NLP assignment2 Named Entity Recognition with Bidirectional LSTM

姓名：王涵 学号：2015211984

Keywords:
NER, LSTM, BiLSTM

Problem 4 Name Entity Recognization

Problem Description

Named Entity Recognition is a subtask of information extraction that seeks to locate and classify named entities in text into pre-defined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages.

Our objective is to build a machine learning named entity recognition system, which when given a new previously unseen text can identify and classify the named entities in the text. This means that your system should annotate each word in the text with one of the four possible classes.

Several methods can be applied to solve this problem, such as Hidden Markov Model, Conditional Random Fields and GRU, as well as LSTM.

The model I used in this experiment is a Bidirectional LSTM Model. LSTM, also called Long-Short Term Memory model is a simple recurrent neural network which can be used as a building component or block (of hidden layers) for an eventually bigger recurrent neural network. The LSTM block is itself a recurrent network because it contains recurrent connections similar to connections in a conventional recurrent neural network.

Usage

Software Requirements

```
python 3.6.2
keras=2.0.8
pandas=0.20
numpy=1.12
gensim=2.3.0
tensorflow=1.1.0
```

Dataset

CoNLL 2002 IOB Tags Dataset.
ner-dataset.csv

Source Code

Jupyter Notebook
Ipython notebook with interactive APIs.

lstm.py

Data Preprocessing

Due to the dataset is not unicode-coded, some character may be lost due to encoding problem. This problem potentially changed the structure of a ','-delimited CSV file, causing troubles when processing the tags.

After obtaining the words and the tags of the sentences. A skip-gram model(Word2Vec) is applied to extract vector representations of words. The size of a vector is set as 100. For word that is not emerged in the training set, a vector of normal distribution of size 100 is generated to represent the unseen word.

For a sentence of maximum length of 104 (of all dataset). The shape of matrix representation of a sentence is (104,100) for zero vectors take the place of default part.

The set of named-entities are {'nat', 'geo', 'art', 'gpe', 'eve', 'tim', 'per', 'O', 'org'}

The training output dim is the one-hot representation of the named-entites.

In this way, we get a model of input dim as (size,104,100) and output dim as (size,104,9)

Algorithms and Models

The model of BiLSTM consists of three layers, the first layer is a bidirectional LSTM layer, the following layer is a dropout layer, and the last layer is a full-connected layer used as softmax classification.

```
ner_model=Sequential()
ner_model.add(Bidirectional(LSTM(150,return_sequences=True,
                                 bias_regularizer=l2(0),
                                 activity_regularizer=l2(0.),
                                 kernel_regularizer=l2(0.))
                            ,
                            input_shape=(104,100)
                           ))
ner_model.add(Dropout(0.5))
ner_model.add(TimeDistributed(Dense(9,activation='softmax',
                                    bias_regularizer=l2(0.),
                                   activity_regularizer=l2(0.),
                                   kernel_regularizer=l2(0.))))
ner_model.compile(loss='categorical_crossentropy', optimizer='adam',
                  metrics=['accuracy'])
```

Loss function: categorical cross entropy Optimizer: Adam

Training and Test details

Model are trained and tested on CPU

Epoch=5 , Batch_size=200

```
Epoch 1/5
38366/38366 [==============================] - 412s - loss: 0.1862 - acc: 0.9716
Epoch 2/5
38366/38366 [==============================] - 436s - loss: 0.0520 - acc: 0.9850
Epoch 3/5
38366/38366 [==============================] - 448s - loss: 0.0430 - acc: 0.9873
Epoch 4/5
38366/38366 [==============================] - 464s - loss: 0.0386 - acc: 0.9886
Epoch 5/5
38366/38366 [==============================] - 465s - loss: 0.0357 - acc: 0.9894
```

Perfomances

After training for 5 epochs the accuracy on trainig set reached 98.94 %

We then use the model to predict on test set. And we got

```
Accuracy: 99.06%
```

The predicted tags on an unseen sentences:

> International Atomic Energy Agency is a place in America , while World Trade Organization is a place in Canada.

And we got the tagged sequence of

```
[('International', 'org'), ('Atomic', 'org'), ('Energy', 'org'),
 ('Agency', 'org'), ('is', 'O'), ('a', 'O'), ('place', 'O'),
 ('in', 'O'), ('America', 'geo'), (',', 'O'), ('while', 'O'),
 ('World', 'O'), ('Trade', 'org'), ('Organization', 'org'),
 ('is', 'O'), ('a', 'O'), ('place', 'O'), ('in', 'O'), ('Canada', 'geo')]
```

The tag is basically all right except for ('World','O')

Analysis

We can see that the BiLSTM model is doing extremely well on conll2002 dataset. For one thing, this model is one of the most state-of-art method to solve named entity recognition problem.

Firstly, I thought that the skip-gram model(Word2Vec) may take the credit from the BiLSTM model for the feature of the position of a word can be represented. In proving my thoughs, I pulled off a simple experiment. I apply the input to a Dense Layer to do softmax classification, after 3 epochs of training, the accuracy is 84 %. In that way, we can see that LSTM is crucial to position of a word. Then I predict a sentence with unseen words with the model, the result is almost random. However, the BiLSTM model is doing good.

A conclusion that LSTM is really suitable for analyzing the position of certain word in the context can be easily drawed from the comparsion to other methods. The layer wrapper of the Bidirectional helps further improve the accuracy.