bert-bilstm-crf命名实体识别

1、运行环境

- python3.6
- tensorflow=1.10.0

2、模型文件

- 其中文件out-bert，output-wwm，output-wwm-ext是输出后得到的模型。

（1）bert文件夹

存放有bert模型，bert源代码。其中bert-wmmm，bert-wwm-ext，chinese_wwm_L-12-H-768_A-12.zip，预训练好的模型存储到百度网盘中。

（2）db文件夹，是一些常用的数据库处理方法、

（3）NERdata文件夹和NLPCC2016KBQA文件夹，是使用的数据集，将存储在百度网盘中。

（4）bert_lstm_ner.py文件

该文件夹包含数据参数，bert_lstm_ner模型，该文件可以用于训练。

（5）data_process.py文件

用于语料库的处理，全部处理成小于max_seq_length的序列，这样可以避免解码出现不合法的数据或者在最后算结果的时候出现out of range 的错误。

（6）lstm_crf_layer.py文件

是BLSTM-CRF网络。

（7）terminal_predict.py文件

基于命令行的在线预测方法，需要基本的参数配置，以及预测模型的调用。

3、模型训练

模型训练需要，基于bert_lstm_ner.py文件，对模型进行训练。

其中运行的主要函数是main函数，需要自己修改参数，其运行方法主要如下。

```
task_name = "NER"
do_train = True
do_eval = True
```

```
do_predict = True
data_dir = "NERdata"
max_seq_length = 128
train_batch_size = 32
learning_rate = 2e-5
num_train_epochs = 3.0
 output_dir ="./output_qa/result_dir_ner/"
 tf.app.run()
 # filter model

 if FLAGS.filter_adam_var:
  adam_filter(FLAGS.output_dir)

```

其中一些常用参数，需要训练的时候自己设置。

4、模型预测

基于terminal_predict.py文件，实现模型预测。其中需要设置模型路径，语料路径。其使用模型的方法如下：

```
sentence = '刚开增值税普通发票时，把发票号码弄错了怎么办？打印出来的号和实际发票上的号不符，怎么办？'
predict(sentence)
def convert(line):
  feature = convert_single_example(0, line, label_list, FLAGS.max_seq_length, tokenizer, 'p')
  input_ids = np.reshape([feature.input_ids],(batch_size, FLAGS.max_seq_length))
  input_mask = np.reshape([feature.input_mask],(batch_size, FLAGS.max_seq_length))
  segment_ids = np.reshape([feature.segment_ids],(batch_size, FLAGS.max_seq_length))
  label_ids =np.reshape([feature.label_ids],(batch_size, FLAGS.max_seq_length))
  return input_ids, input_mask, segment_ids, label_ids


file = open('./question_entity_result.txt', mode='r', encoding='utf-8')
file_write = open('./question_entity_result1.txt', mode='a', encoding='utf-8')
sentences = file.readlines()
sentences_lsit = []
for line in sentences:
 sentences_lsit.append(line)
```

```
from tqdm import tqdm

with graph.as_default():
 print(id2label)
 start = datetime.now()
 for i in tqdm(range(int(len(sentences_lsit)/8))):
 sentence = sentences_lsit[i*8]
 sentence = tokenizer.tokenize(sentence)
 # print('your input is:{}'.format(sentence))
 input_ids, input_mask, segment_ids, label_ids = convert(sentence)

 feed_dict = {input_ids_p: input_ids,
 input_mask_p: input_mask,
 segment_ids_p:segment_ids,
 label_ids_p:label_ids}
 # run session get current feed_dict result
 pred_ids_result = sess.run([pred_ids], feed_dict)
 pred_label_result = convert_id_to_label(pred_ids_result, id2label)
 #todo: 组合策略
 result = strage_combined_link_org_loc(sentence, pred_label_result[0])

 file_write.write(sentences_lsit[i * 8])
 file_write.write(sentences_lsit[i * 8 + 1])
 file_write.write(sentences_lsit[i * 8 + 2])
 file_write.write(sentences_lsit[i * 8 + 3])
 file_write.write(sentences_lsit[i * 8 + 4])
 file_write.write(sentences_lsit[i * 8 + 5])
 file_write.write('bert_lstm_crf_book:' + str([result])+'\n')
 file_write.write(sentences_lsit[i * 8 + 6])
 file_write.write(sentences[i * 8 + 7])

```

其中包括，单个文本的预测，以及批量的预测。