

Code and Named Entity Recognition on StackOverflow

Tabassum et al., 2020, <https://arxiv.org/abs/2005.01634>

Team members: Changheng Liou, Zhiying Cui



Named entity recognition on text with codes

- 8 code entities: CLASS, VARIABLE, DATATYPE, FUNCTION...
- 12 natural language entities: APPLICATION, UI ELEMENT, ALGORITHM, OS, DEVICE...

Example:



Not an easy task

- Ambiguous word: **list** can be variable name and data structure

- Annotated 1237 threads on StackOverflow
- Customized tokenizer
 - `std::vector<int>` -> ['std', ':', ':', 'vector', '<', 'int', '>']
- **Code Recognizer**
 - A token is a code or not regardless of context
- **Entity Segmenter**
 - A token is a given name entity in the given sentence
- **Word Representation (BERT[1])**

SoftNER Model Overview

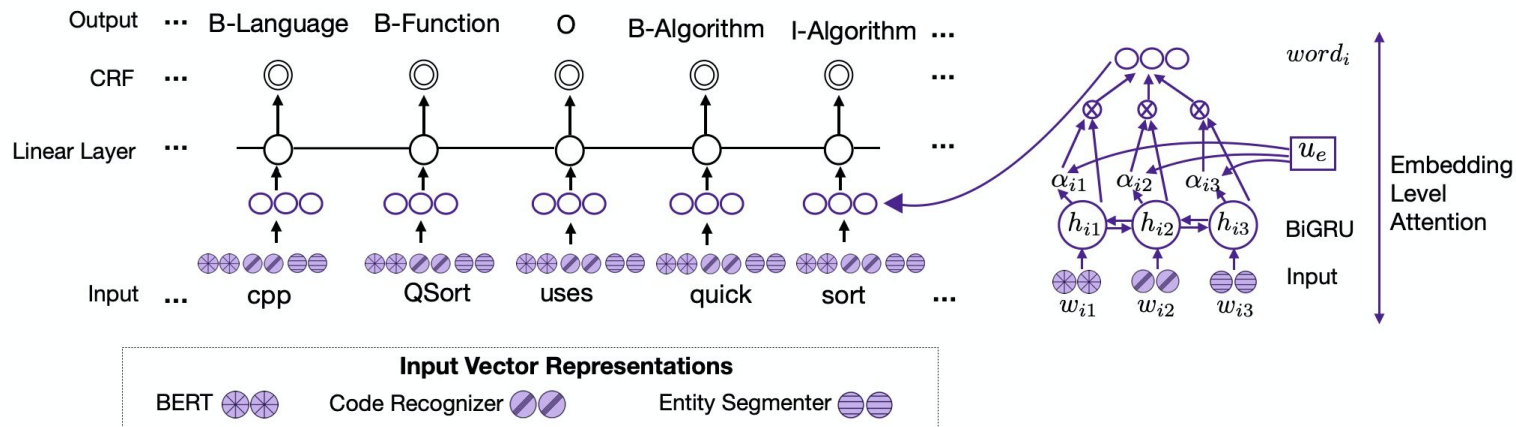


Fig 1. SoftNER Model [2]

- Vector representation: BERT + Code recognizer + Entity Segmenter
- Attention network

Train Code Recognizer

Parameters

- Learning rate: 0.0015
- Epochs: 70
- Word vector dimension: 300
- Hidden layer dimension: 30

Process

- Get features from Gigaword corpus and Stack Overflow archives
- Transform each n-gram probability into a k-dimensional vector using Gaussian binning
- Concatenate the output with pre-trained FastText[3] character-level embeddings

----- test -----					
P:	79.0875	R: 78.7879	F: 78.9374		
		precision	recall	f1-score	support
	0	0.92	0.93	0.92	736
	1	0.79	0.79	0.79	264
accuracy				0.89	1000
macro avg		0.86	0.86	0.86	1000
weighted avg		0.89	0.89	0.89	1000

Our training results

	P	R	F ₁
Token Frequency	33.33	2.25	4.22
Most Frequent Label	82.21	58.59	68.42
Our Code Recognition Model	78.43	83.33	80.80
- Character ngram LMs	64.13	84.51	72.90
- Word ngram LMs	67.98	72.96	70.38
- FastText Embeddings	76.12	81.69	78.81

Table 5: Evaluation results and feature ablation of our code recognition model on SOLEXICON *test* set of 1000 manually labeled unique tokens, which are sampled from the *train* set of StackOverflow NER corpus.

Training results in paper[2]

Fig.2 Train Code Recognizer

Evaluation

Train SoftNER Model

- Input: vectors extracted from Code Recognizer and Entity Segmenter
- 1 epoch costs more than 4 hours with a CPU (100 epochs)

	P	R	F ₁
Test set			
Feature-based CRF	71.77	39.70	51.12
BiLSTM-CRF (ELMoVerflow)	73.03	64.82	68.68
Attentive BiLSTM-CRF (ELMoVerflow)	78.22	78.59	78.41
Fine-tuned BERT	77.02	43.92	57.34
Fine-tuned BERTOverflow	68.77	67.47	68.12
SoftNER (BERTOverflow)	78.42	79.79	79.10
Dev set			
Feature-based CRF	66.85	46.19	54.64
BiLSTM-CRF (ELMoVerflow)	74.44	68.71	71.46
Attentive BiLSTM-CRF (ELMoVerflow)	79.43	80.00	79.72
Fine-tuned BERT	79.57	46.42	58.64
Fine-tuned BERTOverflow	72.11	70.51	71.30
SoftNER (BERTOverflow)	78.81	81.72	80.24

Table 2: Evaluation on the dev and test sets of the StackOverflow NER corpus. Our SoftNER model outperforms the existing approaches.

Training results in paper[2]

```
If 0 If 0
I 0 I 0
would 0 would 0
have 0 have 0
2 0 2 0
tables B-Data_Structure tables 0

CODE_BLOCK B-Code_Block CODE_BLOCK B-Code_Block
: I-Code_Block : I-Code_Block
Q_4780 I-Code_Block Q_4780 I-Code_Block
( I-Code_Block ( I-Code_Block
code I-Code_Block code I-Code_Block
omitted I-Code_Block omitted I-Code_Block
for I-Code_Block for I-Code_Block
annotation I-Code_Block annotation I-Code_Block
) I-Code_Block ) I-Code_Block
```

Part of training dataset

```
If 0
I 0
would 0
have 0
2 0
tables 0
How 0
do 0
get 0
this 0
result 0
The 0
following 0
query 0
needs 0
```

Extracted from Code Recognizer

```
If 0 0
I 0 0
would 0 0
have 0 0
2 0 0
tables Name Name

How 0 0
do 0 0
I 0 0
get 0 0
this 0 0
result 0 0

The 0 0
following 0 0
query 0 0
needs 0 0
```

Extracted from Entity Segmenter

```
test: epoch: 1 P: 70.58 R: 70.22 F1: 70.4
test: epoch: 2 P: 74.94 R: 74.81 F1: 74.87
test: epoch: 3 P: 73.19 R: 72.73 F1: 72.96
```

Running log of prediction on test dataset in each epoch

Fig.3 Training SoftNER Model

Extend the Work - Evaluate the model's performance on Leetcode

We have data, we have the models, we can try:

- Crawl the text dataset from Leetcode discussion
- Use Code Recognizer and Entity Segmenter to extract the features
- Predict the named entities with SoftNER

```
(py36) root@buddy:~/Project# cat leetcode-discuss.txt
https://leetcode.com/problems/meeting-rooms-ii/discuss/67855/Explanation-of-%22Super
r-Easy-Java-Solution-Beats-98.8%22-from-%40pinkfloyda
Explanation of \"Super Easy Java Solution Beats 98.8%\" from @pinkfloyda
The solution is proposed by @pinkfloyda at [\"Super Easy Java Solution Beats 98.8%\"
\"] [1], which is amazing. Here I would like to explain why it works a little bit.
The code from @pinkfloyda: ``` public class Solution { public int minM
eetingRooms(Interval[] intervals) { int[] starts = new int[intervals.le
ngth]; int[] ends = new int[intervals.length]; for(int i=0;
i<intervals.length; i++) { starts[i] = intervals[i].start;
ends[i] = intervals[i].end; } Arrays.sort(starts);
Arrays.sort(ends); int rooms = 0; int endsItr
= 0; for(int i=0; i<starts.length; i++) { if(starts[i]<
ends[endsItr]) rooms++; else endsItr++; } return rooms; } }``` To underst
and why it works, first let's define two events: Meeting Starts Meeting Ends Next
, we acknowledge three facts: The numbers of the intervals give chronological order
s When an ending event occurs, there must be a starting event has happened before t
hat, where \"happen before\" is defined by the chronological orders given by the in
tervals Meetings that started which haven't ended yet have to be put into differen
t meeting rooms, and the number of rooms needed is the number of such meetings So,
what this algorithm works as follows: for example, we have meetings that span alo
ng time as follows: |____| |____| |____| |____|
| Then, the start time array and end time array after sorting appear like foll
ows: | | | | | Initially, 'endsItr' points to the first
end event, and we move 'i' which is the start event pointer. As we examine the star
```

Fig.4 Text data in Leetcode discussion

Conclusions

- Understand the architecture of SoftNER model
- Trained the two models

Challenges and Solutions

- Bugs in source codes, such as encoding problem, commented or deleted code blocks
 - Debug codes
- Some files are missing, such as fasttext file, word frequency file, Entity Segmenter model
 - Contact the author

Remaining problems and Future work

- Not fully functional: Entity Segmenter
- Add or change embeddings to see how it works

Thank you!

Changheng Liou

Email: cl5533@nyu.edu

Zhiying Cui

Email: zc2191@nyu.edu