NLP（二十二）利用ALBERT实现文本二分类

原创 jclian Python爬虫与算法 3月4日

收录于话题
#NLP入门系列文章

29个

在文章NLP（二十）利用BERT实现文本二分类中，笔者介绍了如何使用BERT来实现文本二分类功能，以判别是否属于出访类事件为例子。但是呢，利用BERT在做模型预测的时候存在预测时间较长的问题。因此，我们考虑用新出来的预训练模型来加快模型预测速度。

本文将介绍如何利用ALBERT来实现文本二分类。

关于ALBERT

ALBERT的提出时间大约是在2019年10月，其第一作者为谷歌科学家蓝振忠博士。ALBERT的论文地址为：https://openreview.net/pdf?id=H1eA7AEtvS ，Github项目地址为：https://github.com/brightmart/albert_zh 。

简单说来，ALBERT是BERT的一个精简版，它在BERT模型的基础上进行改造，减少了大量参数，使得其在模型训练和模型预测的速度上有很大提升，而模型的效果只会有微小幅度的下降，具体的效果和速度方面的说明可以参考Github项目。

ALBERT相对于BERT的改进如下：

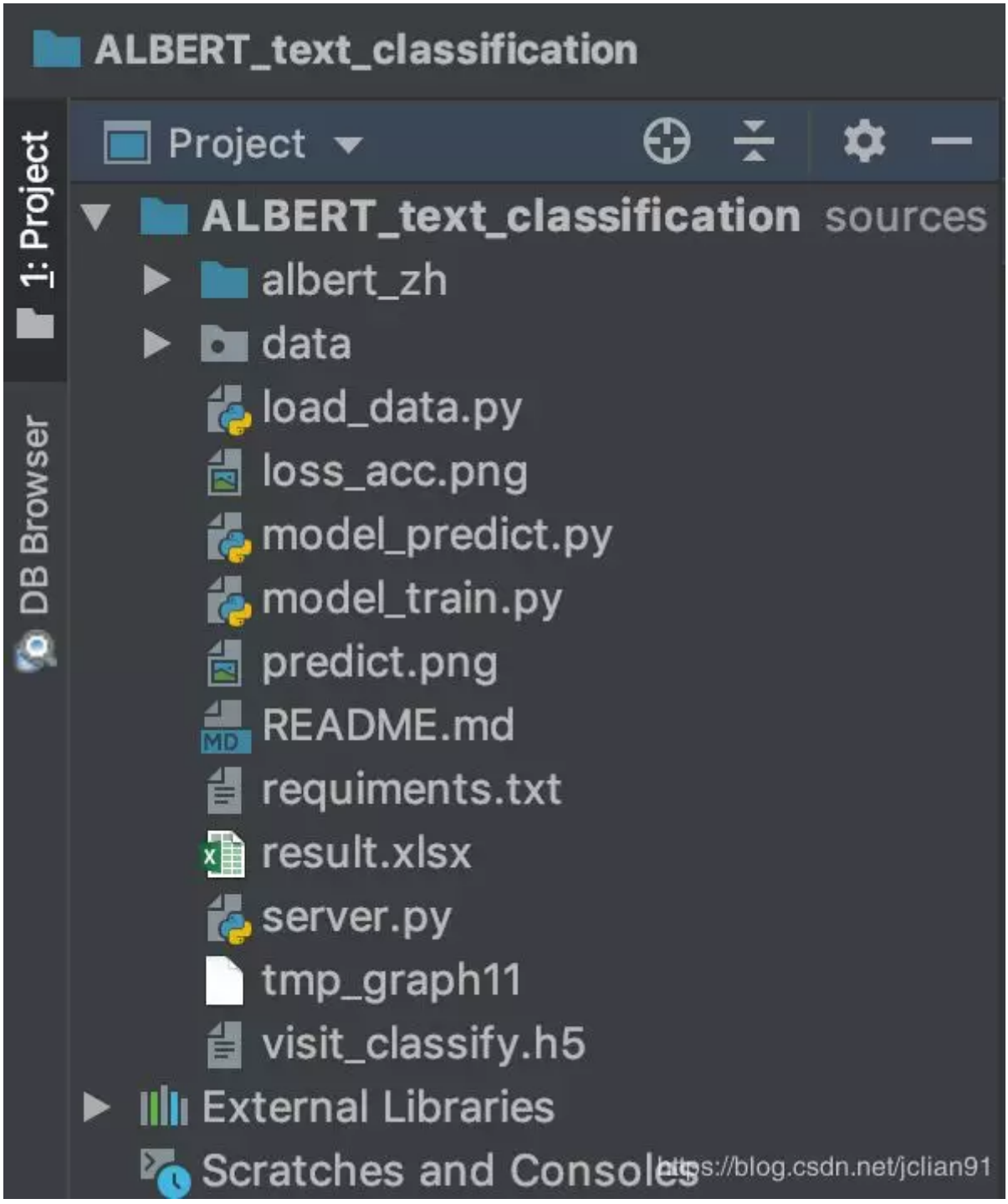
- 对Embedding因式分解（Factorized embedding parameterization）；
- 跨层的参数共享（Cross-layer parameter sharing）；
- 句间连贯（Inter-sentence coherence loss）；
- 移除dropout 。

笔者在北京的时候也写过ALBERT在提升序列标注算法的预测速度方面的一篇文章：NLP（十八）利用ALBERT提升模型预测速度的一次尝试 ，该项目的Github地址为：https://github.com/percent4/ALBERT_4_Time_Recognition 。

项目说明

本项目的数据和代码主要参考笔者的文章NLP（二十）利用BERT实现文本二分类，该项目是想判别输入的句子是否属于政治上的出访类事件。笔者一共收集了340条数据，其中280条用作训练集，60条用作测试集。

项目结构如下图：



项目结构

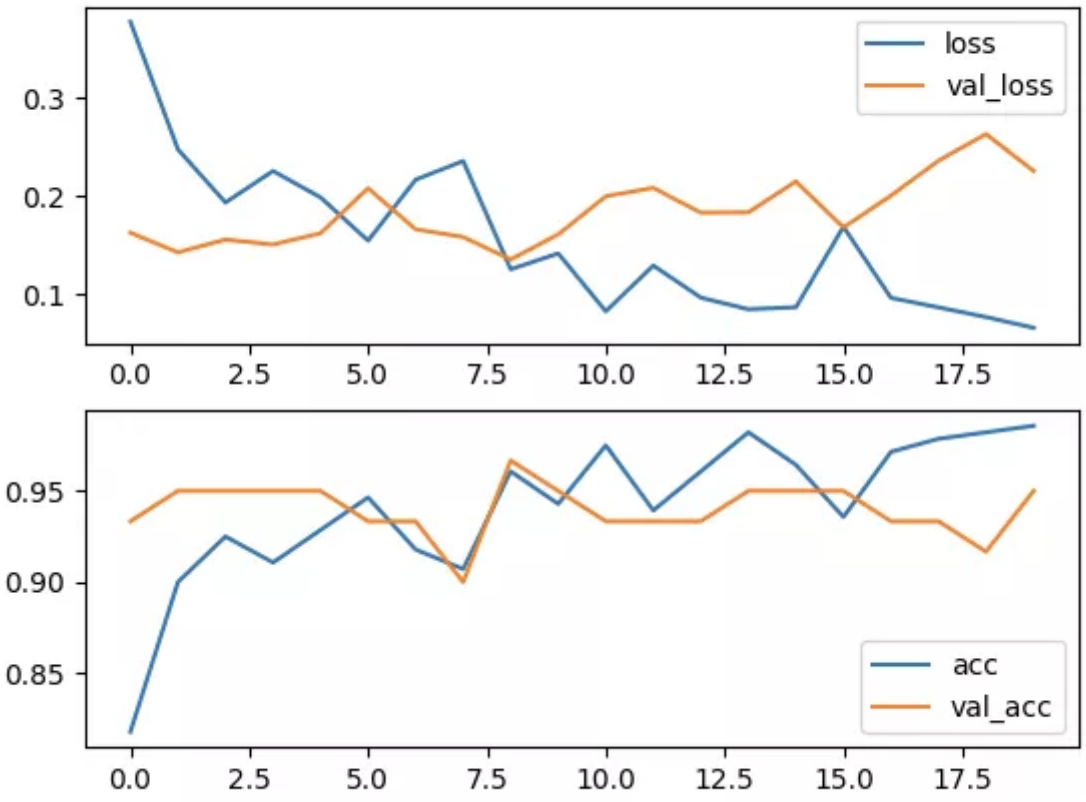
在这里我们使用ALBERT已经训练好的文件 albert_tiny ，借鉴BERT的调用方法，我们在这里给出 albert_zh 模块，能够让ALBERT提取文本的特征，具体代码不在这里给出，有兴趣的读者可以访问该项目的Github地址：。

注意， albert_tiny 给出的向量维度为312，我们的模型训练代码（model_train.py）如下：

```
# -*- coding: utf-8 -*-
# author: Jclian91
# place: Pudong Shanghai
# time: 2020-03-04 13:37

import os

import numpy as np
from load_data import train_df, test_df
from keras.utils import to_categorical
from keras.models import Model
from keras.optimizers import Adam
from keras.layers import Input, BatchNormalization, Dense
import matplotlib.pyplot as plt

from albert_zh.extract_feature import BertVector

# 读取文件并进行转换
bert_model = BertVector(pooling_strategy="REDUCE_MEAN", max_seq_len=100)
print('begin encoding')
f = lambda text: bert_model.encode([text])["encodes"][0]
train_df['x'] = train_df['text'].apply(f)
test_df['x'] = test_df['text'].apply(f)
print('end encoding')

x_train = np.array([vec for vec in train_df['x']])
x_test = np.array([vec for vec in test_df['x']])
y_train = np.array([vec for vec in train_df['label']])
y_test = np.array([vec for vec in test_df['label']])
print('x_train: ', x_train.shape)

# Convert class vectors to binary class matrices.
num_classes = 2
y_train = to_categorical(y_train, num_classes)
y_test = to_categorical(y_test, num_classes)

# 创建模型
x_in = Input(shape=(312, ))
x_out = Dense(32, activation="relu")(x_in)
x_out = BatchNormalization()(x_out)
x_out = Dense(num_classes, activation="softmax")(x_out)
model = Model(inputs=x_in, outputs=x_out)
print(model.summary())

model.compile(loss='categorical_crossentropy',
              optimizer=Adam(),
              metrics=['accuracy'])

# 模型训练以及评估
history = model.fit(x_train, y_train, validation_data=(x_test, y_test), batch_size=8, e
model.save('visit_classify.h5')
print(model.evaluate(x_test, y_test))

# 绘制loss和acc图像
plt.subplot(2, 1, 1)
epochs = len(history.history['loss'])
```

```
plt.plot(range(epochs), history.history['loss'], label='loss')
plt.plot(range(epochs), history.history['val_loss'], label='val_loss')
plt.legend()

plt.subplot(2, 1, 2)
epochs = len(history.history['acc'])
plt.plot(range(epochs), history.history['acc'], label='acc')
plt.plot(range(epochs), history.history['val_acc'], label='val_acc')
plt.legend()
plt.savefig("loss_acc.png")
```

模型训练的效果很不错，在训练集的acc为0.9857,在测试集上的acc为0.9500，具体如下：



训练过程中的loss和acc图

与BERT的预测对比

接下来我们在模型预测上的时间，与BERT的文本二分类模型预测时间做一个对比，这样有助于提升我们对ALBERT的印象。

BERT的文本二分类模型预测可以参考文章NLP（二十）利用BERT实现文本二分类，本文给出的代码与BERT实现的模型预测代码基本一致，只不过BERT提取特征改成

ALBERT提取特征。

本文的模型预测代码（model_predict.py）如下：

```
# -*- coding: utf-8 -*-
# author: Jclian91
# place: Pudong Shanghai
# time: 2020-03-04 17:33

import time
import pandas as pd
import numpy as np
from albert_zh.extract_feature import BertVector
from keras.models import load_model
load_model = load_model("visit_classify.h5")

# 预测语句
texts = ['在访问限制中，用户可以选择禁用iPhone的功能，包括Siri、iTunes购买功能、安装/删除应用
         'IT之家4月23日消息 近日，谷歌在其官方论坛发布消息表示，他们为Android Auto添加了一项
         '要通过telnet 访问路由器，需要先通过console 口对路由器进行基本配置，例如：IP地址、密
         'IT之家3月26日消息 近日反盗版的国际咨询公司MUSO发布了2017年的年度报告，其中的数据显示
         '2月26日至3月2日，应香港特区政府“内地贵宾访港计划”邀请，省委常委、常务副省长陈向群赴港
         '目前A站已经恢复了访问，可以直接登录，网页加载正常，视频已经可以正常播放。',
         '难民署特使安吉丽娜·朱莉6月8日结束了对哥伦比亚和委内瑞拉边境地区的难民营地为期两天的访
         '据《南德意志报》报道，德国总理默克尔计划明年1月就前往安卡拉，和土耳其总统埃尔多安进行
         'Win7电脑提示无线适配器或访问点有问题怎么办?很多用户在使用无线网连接上网时，发现无线网
         '未开发所有安全组之前访问，FTP可以链接上，但是打开会很慢，需要1-2分钟才能链接上',
         'win7系统电脑的用户，在连接WIFI网络网上时，有时候会遇到突然上不了网，查看连接的WIFI出
         '联合国秘书长潘基文 8 日访问了日本福岛县，与当地灾民交流并访问了一所高中。',
         '正在中国访问的巴巴多斯总理斯图尔特 1 5 日在陕西西安参观访问。',
         '据外媒报道,当地时间10日,美国白宫发声明称,美国总统特朗普将于2月底访问印度,与印度总理
         '2月28日，唐山曹妃甸蓝色海洋科技有限公司董事长赵力军等一行5人到黄海水产研究所交流访问
         '2018年7月2日，莫斯科孔子文化促进会会长姜彦彬，常务副会长陈国建，在中国著名留俄油画大
         '据外媒报道，当地时间26日晚，阿尔及利亚总统特本抵达沙特阿拉伯，进行为期三天的访问。两
         '与标准Mozy一样，Stash文件夹为用户提供了对其备份文件的基于云的访问，但是它们还使他们
         '研究表明，每个网页的平均预期寿命为44至100天。当用户通过浏览器访问已消失的网页时，就会
         '据外媒报道，土耳其总统府于当地时间2日表示，土耳其总统埃尔多安计划于5日对俄罗斯进行为
         '3日，根据三星电子的消息，李在镕副会长这天访问了位于韩国庆尚北道龟尾市的三星电子工厂。

labels = []

bert_model = BertVector(pooling_strategy="REDUCE_MEAN", max_seq_len=100)

init_time = time.time()

# 对上述句子进行预测
for text in texts:

    # 将句子转换成向量
    vec = bert_model.encode([text])["encodes"][0]
    x_train = np.array([vec])

    # 模型预测
    predicted = load_model.predict(x_train)
    y = np.argmax(predicted[0])
    label = 'Y' if y else 'N'
    labels.append(label)
```

```
cost_time = time.time() - init_time
print("Average cost time: %s." % (cost_time/len(texts)))

for text, label in zip(texts, labels):
    print('%s\t%s' % (label, text))

df = pd.DataFrame({'句子':texts, "是否属于出访类事件": labels})
df.to_excel('./result.xlsx', index=False)
```

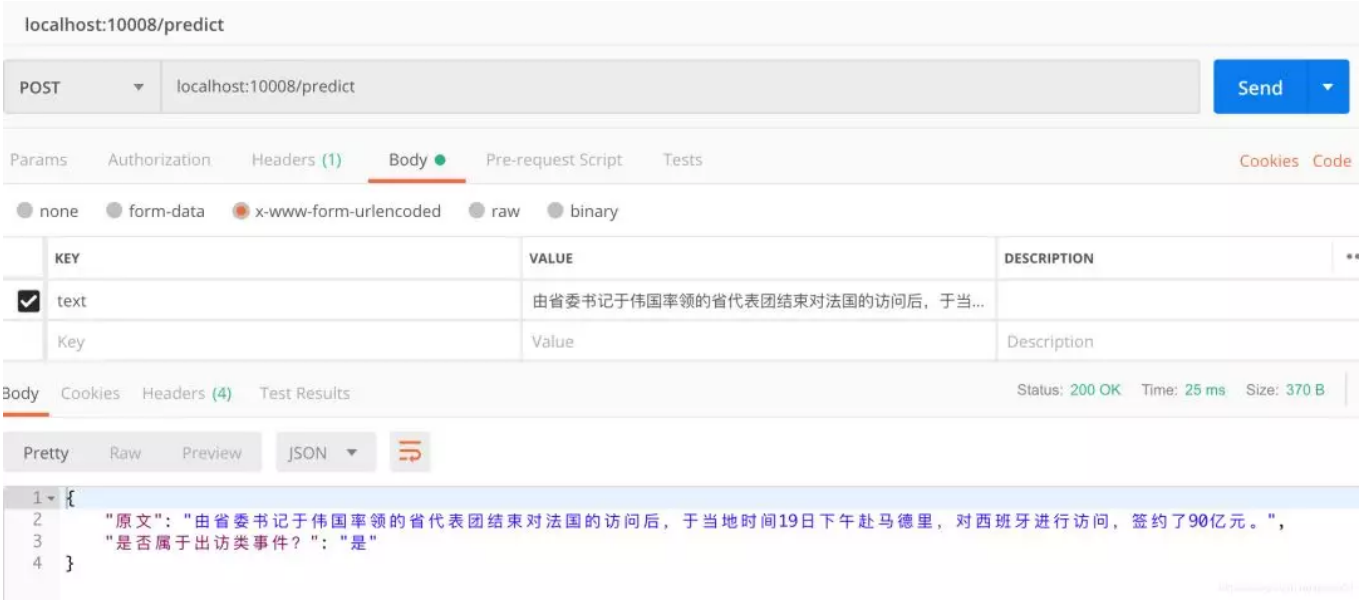
输出的平均预测时长为： 16.98ms ，而BERT版的平均预测时间为： 257.31ms 。

我们将模型预测写成HTTP服务，代码（server.py）如下：

```
# -*- coding: utf-8 -*-
# author: Jclian91
# place: Pudong Shanghai
# time: 2020-03-04 20:13

import tornado.httpserver
import tornado.ioloop
import tornado.options
import tornado.web
from tornado.options import define, options

import json
import numpy as np
from albert_zh.extract_feature import BertVector
from keras.models import load_model


# 定义端口为10008
define("port", default=10008, help="run on the given port", type=int)

# 加载ALBERT
bert_model = BertVector(pooling_strategy="REDUCE_MEAN", max_seq_len=100)
# 加载已经训练好的模型
load_model = load_model("visit_classify.h5")


# 对句子进行预测
class PredictHandler(tornado.web.RequestHandler):

    def post(self):

        text = self.get_argument("text")

        # 将句子转换成向量
        vec = bert_model.encode([text])["encodes"][0]
        x_train = np.array([vec])

        # 模型预测
        predicted = load_model.predict(x_train)
        y = np.argmax(predicted[0])
        label = '是' if y else "否"
```

```
        # 返回结果
        result = {"原文": text, "是否属于出访类事件？": label}

        self.write(json.dumps(result, ensure_ascii=False, indent=2))


# 主函数
def main():

    # 开启tornado服务
    tornado.options.parse_command_line()
    # 定义app
    app = tornado.web.Application(
            handlers=[(r'/predict', PredictHandler)] #网页路径控制
           )
    http_server = tornado.httpserver.HTTPServer(app)
    http_server.listen(options.port)
    tornado.ioloop.IOLoop.instance().start()


main()
```

用Postman进行测试，如下图：



实践证明，用ALBERT做文本特征提取，模型训练的效果基本与BERT差别微小，模型训练速度明显提升，更重要的是，模型预测的速度只有BERT版本的6.6%（不同情况下可能有略微差异），这在生产上是十分有帮助的。

参考网址

1. 中文预训练ALBERT模型来了：小模型登顶GLUE，Base版模型小10倍速度快1倍：https://zhuanlan.zhihu.com/p/85037097