**Archers**

**基于视角的领域情感分析**

**代码说明**

本文对Archers队基于视角的领域情感分析的整体程序代码做一个说明，算法代码分为5个部分，分别是数据爬取，汽车领域词典生成，视角抽取，句子预处理，词嵌入训练，情感分析核心算法。下面分别对这5个部分的算法作详细的阐释。

**一．数据爬取**

数据爬取主要是利用网络爬虫技术，采用Python语言中requests，urllib，BeautifulSoup等第三方库，首先利用requests或urllib请求目标网页，得到网页源码的返回，然后对目标网页进行分析，找出目标信息所在的区域，利用BeautifulSoup这款强大的HTML解析工具来获得核心信息，这就是一次爬取的过程。我们可以利用循环，来爬取多个网页的信息，达到大量数据的获取。

数据来源均是公开透明，任何人皆可获取的。

爬取的网站包括：

1.车主之家 （<http://www.16888.com/>）

2.汽车之家 （<http://www.16888.com/>）

3.搜狐汽车 （<http://auto.sohu.com/>）

4.网易汽车 （<http://auto.163.com/>）

**二．汽车领域词典生成**

**2.1处理流程**

从太平洋汽车网，汽车之家以及车主之家网站爬取的初始数据集，经过统计得到一定的规则，利用这些规则来处理生成汽车领域词典。

处理仅供参考，最终结果为final\_user.txt

**2.2类、函数介绍**

Chezhuzhijia\_extract.py, taipingyangcars\_extract.py, qichezhijia\_extract.py分别处理不同的数据

函数的Path表示输入文件的路径。

* chezhuzhijia\_extract.py：处理车主之家数据
* deal\_chezhuzhijia (file\_path,result\_path):

输入：file\_path：输入数据文件名，chezhuzhijiacars.json

输出 ：result\_path: 输出数据文件名

所有的数据均存储在自定义的path路径下。

* taipingyangcars\_extract.py：处理太平洋汽车网的数据
  + deal\_taipingyangcars(file\_path,result\_path)

输入：file\_path：输入数据文件名，taipingyangcars.json

输出 ：result\_path: 输出数据文件名，extract

所有的数据均存储在自定义的path路径下。

* qichezhijia\_extract.py：处理汽车之家数据,得到汽车之家处理后的数据在extract\_qichezhijia.txt中
* 三个网站结果，集合，去重，过滤, 得到最终纯净版视角final\_user.txt

**三．视角抽取**

本部分包含了视角抽取的总体过程，所有过程都基于网络爬虫形成的词典：

**一、基于多模匹配，利用ViewExtract.java()抽取出两部分**

一部分是单个视角，此类视角由于精确匹配，字面上一定是准确的，该文件输出

为view\_part1.txt

连续的多个视角，此类视角由于连接在一起，极有可能是同一个视角，但也有可能不是同一个视角，因此由基于最长公共子串匹配的python部分程序进一步划分, 该部分文件分为combined\_views.txt

**二、基于最长公共子串的模式匹配**

利用split\_combined\_views.py ,处理 **1** 当中**b**部分产生的combined\_views.txt，得出view\_part2.txt

将view\_part1.txt和view\_part2.txt组合成最终的generate\_view.txt

将结果进行尾部处理排序输出进入后期的规则处理并形成最终的generate\_view.txt

**三、运行步骤**

1. 用ideaIntelliJ打开，选择maven方式,修改split\_combined\_views.py中的base\_path 为该文件当前所在的绝对路径，编译运行/src/test/java/org/viewextract/ViewExtract.java，得到generate\_view.txt
2. 在linuxterminal中运行：

cd 到 ViewExtract.java 所在目录， 运行/bin/bash view\_review.sh, 进行排序去重等操作

**四．数据预处理**

1. **整体说明**

　　predeal\_data.py文件主要实现将训练文本和测试文本转化成具有一定格

　　式，用于后期情感分析任务的处理：

　　训练文本转化的格式为: id\tview\pol\ttext

　　测试文本转化为：id\tview\ttext

　　(id表示记录编号，view表示视角,pol表示情感，text表示相应的句子，\t表示分隔符)

1. **函数介绍**

* **get\_new\_path (train\_or\_test,isSecond,isSeg):**

根据不同的处理方式设置不同的存储路径，

Train\_or\_test:为0表示是训练集

isSecond:1表示是复赛训练集

isSeg:是否拆分，拆分的意思为对于一个句子中出现多个视角，按语义将文本拆分给不同的视角或者直接把整个句子作为视角的文本，此框架处理采用不需要划分方式。

* **predeal\_data(t\_path,t\_view\_path,t\_data\_result\_path,train\_or\_test,isSecond,isSeg):**

t\_path:表示文本train 或者 test，

t\_view\_path：表示抽取的视角文件

t\_data\_result\_path:结果文件

train\_or\_test:表示处理的是训练集还是测试集，0表示训练集，1表示测试集

isSecond：表示处理的是初赛的数据还是复赛的数据，1表示复赛

isSeg:表示要不要按视角对整个句子进行划分，1表示要划分

* **trans\_space(raw\_path,del\_path,train\_or\_test,isSecond,isSeg):**

采取jieba分词，如果视角中出现空格，要做特殊处理，转化成^号，后面相应的句子中出现的此视角中的空格也为^。

Raw\_path:表示待处理文件

Del\_path:表示结果文件

Train\_or\_path：处理的是训练集还是测试集，0表示训练集，1为测试集

isSecond：表示是否是复赛数据集，0表示初赛数据集

isSeg：表示是否需要切分，0表示不需要，1表示需要

* **filter\_view(text\_path,result\_path,train\_or\_test,isSecond,isSeg)**

根据统计得到的规则，对文本进行视角过滤，过滤不符合规则的视角。

1. **运行说明**

对于训练集，只需运行：predeal\_data以及trans\_space函数，得到的结果即为训练集结果(采取初赛和复赛训练集结果拼接而成)

* 运行初赛训练数据

predeal\_data('Train.csv', 'Label.csv', 'train\_id\_view\_pol\_t.txt', 0,0,False)  
trans\_space('train\_id\_view\_pol\_t.txt', 'train\_id\_view\_pol\_trans\_t.txt', 0,0,False)

* 运行复赛训练数据

predeal\_data('TrainSecond.csv','LabelSecond.csv','train\_second\_id\_view\_pol\_t.txt', 0,1,False)

trans\_space('train\_second\_id\_view\_pol\_t.txt','train\_second\_id\_view\_pol\_trans\_t.txt', 0,1,False)

* 手动拼接train\_id\_view\_pol\_trans\_t.txt和

train\_second\_id\_view\_pol\_trans\_t.txt文件作为下步算法的输入

* 运行测试集数据

输入的generate\_view.txt为视角抽取结果

predeal\_data('Test.csv','generate\_view.txt','test\_second\_id\_view\_text\_t.txt', 1,0,False)

trans\_space('test\_second\_id\_view\_text\_t.txt','test\_second\_id\_view\_trans\_t.txt', 1,0,False)

filter\_view('test\_second\_id\_view\_trans\_t.txt','test\_second\_id\_view\_t.csv',1,0,False)

得到的‘test\_second\_id\_view\_t.csv’用于下步情感分析处理

**五．词嵌入训练**

为了不断增加词向量模型的精确度和表现力，我们不仅采用了大规模的互联网上的综合文字数据，如搜狗实验室的互联网全网新闻数据，包含各个类别的新闻，大小超过2G，这有效地训练了非汽车领域词的词向量精确度，为了提高汽车领域的情感分析效果，我们又通过下载和自己爬取等方式，总共收集到了将近500M的汽车语料，总共2.73GB的语料，采用C语言编写的word2vec训练程序进行训练成128维的词向量模型，训练批处理程序train.sh如下：

time ./word2vec -train Train\_segmented.txt -output vectors128.bin -cbow 5 -size 112 -window 8 -negative 25 -hs 0 -sample 1e-4 -threads 20 -binary 1 -iter 15

最后我们直接将vectors128.bin利用到核心算法中去。

**六．深度学习核心算法BFGRU**

核心代码分为预处理，数据变换和模型训练与预测三个部分。下面分三个部分分别进行介绍。

**一、预处理部分**

程序文件名： getComment.py

本部分将上一部分处理出来的特定格式的train和test集文件，预处理成适合模型输入的文件。本部分的算法说明见算法说明文档。

核心函数介绍如下：

1. genTrainData\_Multi(maxlen=200)

本函数生成一个句子中某个视角某个位置的左右部分。部分左右句子的处理核心代码如下：

l\_sentence = map(lambda x: x.decode('utf8'), tokenize\_text([LEFT])) # 分词

l\_sentence = Tools.drop\_stopwords(l\_sentence) # 去除停用词

l\_sentence = Tools.find\_view\_unfinded(l\_sentence, views) # 找出分错词的汽车品牌，重新整合句子

r\_sentence = map(lambda x: x.decode('utf8'), tokenize\_text([RIGHT])) # 分词

r\_sentence = Tools.drop\_stopwords(r\_sentence) # 去除停用词

r\_sentence = Tools.find\_view\_unfinded(r\_sentence, views) # 找出分错词的汽车品牌，重新整合句子

以下部分是对句子长度的一个规整，即如果句子长度超过了最大长度，我们提取maxlen-3个关键性的词语，按顺序连接起来，即相当于去除了一些较为普通，无关的词语，有利于最大程度地减小句子信息损失。

if len(l\_sentence) >= maxlen:

tagwords = jieba.analyse.extract\_tags(" ".join(l\_sentence), topK=maxlen-3)

if VIEW not in tagwords:

tagwords.append(VIEW)

l\_sentence = Tools.sentence\_keep\_key(l\_sentence, tagwords) # 保留关键词，使整个句子的长度小于等于 max\_sentence\_length

1. genTestData\_Multi(maxlen=200)

本函数的处理关键部分与上一个函数相同，只是处理的是测试集的数据。

**二、数据变换**

程序文件名： DataOperation.py

程序部分核心方法及代码如下：

def **load\_data\_bi\_word2vec**(maxlen, words\_keep, validation\_portion, embedding\_dim, ma): # TD模式的数据加载，将词替换为词向量，也是实际使用的

l\_trainX = [map(lambda x: model[x] if x in model else ones, map(lambda x: x.decode('utf8'), sen)) for sen in l\_trainset]

l\_testX = [map(lambda x: model[x] if x in model else ones, map(lambda x: x.decode('utf8'), sen)) for sen in l\_testset]

r\_trainX = [map(lambda x: model[x] if x in model else ones, map(lambda x: x.decode('utf8'), sen)) for sen in r\_trainset]

r\_testX = [map(lambda x: model[x] if x in model else ones, map(lambda x: x.decode('utf8'), sen)) for sen in r\_testset]

# 上一部分代码表示将词替换成词向量，即词x替换成model[x]，为固定维数的向量，这里我们取128维

l\_trainX = pad\_sequence(l\_trainX, maxlen, embedding\_dim)

l\_testX = pad\_sequence(l\_testX, maxlen, embedding\_dim)

r\_trainX = pad\_sequence(r\_trainX, maxlen, embedding\_dim)

r\_testX = pad\_sequence(r\_testX, maxlen, embedding\_dim)

# 上一部分代码完成长度规整操作，确保长度不超过maxlen

return ret\_L\_trainX, ret\_R\_trainX, ret\_labels, l\_testX, r\_testX

# 最后返回训练集左部分的数据矩阵，右部分的数据矩阵，以及标签，以及测试集的左右部分数据矩阵

**三、模型训练与预测**

程序文件名： word2vecbigru.py

由上一步我们已经得到数据的张量表现形式，然后我们搭建基于视角的情感分析的深度学习模型，将训练集的情感Label转换成one-hot形式，如pos对应[0, 1, 0]，neu对应[1, 0, 0]等。然后搭建视角聚焦型的双向GRU模型。

位于train方法中的模型搭建核心代码如下：

# 搭建左边网络

lnet = tflearn.input\_data([None, maxlen, embedding\_dim])

# 搭建右边网络

rnet = tflearn.input\_data([None, maxlen, embedding\_dim])

# 左边网络设置为动态GRU单元，dropout值设置为0.8，只返回最后状态

lnet = tflearn.gru(lnet, embedding\_dim, dropout=0.8, return\_seq=False, dynamic=True)

# 右边网络设置为动态GRU单元，dropout值设置为0.8，只返回最后状态

rnet = tflearn.gru(rnet, embedding\_dim, dropout=0.8, return\_seq=False, dynamic=True)

# 将左右网络的输出融合为一个向量

net = tflearn.layers.merge\_outputs([lnet, rnet])

# 将该向量用softmax进行分类，分为3类，分别代表3类情感

net = tflearn.fully\_connected(net, 3, activation='softmax')

# 采用adam优化器，学习率0.001，并采用类交叉商作为损失函数，对模型进行训练

net = tflearn.regression(net, optimizer='adam', learning\_rate=0.001,

loss='categorical\_crossentropy')

# 最终建模，训练

model = tflearn.DNN(net, tensorboard\_verbose=0)

最后，对测试集数据进行预测，预测值做一定的后处理，处理成适合比赛要求的提交数据，在后处理阶段，由于在一个句子中同一个视角出现多次，我们会拆分成不同的左右部分，所以对于一个视角V和一个句子S，假设V在S中出现C次，那么句子S就会被分成C种不同的左右部分，然后相当于视角V和句子S重复出现C次，最后我们需要将V在S中的情感综合考虑，总结出来，这时我们采用的是**加和打分算法**，就是说，每次V,S的重复出现都会得到一个情感分类预测的原始结果，比如[0.2, 0.7, 0.1]，这就代表在句子中某个位置出现的视角V，情感更加倾向于pos，如果在另一个位置情感分类结果为[0.9, 0.05, 0.05]，那么说明在这个位置更加倾向于neu，这时我们综合一下，将各类分数相加，得到[1.1, 0.75, 0.15]，最终得到V在S中的综合情感为neu。这就是我们加和打分算法的思想。

做了这样一个合并之后，我们再将没有出现视角的句子的ID写入到结果文件中，就得到了需要提交的格式的一个结果文件。

**四、运行说明**

1.运行训练程序，只需运行word2vecbigru.py即可。

2.完整的一套运行流程，需要在getComment.py中，保证以下语句

genTrainData\_Multi(200)

genTestData\_Multi(200)

不被注释的情况下，运行getComment.py.

然后在word2vecbigru.py中确保以下语句：

model.fit([l\_trainX, r\_trainX], trainY, validation\_set=0.1, show\_metric=True,

batch\_size=32)

model.save('MODELS/E\_W2V\_GRU\_TC{}\_{}.dy'.format(embedding\_dim, maxlen))

不被注释的情况下，运行word2vecbigru.py.