Assignment-3 sequence tagger

Nidhi Kumari (15127)

Abstract

The goal of the assignment is to build an NER system for diseases and treatments. The input of the code will be a set of tokenized sentences and the output will be a label for each token in the sentence. Labels can be D, T or O signifying disease, treatment or other

1 Introduction

The task in NER is to find the entity-type of words. Entities can, for example, be locations, time expressions or names. Given an input sentence $x = \{w1, w2, \ldots, wn\}$, we output a tag for each of the words in the sentence $y = \{y1, y2, \ldots, yn\}$. we try to predict a tag for each word in the given sentence. from the given tags = D, T, O where D represents disease, T represents treatment and O represents other.

2 Dataset

The format of each line in the training dataset is token label. There is one token per line followed by a space and its label. Blank lines indicate the end of a sentence. It has a total of 3655 sentences.

3 Preprocessing

divide the dataset in train ,test, and development Train on 70% of the data.20% of data test set and 10% as development set.

first make the sentence from the given words then we map the sentences to a sequence of numbers and then pad the sequence.Then each word is encoded as 50 dimensional vector using embedding layer.For training the network we also need to change the labels y to categorical.

4 Model Used

a input sequence $x = (x_1, \ldots, x_m)$, , i.e. the words of a sentence and a sequence of output states $s = (s_1, \ldots, s_m)$. This can be done by defining a feature map

$\Phi(x_1, \ldots, x_m, s_1, \ldots, s_m) \in \mathbb{R}^d$

that maps an entire input sequence paired with an entire state sequence s to some d -dimensional feature vector. Then model the probability as a log-linear model with the parameter vector

$$p(s|x; w) = \frac{\exp(w \cdot \Phi(x, s))}{\sum_{s'} \exp(w \cdot \Phi(x, s'))}$$

where s' ranges over all possible output sequences . After that I have used non linear scoring function

$$\text{score}_{lstm-crf}(x, s) = \sum_{i=0}^{n} W_{s_{i-1}, s_i} \cdot \text{LSTM}(x)_i + b_{s_{i-1}, s_i}$$

where W_{s_{i-1}, s_i} and b are the weight vector and the bias corresponding to the transition from s_{i-1} to s_i, respectively.

5 Metric Used

- Precision
- Recall
- Accuracy
- F1 Score = 2*(Recall * Precision) / (Recall + Precision)

6 Result

Accuracy on test data 91.4%

	precision	recall	f1-score	support
D	0.76	0.63	0.69	987
O	0.95	0.97	0.96	11500
T	0.52	0.48	0.50	703
avg / total	0.91	0.91	0.91	13190

Figure 1

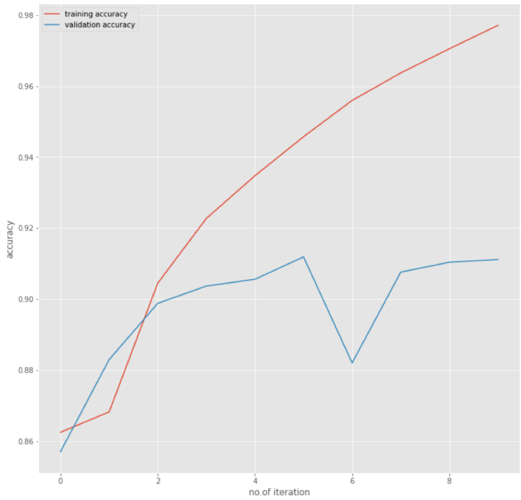

Figure 2

```
Confusion matrix, without normalization
[[  623   281    83]
 [  164 11107   229]
 [   34   334   335]]
```

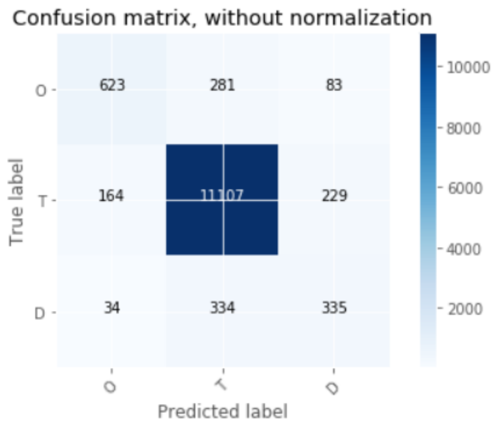

Figure 3

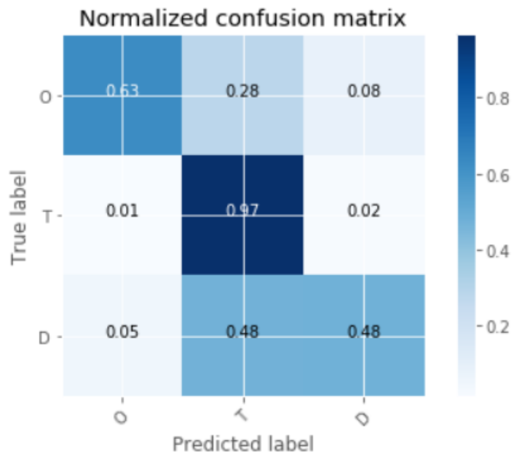

Figure 4

7 Conclusion

deep neural networks work well on the sequence tagging problems. LSTM-CRF is giving the best result to named entity recognition.

8 github link

https://git.io/vpI1U