A span-based model jointly optimized on named entity recognition and relation extraction

Yifei Zhou

Jun 26 2021

1 Overview

In this paper, we implement a span-based model jointly optimized on two tasks for ACE05: named entity recognition and relation extraction. Different from normal approaches to tackle named entity recognition as a sequence tagging, we implement a novel span-based model with bert as the sentence encoder because we suspect that a span-based model can achieve better performances for nested entities in ACE05. A span-based model enumerates all possible spans for entities up a certain length to determine whether they are entities or not. For those spans chosen to be entities, they are passed to the relation classification procedure and the loss is given after the relation classification.

2 Data preprocessing

For data preprocessing, we follow the standard convention to break a document into sentences because relations rarely exist across different sentences. We tokenize each sentence by words and use those word tokens as possible boundaries of entity spans and relation spans.

3 Task definition

We define the task as follows: given an input sentence X, let $x_1, x_2, ..., x_n$ be the sequence of input tokens of the sentence of length n. Let $S = s_1, s_2, ..., s_m$ be all possible spans from the sentence X, where $s_t = (x_i, x_j), i \leq j$. In this task, we restrict the largest span length to $L = 8$.

3.1 Named Entity Recognition

We define E to be the set of all predefined entity types for spans and ϵ to be the special notation for "Nontype", which indicates that this span is not an entity. The task of named entity recognition is to assign an label $y_e(s_i) \in E \cup \epsilon$ for each $s_i \in S$.

3.2 Relation Extraction

We define R to be the set of all predefined relation types for relation spans (s_i, s_j) where s_i, s_j are entities, and ϵ to be the special notation for "Nontype" which indicates that this span tuple does not involve any relation. Give the set S' that contains all the spans with entity labels in E labeled in the NER period, the goal of relation extraction is to assign an label $y_r(s_i, s_j), \forall s_i, s_j \in S'$.

4 Model Design

4.1 Token encoding

Inspired by the recent success of pretraining language model BERT(bert-base-cased) in Natural Language Processing(NLP), we apply fixed pretrained model bert-base-cased to get the contextualized word embeddings for each token. A task-specific bidirectional LSTM is added on top of the results of fixed bert embeddings to be optimized along with other task-specific multi-layer perceptron(MLP).

4.2 Span representation

We concatenate the bi-LSTM outputs for the start token and the end token to be the representation for each span. The start and end token position is also added to the end of each span representation.

4.3 Named Entity Recognition

We pass all the span representations from the sentence into a feed forward neural network with one hidden layer. The results of the last output layer are used for entity prediction.

4.4 Relation Extraction

For those spans predicted as entities, their pair combinations are enumerated in the relation extraction stage. As aggregate representations for the span pair, we concatenate the span representation for both spans and their outputs of the last layer MLP in the named entity recognition stage because we suspect that the entity type of both spans can help to find the relation type for the pair. For each pair, we pass its pair representation to a MLP to classify its relation types.

4.5 Loss Function

We define the loss function to be a linear combination of the cross entropy loss from two stages. So that:
$L = \lambda_e \sum_{s_i \in S} logP_e(e_i^*|s_i) + \lambda_r \sum_{s_i,s_j \in S'} logP_r(r_{i,j}^*|s_i, s_j)$
Where λ_e and λ_r are the weights for entity loss and relation loss subject to hyperparameter tuning.

5 Training details

5.1 Warm-up training

In the early epochs of training, the model can hardly pick any entities from the span set, so we would like to set our training focus on the Named Entity Recognition task. In the first 10 epochs of training, we set the $\lambda_r = 0$ and $\lambda_e = 1$ for warming up, after that we set $\lambda_e = \lambda_r = 1$ for joint optimization for relation extraction and entity recognition.

5.2 Implementation details

We split training and development set from the training folder randomly by a ratio of 9:1. We trained on the training data and tune the hyperparameters on the developing data. We choose the maximum span length to be 8. We use a bidirectional LSTM with hidden dimension 256. All of the MLPs are with one hidden layer of dimension 100. The dropout rate for both the hidden layer and the input layers are 0.4. We use an Adam optimizer with learning rate 1e-4 and weight decay 1e-4.

6 Testing results

Our model achieves the following precision, recall and f-1 score on all entity types and relation types.

6.1 Entity types

Entity Type	Precision	Recall	F1
FAC	54.79	35.08	42.78
GPE	79.81	65.68	72.06
LOC	54.55	21.95	31.30
ORG	69.77	54.37	61.15
PER	80.37	84.15	82.21
VEH	69.23	36.73	48.00
WEA	57.14	51.28	54.05

6.2 Relation types

Relation Type	Precision	Recall	F1
ART	38.09	17.39	23.88
GEN-AFF	42.85	14.11	21.24
ORG-AFF	62.77	28.09	38.82
PART-WHOLE	50.00	15.68	23.88
PER-SOC	64.58	46.97	54.39
PHYS	40.00	20.90	27.46

7 Conclusion

This paper implements a simple span-based model jointly optimized for named entity recognition and relation extraction, and somewhat satisfactory results are achieved. Span-based models can avoid the issue of nested entities typical in ACE2005 but suffer from the imbalance of training data. Most of the enumerated spans are meaningless and only a small proportion of them are entities with some types. This data imbalance issue is fatal to span-based models. Joint optimization is also implemented so that relations are extracted in one step so that error propagation is avoided. Still, this model is far from perfect, more tricks and tuning can be done to boost the testing result.