NLP - 15 分钟搭建中文文本分类模型

原创 艾力亚尔 AINLP 2019-01-29

本文系作者投稿，作者：艾力亚尔（微博 @艾力亚尔），暴风大脑研究院研发工程师，现负责电视端的语音助手相关工作。

原文地址，可点击文末“阅读原文”直达：

https://eliyar.biz/nlp_chinese_text_classification_in_15mins/

欢迎大家投稿，AI、NLP相关即可。

文本分类是自然语言处理核心任务之一，常见用文本审核、广告过滤、情感分析、语音控制和反黄识别等NLP领域。本文主要是介绍我最近开源的极简文本分类和序列标注框架 Kashgari （https://github.com/BrikerMan/Kashgari）

搭建环境和数据准备

准备工作，先准备 python 环境和数据集。

- Python 3.6 环境
- THUCNews 数据集子集,链接: https://pan.baidu.com/s/1hugrfRu 密码: qfud

如果需要完整数据集请自行到 THUCTC：一个高效的中文文本分类工具包 下载，请遵循数据提供方的开源协议。上面的子数据集包括一下 10 个分类。

```
1  体育，财经，房产，家居，教育，科技，时尚，时政，游戏，娱
```

每个分类 6500 条数据。感谢 @gaussic 在使用卷积神经网络以及循环神经网络进行中文文本分类 分享。

虚拟环境中安装所有需要的依赖

```
1  pip install jieba
2  pip install kashgari
3  pip install tensorflow
```

```
# 如果有 GPU 则可以安装
pip install tensorflow-gpu
```

数据分别为格式为一样一条新闻，每一行是 `分类\t新闻内容` 格式。我们需要把新闻内容分词后作为输入喂给模型。

```
import tqdm
import jieba

def read_data_file(path):
    lines = open(path, 'r', encoding='utf-8').read().splitline
    x_list = []
    y_list = []
    for line in tqdm.tqdm(lines):
        rows = line.split('\t')
        if len(rows) >= 2:
            y_list.append(rows[0])
            x_list.append(list(jieba.cut('\t'.join(rows[1:]))))
        else:
            print(rows)
    return x_list, y_list

test_x, test_y = read_file('cnews/cnews.test.txt')
train_x, train_y = read_file('cnews/cnews.train.txt')
val_x, val_y = read_file('cnews/cnews.val.txt')
```

训练与验证模型

Kashgari 目前提供了三种分类模型结构 `CNNModel` `CNNLSTMModel` 和 `BLSTMModel` 。我们先使用 `CNNModel`

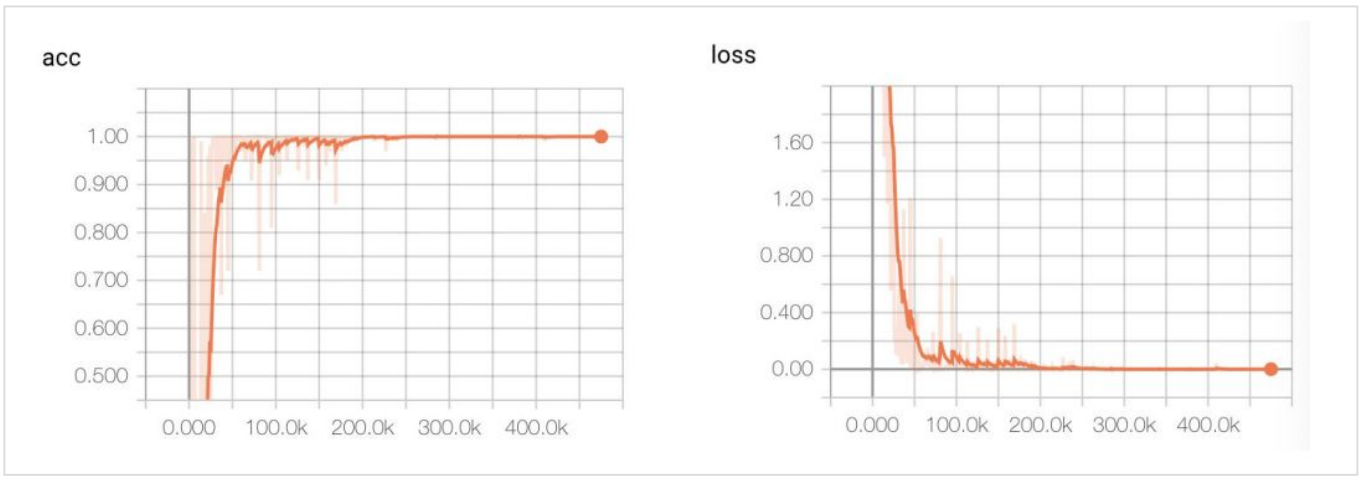
```
from kashgari.tasks.classification import CNNModel

model = CNNModel()
model.fit(train_x, train_y, val_x, val_y, batch_size=12
```

运行结果

```
Layer (type)                 Output Shape              Param #
=================================================================
```

```
input_17 (InputLayer)        (None, 1462)              0
_________________________________________________________________
embedding_17 (Embedding)     (None, 1462, 100)         19139400
_________________________________________________________________
conv1d_9 (Conv1D)            (None, 1458, 128)         64128
_________________________________________________________________
global_max_pooling1d_8 (Glob (None, 128)               0
_________________________________________________________________
dense_16 (Dense)             (None, 64)                8256
_________________________________________________________________
dense_17 (Dense)             (None, 11)                715
=================================================================
Total params: 19,212,499
Trainable params: 19,212,499
Non-trainable params: 0
_________________________________________________________________
Epoch 1/5
950/950 [==============================] - 28s 30ms/step - loss: 0.6581 - acc
Epoch 2/5
950/950 [==============================] - 28s 29ms/step - loss: 0.0403 - acc
Epoch 3/5
950/950 [==============================] - 27s 29ms/step - loss: 0.0068 - acc
...
```



由于数据特征比较明显，几轮就达到了 0.9999，val_acc 都 1.0 了。

再拿验证机验证一下模型，测试集上 F1 达到了 0.98，相当不错的成绩了。

```
model.evaluate(test_x, test_y)
"""
```

```
              precision    recall  f1-score   support

          体育       1.00      1.00      1.00      10
          娱乐       0.99      0.99      0.99      10
          家居       0.99      0.94      0.97      10
          房产       1.00      1.00      1.00      10
          教育       0.98      0.95      0.97      10
          时尚       0.99      0.99      0.99      10
          时政       0.96      0.97      0.97      10
          游戏       0.99      0.99      0.99      10
          科技       0.97      0.99      0.98      10
          财经       0.96      1.00      0.98      10

   micro avg       0.98      0.98      0.98     1000
   macro avg       0.98      0.98      0.98     1000
weighted avg       0.98      0.98      0.98     1000
"""
```

保存模型和加载保存模型

模型的保存和重新加载都非常简单

```
model.save('./model')

new_model = CNNModel.load_model('./model')
news = """「DeepMind 击败人类职业玩家的方式与他们声称的 AI 使命，以及所声称的『正确』
DeepMind 的人工智能 AlphaStar 一战成名，击败两名人类职业选手。掌声和欢呼之余，它也引起
"""
x = list(jieba.cut(news))
new_model.predict(x)
'游戏'
```

使用 tensorboard 可视化训练过程

Kashgari 是基于 Keras 封装，所以可以很方便的使用 keras 的各种回调函数来记录训练过程，比如我们可以使用 `keras.callbacks.TensorBoard` 来可视化训练过程。

```
import keras

```

```
tf_board_callback = keras.callbacks.TensorBoard(log_dir='./logs', update_freq


model = CNNModel()
model.fit(train_x,
          train_y,
          val_x,
          val_y,
          batch_size=100,
          fit_kwargs={'callbacks': [tf_board_callback]})
```

在项目目录运行下面代码即可启动 tensorboard 查看可视化效果

```
$ tensorboard --log-dir logs
```

使用预训练词向量

由于长新闻特征比较明显，语料量也比较大，很容易取得比较不错的结果。但是如果我们的语料比较少，特征不是很明显时候直接训练可能会导致模型过拟合，泛化能力很差，此时我们可以使用预训练的词 Embedding 层来提高模型的泛化能力。

```
# 初始化 word2vec embedding
from kashgari.embeddings import WordEmbeddings
embedding = WordEmbeddings('<embedding-file-path>', sequence_length=6

# 初始化 BERT embedding
from kashgari.embeddings import BERTEmbedding
embedding = BERTEmbedding('bert-base-chinese', sequence_length=600)

# 使用 embedding 初始化模型
from kashgari.tasks.classification import CNNModel
model = CNNModel(embedding)
```

参考

- 使用卷积神经网络以及循环神经网络进行中文文本分类
- 中文文本分类对比（经典方法和CNN）