pre-training of deep bidirectional transformers for language understanding.* arXiv preprint arXiv:1810.04805.

Westerhout, E. 2009. *Definition extraction using linguistic and structural features.* In Proceedings of the 1st Workshop on Definition Extraction.

Yu Zhang and Qiang Yang. 2017. *A survey on multi-task learning.* arXiv preprint arXiv:1707.08114.

Fahmi, I., and Bouma, G. 2006. *Learning to identify definitions using syntactic features.* In Proceedings of the Workshop on Learning Structured Information in Natural Language Applications.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, Radu Soricut. 2019. *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations.* ICLR, 2019

Xiaodong Liu, Pengcheng He, Weizhu Chen, Jianfeng Gao, 2019. *Multi-Task Deep Neural Networks for Natural Language Understanding.* ACL, 2019

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. *RoBERTa: A robustly optimized BERT pretraining approach.* arXiv preprint arXiv:1907.11692, 2019.

Sasha Spala, Nicholas A Miller, Yiming Yang, Franck Dernoncourt, Carl Dockhorn. 2019. *DEFT: A corpus for definition extraction in free- and semi-structured text.* Proceedings of the 13th Linguistic Annotation Workshop, pages 124–131 2019 Association for Computational Linguistics

Spala, Sasha and Miller, Nicholas and Dernoncourt, Franck and Dockhorn, Carl. 2020. *SemEval-2020 Task 6: Definition extraction from free text with the DEFT corpus.* Proceedings of the 14th International Workshop on Semantic Evaluation, 2020

Amir Pouran Ben Veyseh, Franck Dernoncourt, Dejing Dou, Thien Huu Nguyen. 2019. *A Joint Model for Definition Extraction with Syntactic Connection and Semantic Consistency.* arXiv:1911.01678, 2019

Shanchan Wu, Yifan He. 2019. *Enriching Pre-trained Language Model with Entity Information for Relation Classification.* Proceedings of the 28th ACM International Conference, arXiv:1905.08284, 2019

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. *XLNet: Generalized autoregressive pretraining for language understanding.* In NeurIPS, pages 5754–5764, 2019.

The competition details in codalab https://competitions.codalab.org/competitions /22759#learn_the_details-evaluation