**说明文档**

本代码提供了预先训练好的模型，该模型在 checkpoints/textcnn/ind\_sl15\_fc1 （行业编码）和 checkpoints/textcnn/occ\_sl15\_fc1（职业编码）下。

而本文下面**红字下方的内容是从数据处理到训练测试的完整运行过程**。

**1、环境**

- Python 2/3

- TensorFlow 1.3以上

- numpy

- scikit-learn

- scipy

**2、数据集**

北京大学开放研究数据平台获取的（赵, 银霞, 2018, ）“中国某社科调查的职业和行业编码数据”作为数据集;

其中包含两个数据集文件，重命名后为data/dataset1.xlsx和data/dataset2.xlsx;

样本说明和分类类别信息在data/classes.xls中;

对数据集的基本分析结果在data/analysis/下。

**3、预处理**

`data/helper.py`为数据的预处理工具文件。

- `read\_file()`: 读取文件数据;

- `build\_vocab()`: 构建词汇表，使用字符级的表示，这一函数会将词汇表存储下来，避免每一次重复处理;

- `read\_vocab()`: 读取上一步存储的词汇表，转换为`{词：id}`表示;

- `read\_category()`: 读取各级编码与类别号映射，转换为`{类别: id}`表示;

- `read\_encode()`: 读取所有完整编码类别，包括0和999999，转换为`{类别: id}`表示;

- `to\_words()`: 将一条由id表示的数据重新转换为文字;

- `process\_file()`: 将数据集从文字转换为固定长度的id序列表示;

- `batch\_iter()`: 为神经网络的训练准备经过shuffle的批次的数据。

- `multi\_sub`:替换字符串中多个指定位置为指定字符;

- `is\_chinese`:判断是否是中文;

- `is\_digit`:判断是否是数字;

- `is\_letter`:判断是否是英文字母;

- `getlistnum`:对列表的每个元素进行计数;

- `cut\_by\_seqlength`:将文本切成固定长度的若干段.

**预处理完整过程为（按顺序运行下列文件）（上传的文件夹中data/ind和data/occ下是本文事先运行下列文件后得到的结果）**：

- cate2cls.py:产生各级编码类别和id的映射关系，并形成文件（在data/mapping/下）。

- loader.py：将原始数据集分为行业和职业数据集;

对于行业数据，命名为“ind\_raw.txt”；对于职业数据，命名为“occ\_raw.txt”。

- da\_ind.py和da\_occ.py：进行数据增强（**第84和85行请填上自己的百度翻译API账号和密码**）;

对于行业数据，分别命名为“ind\_da0.txt”和“ind\_da.txt”；对于职业数据，分别命名为“occ\_da0.txt”和“occ\_da.txt”。

- sample\_ind.py和sample\_occ.py:进行过采样和欠采样，并进行随机抽样，划分训练集、验证集和测试集;

对于行业数据，分别命名为“ind\_train.txt”、“ind\_val.txt”和“ind\_test.txt”；对于职业数据，分别命名为“occ\_train.txt”、“occ\_val.txt”和“occ\_ test.txt”。

- splitClass.py:将数据集按高位中位和低位进行划分。

对于行业数据，分别命名为“ind\_train34.txt”、“ind\_val34.txt”和“ind\_test34.txt”（“ind\_train56.txt”、“ind\_val56.txt”和“ind\_test56.txt”）（“ind\_train56.txt”、“ind\_val56.txt”和“ind\_test56.txt”）；对于职业数据，分别命名为“occ\_train12.txt”、“occ\_val12.txt”和“occ\_ test12.txt”（“occ\_train34.txt”、“occ\_val34.txt”和“occ\_ test34.txt”）（“occ\_train56.txt”、“occ\_val56.txt”和“occ\_ test56.txt”）。

**4、CNN卷积神经网络**

### 配置项

CNN可配置的参数如下所示，在`cnn\_model.py`中。

```python

class TCNNConfig(object):

"""CNN配置参数"""

embedding\_dim = 64 # 词向量维度

seq\_length = 600 # 序列长度

num\_filters = 128 # 卷积核数目

kernel\_size = 5 # 卷积核尺寸

hidden\_dim = 128 # 全连接层神经元

dropout\_keep\_prob = 0.5 # dropout保留比例

learning\_rate = 1e-3 # 学习率

batch\_size = 64 # 每批训练大小

num\_epochs = 10 # 总迭代轮次

print\_per\_batch = 100 # 每多少轮输出一次结果

save\_per\_batch = 10 # 每多少轮存入tensorboard

```

**训练与验证过程为：**

运行 `python train\_ind[34/56].py` 或者 `python train\_occ[12/34/56].py` 可以开始训练。

如： `python train\_ind34.py` 则对行业中间两位预测编码模型进行训练。

对于一次完整的行业和职业编码模型训练，需要进行5次训练。

> 每次修改参数进行新的训练时，可先将checkpoints下对应的occ或ind文件夹改名或保存至别处;

把tensorboard/textcnn下对应的occ或ind文件夹改名或保存至别处。

**测试过程为：**

运行 `python test\_ind.py` 或者 `python test\_occ.py` 在测试集上进行测试。

测试过程包括对各级模型的测试和融合预测编码的测试。

**预测过程为：**

为方便预测， 提供了两种模型的预测方法。

运行 `python predict\_ind.py arg\_ans1[,arg\_ans2,arg\_ans3...]` 或者 `python predict\_occ.py arg\_ans1[,arg\_ans2,arg\_ans3...]`可以对命令行输入的答案进行预测;

运行 `python predict\_all.py arg\_ans1[,arg\_ans2,arg\_ans3...]` 可以对命令行输入的文件内容进行预测，文件格式应与原数据集格式一致。如：运行 `python predict\_all.py a.xlsx` 预测结果保存在“predict.xlsx”中。