1.概述

本项目对**文本分类**任务的相关模型进行了简单的探究。模型包括**NaiveBayes**，**FastText**和**TextCNN**，其他模型有待后续增加。

2.项目结构

- **configs** 配置文件目录
- **data** 数据文件目录
- **models** 模型定义脚本目录
- **outputs** 数据输出目录，包括训练好的模型等
- **preprocess** 数据预处理脚本
- **utils** 通用工具类脚本，包括词典，切词等
- classifier.py 入口脚本

3.依赖安装

- linux

  pip install -r requirements.txt

- windows

  由于FastText没有提供Windows安装包，需要下载第三方预编译安装包进行安装：

  https://www.lfd.uci.edu/~gohlke/pythonlibs/#fasttext

  下载相应版本后，执行：pip install path_to_your_package 进行安装

  其余依赖项均可使用pip直接进行安装

4.模型简介

4.1.朴素贝叶斯

朴素贝叶斯分类器基于词袋模型，通过词袋模型我们可识别出文本中出现的词属于积极还是消极，若这个词出现在积极的词语列表中，文本的总体分数 +1，若总体分数为正，该段文本被分类为积极，反之亦然。

朴素贝叶斯通过统计词频信息学习联合概率分布$p(x,c)$，同时假设所有属性之间相互独立（这里即每个词之间相互独立）。于是联合概率可以通过词的概率乘积获得，后验概率分布可以通过贝叶斯公式求得：

$$p(c|x)=\frac{p(c)*p(x|c)}{p(x)}=\frac{p(c)*\prod_{i=1}^{d}p(x_i|c)}{p(x)} \tag{1}$$

其中，$p(c)$为类别先验分布，$p(x_i|c)$为类条件概率，即c类文档中，单词x_i出现的概率。$p(x)$为文档出现的概率，通常对于一个问题来说，$p(x)$为固定值，所以朴素贝叶斯分类器表达式为：

$$p(c|x)=p(c)*\prod_{i=1}^{d}p(x_i|c) \tag{2}$$

$p(c)$可以分别统计各个类别单词的数量获得，$p(x_i|c)$可以统计c类中单词x_i出现的频率获得。

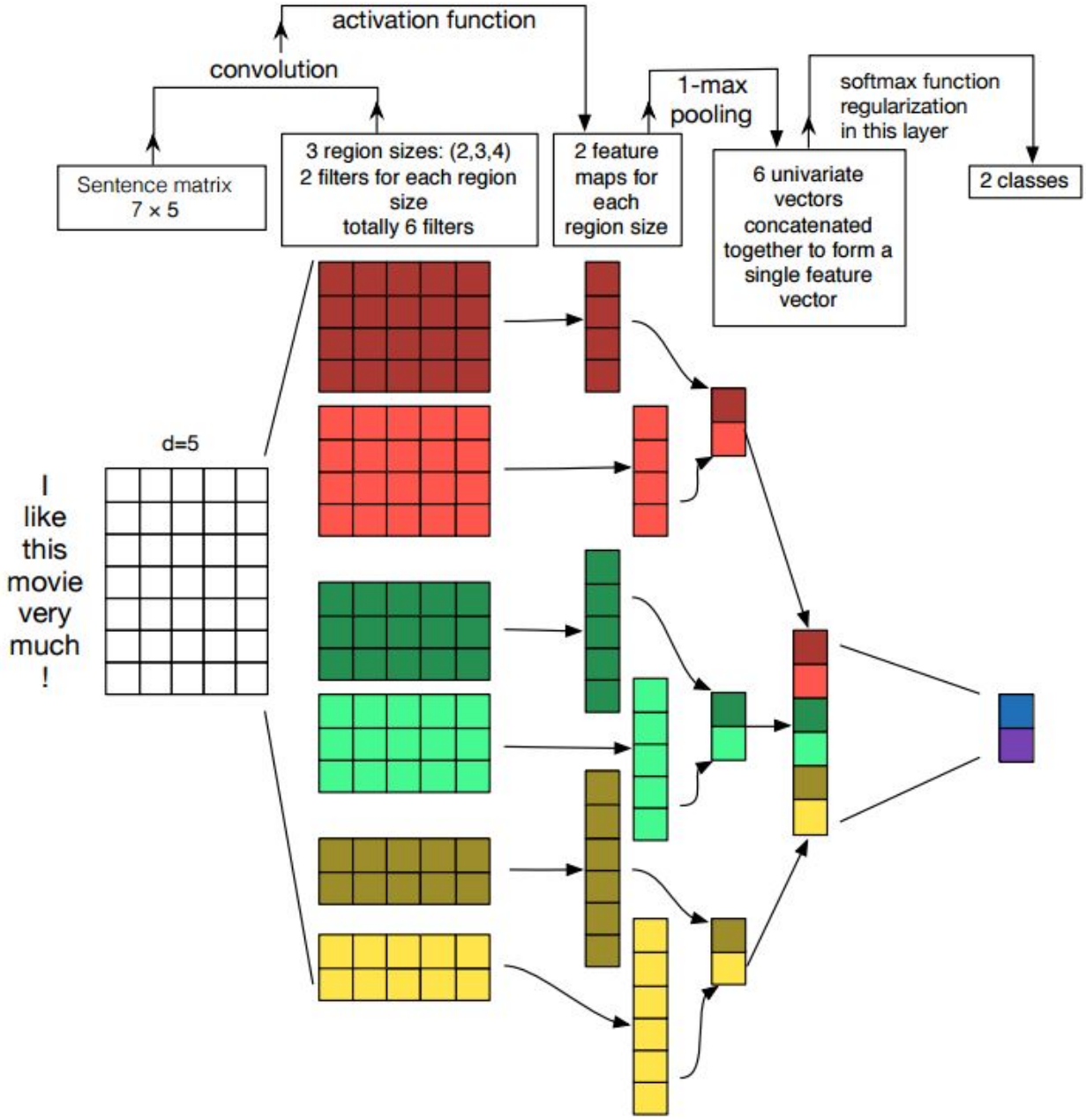
4.2.FastText

FastText的模型和CBOW模型相类似，区别在于fastText是一种有监督的模型，而CBOW属于无监督模型。CBOW通过上下文预测中间词，而fastText则是通过上下文预测标签（这个标签就是文本的类别，是训练模型之前通过人工标注等方法事先确定下来的）。FastText其实就是一个具有单隐层的softmax分类模型，不过这里作者使用了优化的分层softmax来进行分类，详解戳这里。

4.3.TextCNN

TextCNN 是利用卷积神经网络对文本进行分类的算法，由 Yoon Kim 在 “Convolutional Neural Networks for Sentence Classification” 一文 (见参考[1]) 中提出. 是2014年的算法。

模型架构如下：



分别使用不同大小的卷积核对文本矩阵进行一维卷积运算，有利于充分提取语句的n-gram特征。具体细节见textcnn_model.py中的keras实现和注释。

5.使用

5.1.数据准备

语料数据需要处理成文本格式，每行一个样本，样本格式为：label+空格+doc，即标签和文本数据中间用空格隔开，doc为无空格的文本数据，例如酒店评论数据集需要预处理为如下格式：

```
1 房间干净舒适，是出差、旅游度假的理想之选。服务业很不错。
1 总的来说可以，总是再这里住，公司客人还算满意。就是离公司超近，上楼上班下楼回家
1 房间设施难以够得上五星级，服务还不错，有送水果。
0 标准间太差 房间还不如3星的 而且设施非常陈旧。建议酒店把老的标准间从新改善。
0 服务态度极其差，前台接待好象没有受过培训，连基本的礼貌都不懂。
0 地理位置还不错，到哪里都比较方便，但是服务不象是豪生集团管理的，比较差。。
```

5.2.配置文件

配置文件采用configparser模块进行解析，包含如下字段：

- **[global]**

  global域主要定义通用参数

  - model_type 指定当前配置文件的模型类型。FastTextModel | TextCNNModel | BayesModel
  - test_split = 0.2 测试集切分比例。如果非0，则将语料数据随机抽取相应比例作为测试集。
- **[data]**
  - data域主要定义相关数据文件路径及参数
  - user_dict_path 自定义词典路径
  - corpus_path 语料数据路径，即5.1中描述的经过预处理的语料数据
  - seg_corpus_path 分割后语料:用空格分割开的词组,第一个为标签
  - sample_corpus_path 语料预采样路径
  - sample 是否进行预采样
  - vocabs_path 词典保存路径
  - model_path 模型保存路径
  - embedding_model_path 词向量模型保存路径**（仅TextCNN使用）**
  - train_dataset_path 训练集保存路径
  - test_dataset_path 测试集保存路径
- **[model]**

  model域主要定义模型参数，根据具体模型而定，详见代码注释
- **[word_embedding]**

  word_embedding域仅TextCNN模型使用，定义词向量模型参数（此处使用FastText预训练词向量），参数详情见fasttext_model.py注释

5.3.运行

classifier.py为入口脚本，需要提供`config_path`和`mode`两个额外的参数

`config_path`指定配置文件路径

`mode`指定运行模式，train | test | predict

例如，使用TextCNN模型训练，执行如下命令：

```
python classifier.py --config_path=./configs/textcnn.conf --mode=train
```

执行训练过程。

测试:

```
python classifier.py --config_path=./configs/textcnn.conf --mode=test
```

运行后输出测试集的准确率。

预测:

```
python classifier.py --config_path=./configs/textcnn.conf --mode=predict
```

运行后以交互方式输入文本，输出属于各个类别的概率。