基于 BERT 的标点符号预测模型

运行环境：

python3 、Tensorflow 1.9 以上
titan xp 12G

数据集

IWSLT2012

预训练 bert 模型

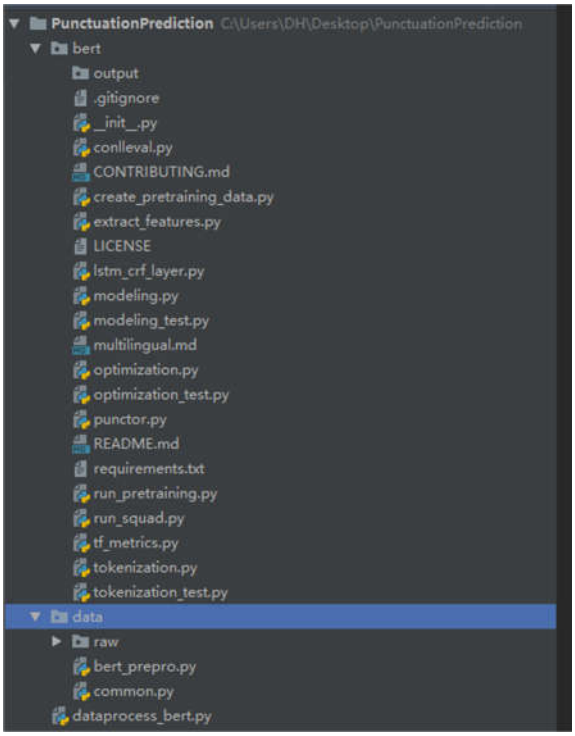
uncased_L-24_H-1024_A-16　Large
uncased_L-12_H-768_A-12　Base
下载地址：
https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-24_H-1024_A-16.zip
https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zip
使用 Large 模型时 batch_size 要设小一点，否则显存不够

预处理、训练

预处理 ： 运行 dataprocess_bert.py 可得到 训练、验证所需的 json 文件
训练： 运行/bert/punctor.py 即可完成训练、验证、测试。可更改超参对过程进行选择

```
flags.DEFINE_boolean('clean', True, 'remove the files which created by last training')
flags.DEFINE_bool("do_train", False, "Whether to run training.")
flags.DEFINE_bool("use_tpu", False, "Whether to use TPU or GPU/CPU.")
flags.DEFINE_bool("do_eval", True, "Whether to run eval on the dev set.")
flags.DEFINE_bool("do_predict", True, "Whether to run eval on the dev set.")
```

项目结构：



\bert\output　模型结构的输出
\bert\ 中其它文件均与 bert 或模型相关，目前只需关注 punctor.py 和 tokenization.py，后面会进行详细说明

\data\raw\LREC 中的文件为训练及测试数据的原始文件
\data\raw\LREC_converted 中为处理过后的数据，一般使用此处数据进行训练及测试。
\data\bert_prepro.py 为对数据进行预处理，处理成如下的格式的 json 文件,其中 chars 没有用到。
[{'words':["i","'m","as","a","font","or","more","precisely","a","high-functioning","autistic","savant"],'chars':[...],'tags':["_SPACE","_SPACE","_SPACE","_SPACE",",COMMA","_SPACE","_SPACE",",COMMA","_SPACE","_SPACE","_SPACE",".PERIOD"]}]
\data\common.py 设置了一些符号的标记，以及一些数据的处理。
\data\dataprocess_bert.py　设置了一些文件的路径，运行此文件即可完成对数据的预处理，将要使用的数据存储在\data\dataset\lrec 中。

主要的代码文件说明

\bert\punctor.py
punctor.py 是对原始 bert 代码中 run_classifier.py 的更改。运行 punctor.py 即可完成模型的训练、验证及预测。以下是对这个文件中主要的类和方法进行说明：
class InputExample(object):和 class InputFeatures(object): 是构建 example 和 feature

class DataProcessor(object): 是一个需要继承重写的类 。其中的

def _read_data(cls, input_file, quotechar=None) 方法是读取数据，返回一个 lines，lines 里面的每个元素为一个形为 [words,labels] 的 list 其中 words 和 labels 均为字符串。

class PunctorProcessor(DataProcessor): 为继承 DataProcessor 的类。其中的

def get_train_examples(self, data_dir)和 def get_dev_examples(self, data_dir):分别是对训练集和验证集 json 文件中获取 example。

def get_test_examples(self,data_dir) 是从原始的测试集(ref.txt 和 asr.txt,而不是 json 文件)获取 example，所以和训练和验证集获取的方式不一样。

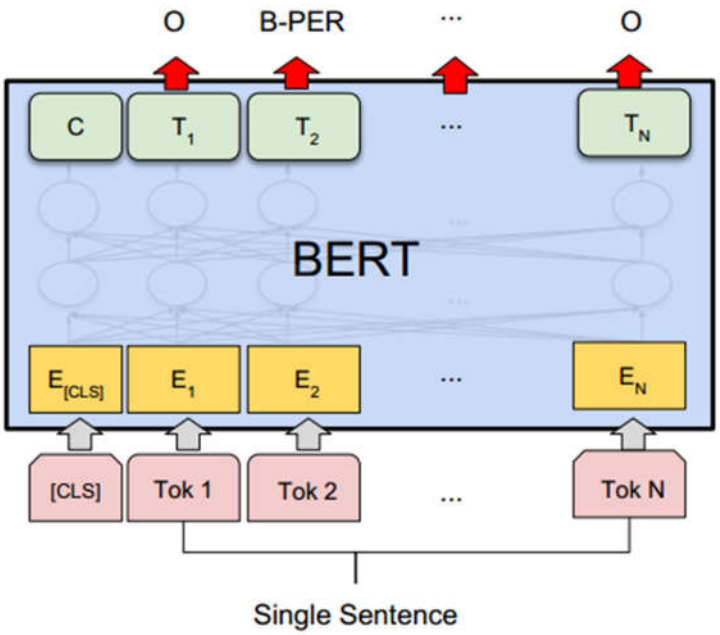
def get_labels(self) 设置标签。

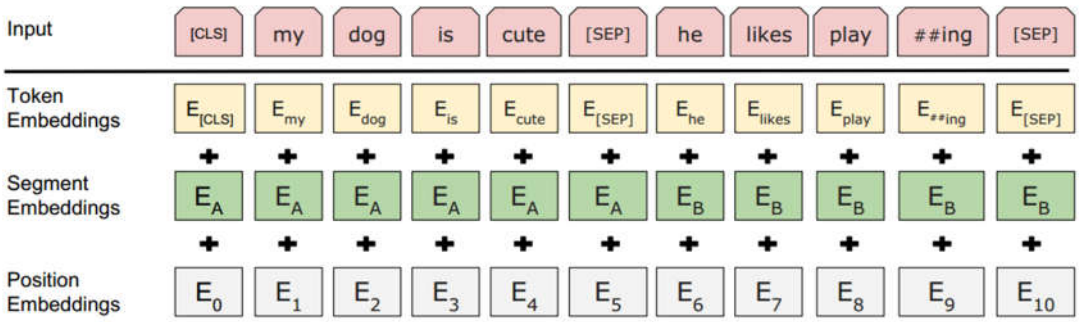
def _create_example(self, lines, set_type): 对 read_data 获取的 lines 封装成需要的 examples，其中 guid 为每个 example 唯一的标识。

def convert_single_example(ex_index, example, label_list, max_seq_length, tokenizer, mode):对每个 example 转化成需要的形式，即获取每个 word 的 id，每个 label 的 id，mask 等信息，形式如下：

```
*** Example ***
guid: test-4
tokens: it so if you played chess your know that sixty four i
input_ids: 2009 2061 2065 2017 2209 7433 2115 2113 2008 8442
input_mask: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
segment_ids: 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
label_ids: 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0
```

这里对 bert 的原始版本做了一些修改。





bert 论文中的输入结构如上图，可以看到在句子的开头有[CLS]标志，结尾有[SEP]标志。句中单词 playing 被分词成 play 和 ##ing，也就是对一些词进行了子词的划分。

我做的修改：

1.去掉了[CLS] 和[SEP]，因为我考虑到在预测的时候可能在句中出现[CLS]和[SEP]的 label，这种标记不属于标点符号，如果加上，需要后续处理还没想好，如果只是在句首或者句尾出现这种 label 就很好处理。

2.没有对每个单词进行子词的划分，不会把 playing 分成 play 和 ##ing。因为如果分开我没想好##ing 的 label 是什么。比如：原始的数据是 words 长度为 200， 对应的 labels 长度也为 200。如果进行子词的切分，words 会增加，而 label 的处理会成为一个问题。所以我在 def convert_single_example(ex_index, example, label_list, max_seq_length, tokenizer, mode)：中使用 split(' ')来进行分词，没有使用 bert 的分词器划分子词。因为没有划分子词，所以有些词在 bert 的 vocab 中找不到，我在 tokenization.py 中 def convert_by_vocab(vocab, items):对找不到的词统一替换为[UNK]。如果是中文就可以很好地解决子词标记这个问题。

def softmax_create_model(bert_config, is_training, input_ids, input_mask,segment_ids, labels, num_labels, use_one_hot_embeddings):用来构建模型，在这个方法中我在 bert 的输出后简单地接了一个全连接层。之后用交叉熵算 loss。

实验结果

REF 测试集

```
INFO:tensorflow:
Evaluate on ref:
-----------------------------------------------
PUNCTUATION      PRECISION RECALL    F-SCORE
,COMMA           71.61     73.86     72.72
.PERIOD          85.40     85.71     85.55
?QUESTIONMARK    70.91     84.78     77.23
-----------------------------------------------
Overall          78.07     79.83     78.94
ERR: 4.1%
SER: 30.8%
```

ASR 测试集

```
Evaluate on asr:
-----------------------------------------------
PUNCTUATION       PRECISION RECALL    F-SCORE
,COMMA            48.94     66.29     56.31
.PERIOD           76.84     77.60     77.22
?QUESTIONMARK     47.92     65.71     55.42
-----------------------------------------------
Overall           60.62     71.85     65.76
ERR: 7.5%
SER: 58.7%
```

单模型效果超过之前所有的模型，之前的一些模型分数如下

1. Self-Attention Based Network for Punctuation Restoration 这篇论文只做了 REF 测试集

Model	Period			Comma			Question			Overall		
	Pr.	Re.	F1	Pr.	Re.	F1	Pr.	Re.	F1	Pr.	Re.	F1
T-LSLM	60.2	53.4	56.6	49.6	41.4	45.1	57.1	43.5	49.4	55.0	47.2	50.8
T-BRNN	72.3	71.5	71.9	64.4	45.2	53.1	67.5	58.7	62.8	68.9	58.1	63.1
T-BRNN-pre	73.3	72.5	72.9	65.5	47.1	54.8	70.7	62.8	66.7	70.0	59.7	64.4
Word CNN	75.8	95.5	84.6	56.1	60.4	58.2	40.8	55.7	47.1	65.8	77.7	71.3
SAPR	96.7	97.3	96.8	57.2	50.8	55.9	70.6	69.2	70.3	78.2	74.4	77.4

TABLE I: Results of the Single Punctuation task on IWSLT dataset

2. Bidirectional Recurrent Neural Network with Attention Mechanism forPunctuation Restoration

	Model	*COMMA*			*PERIOD*			*QUESTION*			*OVERALL*			
		Pr.	*Re.*	F_1	*Pr.*	*Re.*	F_1	*Pr.*	*Re.*	F_1	*Pr.*	*Re.*	F_1	*SER*
Ref.	DNN [5]	58.2	35.7	44.2	61.6	64.8	63.2	0	0	-	60.3	48.6	53.8	62.9
	DNN-A [5]	48.6	42.4	45.3	59.7	68.3	63.7	0	0	-	54.8	53.6	54.2	66.9
	CNN-2A [5]	48.1	44.5	46.2	57.6	69.0	62.8	0	0	-	53.4	55.0	54.2	68.0
	T-LSTM [17]	49.6	41.4	45.1	60.2	53.4	56.6	57.1	43.5	49.4	55.0	47.2	50.8	74.0
	T-BRNN	64.4	45.2	53.1	72.3	71.5	71.9	67.5	58.7	62.8	68.9	58.1	63.1	51.3
	T-BRNN-pre	**65.5**	**47.1**	**54.8**	**73.3**	**72.5**	**72.9**	**70.7**	**63.0**	**66.7**	**70.0**	**59.7**	**64.4**	**49.7**
ASR	DNN [5]	47.2	32.0	38.1	59.0	60.9	60.0	0	0	-	54.4	45.6	49.6	73.3
	DNN-A [5]	41.0	40.9	40.9	56.2	64.5	60.1	0	0	-	49.2	51.6	50.4	79.2
	CNN-2A [5]	37.3	40.5	38.8	54.6	65.5	59.6	0	0	-	46.4	51.9	49.1	83.6
	T-LSTM [17]	41.8	37.8	39.7	56.4	49.3	52.6	55.6	42.9	48.4	49.1	43.6	46.2	83.7
	T-BRNN	**60.0**	**45.1**	**51.5**	69.7	69.2	69.4	**61.5**	45.7	52.5	65.5	57.0	60.9	57.8
	T-BRNN-pre	59.6	42.9	49.9	**70.7**	**72.0**	**71.4**	60.7	**48.6**	**54.0**	**66.0**	**57.3**	**61.4**	**57.0**

3. Distilling Knowledge from an Ensemble of Models for Punctuation Prediction

	Model	Comma			Period			Question			Overall		
		P(%)	*R(%)*	$F_1(\%)$	*P(%)*	*R(%)*	$F_1(\%)$	*P(%)*	*R(%)*	$F_1(\%)$	*P(%)*	*R(%)*	$F_1(\%)$
Ref.	DNN	58.1	35.8	44.3	62.1	64.8	63.4	60.5	48.9	54.1	60.2	49.8	53.9
	T-BRNN-pre [17]	65.5	47.1	54.8	73.3	72.5	72.9	70.7	63.0	66.7	70.0	59.7	64.4
	BLSTM-CRF	58.9	59.1	59.0	68.9	72.1	70.5	71.8	60.6	65.7	66.5	63.9	65.1
	Teacher-Ensemble	**66.2**	**59.9**	**62.9**	**75.1**	**73.7**	**74.4**	**72.3**	**63.8**	**67.8**	**71.2**	**65.8**	**68.4**
ASR	DNN	47.5	32.3	38.5	58.3	60.5	59.4	57.1	46.8	51.4	54.3	46.5	49.8
	T-BRNN-pre [17]	59.6	42.9	49.9	70.7	72.0	71.4	60.7	48.6	54.0	66.0	57.3	61.4
	BLSTM-CRF	55.7	56.8	56.2	68.7	71.5	70.1	63.8	53.4	58.1	62.7	60.6	61.5
	Teacher-Ensemble	**60.6**	**58.3**	**59.4**	**71.7**	**72.9**	**72.3**	**66.2**	**55.8**	**60.6**	**66.2**	**62.3**	**64.1**