**综合课程设计报告**

**姓名**：吴嘉津

**学号**：2016020906006

**摘 要**

随着计算机性能的提高和深度学习的算法的研究加深，如今已经可以使用深度学习的方法研究自然语言处理的问题，而自然语言处理当中情感分析是很重要的一部分。本文基于深度神经网络的理论，以电影影评为数据集，重点研究并实现了二分类的情感分析，同时完成了在线分析功能。

**关键词**：情感分析，深度神经网络，在线判别

目录

[第一章 绪论 3](#_Toc6952591)

[1.1研究工作的背景与意义 3](#_Toc6952592)

[1.2 情感分析的国内外研究现状与应用范畴 3](#_Toc6952593)

[1.3 本文的主要贡献与创新 4](#_Toc6952594)

[1.4 本论文的结构安排 4](#_Toc6952595)

[第二章 理论分析 5](#_Toc6952596)

[2.1深度神经网络(DNN)结构和向前传播过程 5](#_Toc6952597)

[2.2深度神经网络（DNN）的反向传播 6](#_Toc6952598)

[2.3 批量梯度下降法(BGD)、随机梯度下降法(SGD)、小批量梯度下降法(MSGD) 7](#_Toc6952599)

[2.4损失函数 7](#_Toc6952600)

[2.5激活函数 7](#_Toc6952601)

[2.6 TF-IDF提取词库 9](#_Toc6952602)

[2.7文本的表示 10](#_Toc6952603)

[2.8浏览器-服务器(B/S)架构 10](#_Toc6952604)

[2.9 Flask框架原理 11](#_Toc6952605)

[第三章 代码实现 12](#_Toc6952606)

[3.1 模型训练步骤 12](#_Toc6952607)

[3.2 文本处理主要函数说明 12](#_Toc6952608)

[3.2.1 deal\_data(data)函数 12](#_Toc6952609)

[3.2.2 handleOrgData(ifileName,ofileName)函数 13](#_Toc6952610)

[3.2.3 readTreatedData(ifileName)函数 13](#_Toc6952611)

[3.2.4 readLabel(ifileName)函数 14](#_Toc6952612)

[3.2.5 spileDataSet(orgData,label,splitRate,train=True)函数 14](#_Toc6952613)

[3.2.6 buildWordLib(fileName)函数 15](#_Toc6952614)

[3.2.7 buildOneHotTest(orgCombineData,wordLib)函数 15](#_Toc6952615)

[3.2.8 trainStep(train,test,preData,wordLib,net)函数 16](#_Toc6952616)

[3.2.9 tf\_idf(infile,outfile,num)函数 16](#_Toc6952617)

[3.3 神经网络主要函数说明 16](#_Toc6952618)

[3.3.1 均方误差损失函数 16](#_Toc6952619)

[3.3.2 交叉熵损失函数 17](#_Toc6952620)

[3.3.3 神经网络结构定义和初始化 17](#_Toc6952621)

[3.3.4向前传播 18](#_Toc6952622)

[3.3.5反向传播 19](#_Toc6952623)

[3.3.6 随机梯度下降更新参数 19](#_Toc6952624)

[3.3.7 取整函数 20](#_Toc6952625)

[3.3.8 准确值计算 20](#_Toc6952626)

[3.3.9 损失值计算 21](#_Toc6952627)

[3.3.10 保存网络参数 21](#_Toc6952628)

[3.3.11 读取网络参数 22](#_Toc6952629)

[3.4 Flask框架实现B/S架构 23](#_Toc6952630)

[3.4.1 服务器的启动 23](#_Toc6952631)

[3.4.2 在线判别业务 23](#_Toc6952632)

[第四章 结果分析 24](#_Toc6952633)

[4.1 模型训练最好结果 24](#_Toc6952634)

[4.2 模型训练当中出现的现象 25](#_Toc6952635)

[第五章 全文总结与展望 26](#_Toc6952636)

[5.1全文总结 26](#_Toc6952637)

[5.2后续工作展望 26](#_Toc6952638)

# 第一章 绪论

## 1.1研究工作的背景与意义

如今，互联网十分发达，人们通过互联网发表自己的信息表达自己的情感，其中，发表的信息当中，文本是最为常见的形式，是一种十分丰富的遍布在网络的资源，如果能有效利用这些文本资源，可以获取大量有用的信息资源。因此，自然语言处理已经成为新时代的一个热门领域

其中，情感分析是自然语言处理的组成部分，研究情感分析，对构建自然语言处理有很大作用。

## 1.2 情感分析的国内外研究现状与应用范畴

按照文本处理粒度的不同，情感分析分为以下三种情况：

一是词语的情感分析：其中又有基于语料库、基于词典、基于词向量三种方法。

二是语句的情感分析：语句作为词汇的聚合，是整篇情感分析的基础。分为基于知识库、基于网络、基于语料库三种分析方法。

三是篇章的情感分析：篇章的情感分析需要结合上下文进行情感的理解。

目前，文本情感分析的主要方法有基于规则或无监督学习方法和基于有监督的学习。但基于规则或无监督学习的成果比较少，主要还是以基于有监督的学习为主。

情感分析已经有不少知名的商用系统。如基于产品评论的Google Shopping，为用户提供商品检索和比价服务；联合国开发的Global Pulse，用于对全球进行情感检测；Stock Sonar提供民众对与股票的情感信息，为投资者提供参考；还有微软的对话机器人“小冰”。

## 1.3 本文的主要贡献与创新

本文以电影评论的数据集为基础，训练了电影评论的情感分析模型，并开发了在线评测网站，我们可以将此模型获取人们对于电影的满意度，从而采取不同的市场决策。

## 1.4 本论文的结构安排

本论文的章节结构安排如下：

1）理论分析

2）代码实现

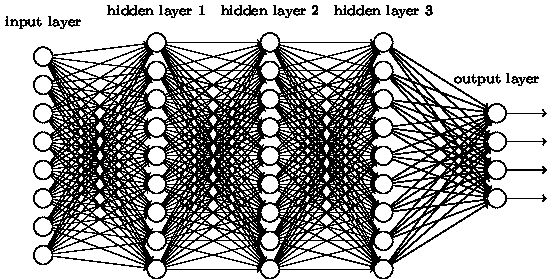
3）结果分析

4）全文总结与展望

# 第二章 理论分析

## 2.1深度神经网络(DNN)结构和向前传播过程

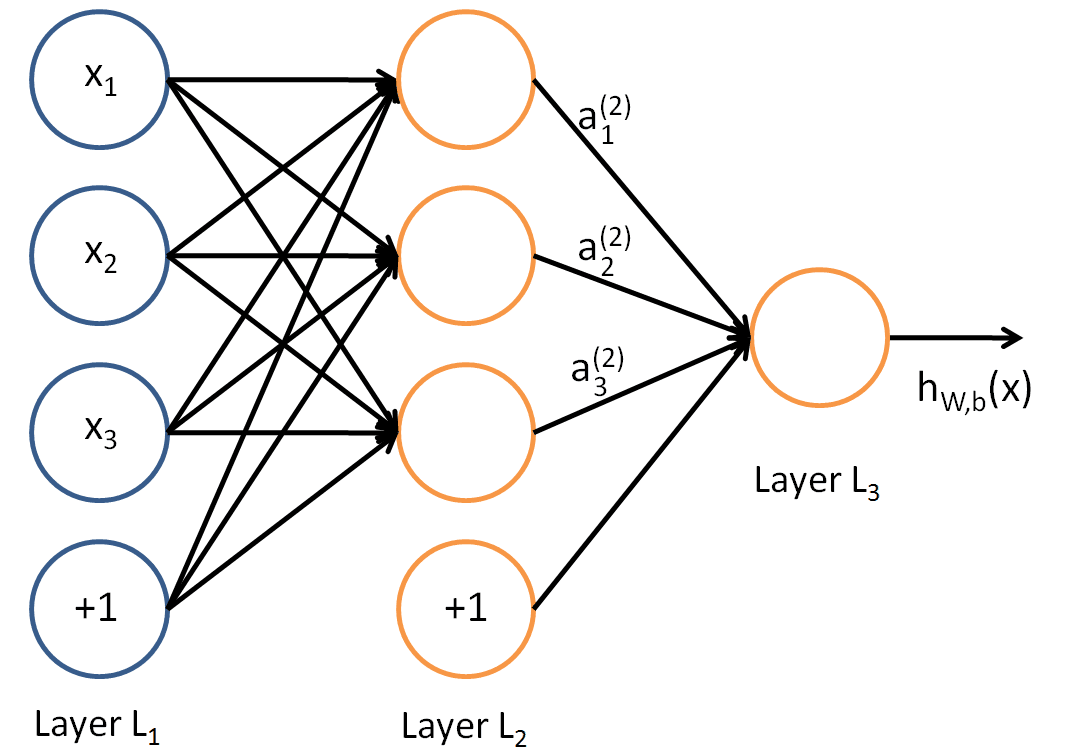
深度神经网络DNN内部的神经网络层可以分为输入层、隐含层、输出层三类，第一层是输入层，隐含层在中间，最后一层是输出层。



神经网络的层与层之间是全连接的，因此，第层的神经元一定与第层的神经元相连。

以下为例，说明DNN向前传播的过程。

首先，输入层的输入为,当输入向前传播，由于每一个第二层的神经元都与第一层相连，权重分别为（表示为当前层的第个神经元与下一层第个神经元的参数），因此第二个神经元的输入为,又因为隐含层和输出层的每一个神经元都有激活函数，设为,因此第二层第一个神经元的输出为。



将上面的推导一般化，我们可以得到DNN第层的输出（使用矩阵表示）：

因此，我们可以得到完整的DNN向前传播的算法。

输入：总层数,所有隐藏层和输出层对应的权重矩阵,偏移量矩阵,输入向量

输出：输出层的输出

算法：

令

for() do

## 2.2深度神经网络（DNN）的反向传播

对于DNN的损失函数，我们选择均方差来进行度量

其中为权重矩阵，为偏移量，为输入向量，为输出向量, 为真实值

用于反向传播的四个公式用矩阵形式表示则为：

其中为权重矩阵，为偏移量，为激活前的值，为激活后的输出值，为真实值

## 2.3 批量梯度下降法(BGD)、随机梯度下降法(SGD)、小批量梯度下降法(MSGD)

假设损失函数为,其中为所求的参数

对于每个输入和其输出,可以计算其损失函数

若每次取n个训练样本，则其损失函数为

若每次训练都将全部样本进行训练，则为批量梯度下降法(BGD)

若每次训练都只取一组样本进行训练，则为随机梯度下降法(SGD)

若每次训练取部分样本进行训练，多次训练将全部样本训练完成，则为小批量梯度下降法(MSGD)

批量梯度下降法由于每次都将训练样本全部一起训练，易于搜索出全局最优解，但计算量较大，当数据量太大的时候，批量梯度下降法不可行。因此，当数据量太大的时候，可以考虑选择随机梯度下降法或者小批量梯度下降法。

## 2.4损失函数

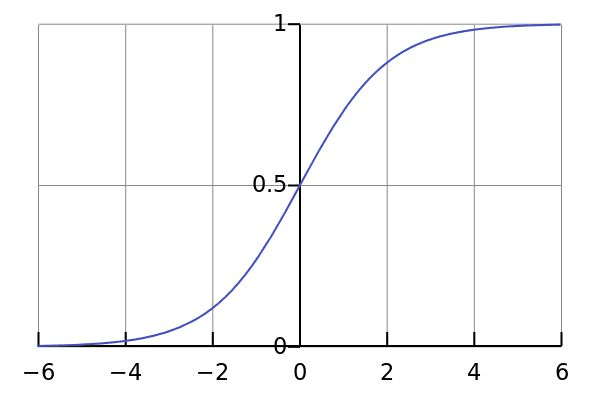
交叉熵：

交叉熵一般用于分类，交叉熵越大，预测值与真实值差别越大

均方误差(MSE)：一般用于回归

## 2.5激活函数

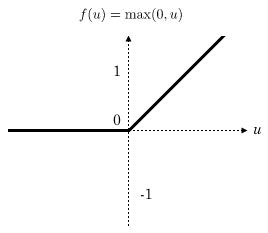
sigmod函数：



softmax函数：一般用于多分类，即将所有类别的输出归一化处理，概率值最大的一项为预测值。softmax函数的另一个特性时对于采用交叉熵作为损失函数的，损失函数对于softmax输入值求偏导会特别简单。

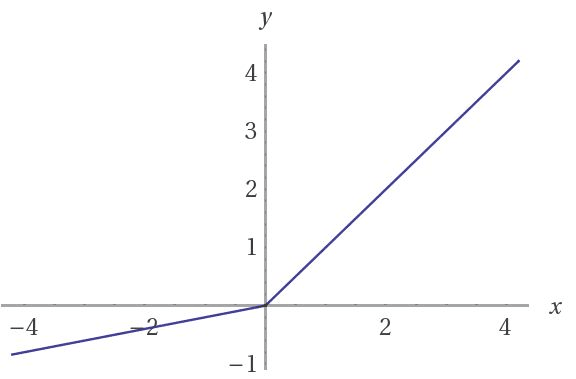
relu函数：

relu函数为线性整流函数，一般用于隐层激活函数和回归输出层激活函数



leakyrely函数：

与relu函数类似，但时，，一般取趋于0的数，一般为0.01



## 2.6 TF-IDF提取词库

TF-IDF是“词频-逆文本频率”，由两部分组成，一部分是TF，一部分是IDF。

TF指的是词频，用以体现词语出现的频率，IDF是逆文本频率，用以体现一个词语的重要性。

若一个词汇在不同文本出现的频率高，即TF高，则这个词的重要性会减小，即IDF会减小。下面则为IDF值的定量计算。

其中表示语料库的文本总数，表示语料库中包含词的文本总数。

则一个词的TF-IDF值为

对语料库中所有词计算其TF-IDF值，并将其进行排序，选择值较高的部分，可以保持词库的有效性的同时，降低词库的词汇量，以降低神经网络的计算量。

## 2.7文本的表示

对于一个输入到深度神经网络的文本，假设词库的词汇量为A，我们可以使用一个N维的向量来表示文本

其中

## 2.8浏览器-服务器(B/S)架构

B/S架构是对C/S架构的一种改进，用户的工作界面由浏览器来实现，极少部分的事务逻辑在浏览器实现，主要事务逻辑在服务器端实现。以下为B/S架构的优缺点。

优点：

1）通用架构，开发成本低

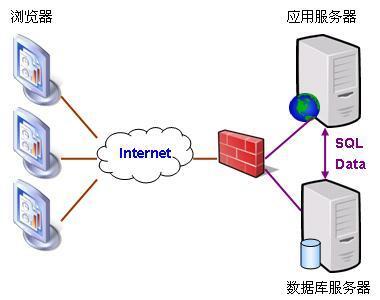
2）不需要安装客户端，因此客户端不需要升级

3）界面修改十分方便

缺点：  
1）耗流量，每次需要加载全部内容

2）无法实现个性化的客户端

3）客户端难以实现安全控制



## 2.9 Flask框架原理

服务器程序：接收、整理客户端发送的请求。

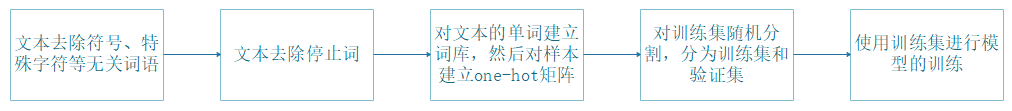
应用程序：处理服务器程序传递过来的请求。

WSGI是python web的开发标准，约定了服务器程序和应用程序遵循的规范，规定了各自使用的接口和功能，以便二者能够配合。

Flask是一个基于python开发并且依赖jinja2模板和Werkzeug WSGI服务的一个微型框架。对于Werkzeug，Flask用于接收http请求并对请求进行处理，然后触发Flask框架，开发人员基于Flask框架提供的功能对请求进行相应处理并返回给用户

# 第三章 代码实现

## 3.1 模型训练步骤



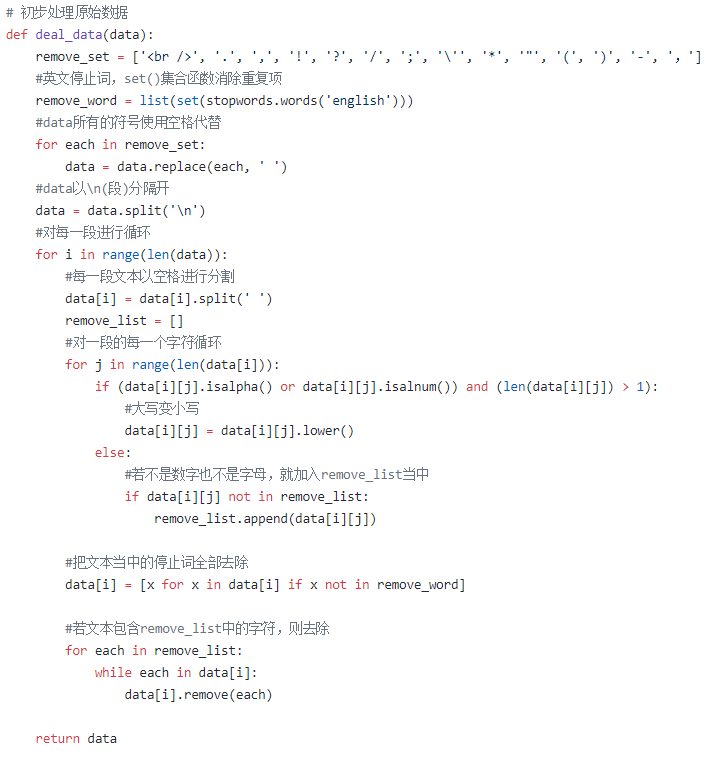
## 3.2 文本处理主要函数说明

### 3.2.1 deal\_data(data)函数

输入：字符串形式的文本

输出：形如[[‘a’,’b’,’c’,’d’],[‘aa’,’bb’],[‘cc’,’dd’],……,[‘ff’,’hello’]]的二维列表，其中，第一维代表不同的评论，第二维则是评论中不同的词

作用：将文本进行分割，存放在列表当中，便于后续的处理，同时去除文本当中的标点符号和特殊字符，并且将与文义无关的停止词去除

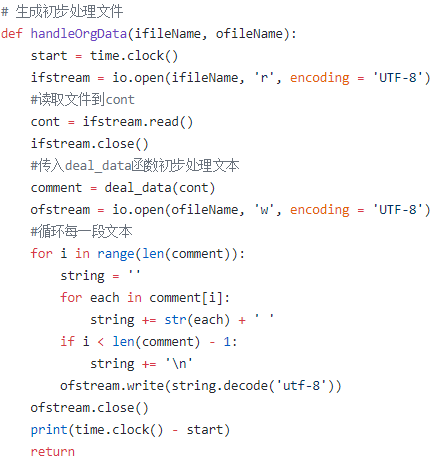


### 3.2.2 handleOrgData(ifileName,ofileName)函数

输入：代处理文本的txt文件和处理后存放结果的txt文件名

输出：无

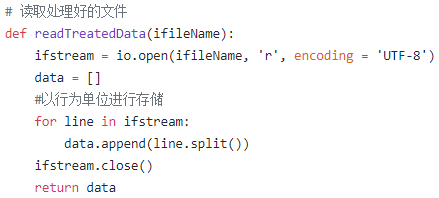
作用：读入需要处理的文本（以txt形式存放），调用deal\_data函数进行标点符号、特殊字符、停止词的相关处理，然后将处理结果写入到新的txt文件当中。



### 3.2.3 readTreatedData(ifileName)函数

输入：需要读入的txt文件，该文件存放着评论文本

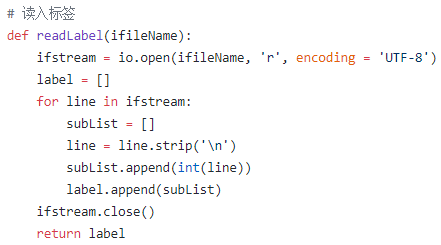
输出：二维列表，第一维表示不同的评论，第二维表示评论中不同的词



### 3.2.4 readLabel(ifileName)函数

输入：需要读入的txt文件，该文件存放着评论的分类标签

输出：一维列表，各个元素均为0或1的分类标签

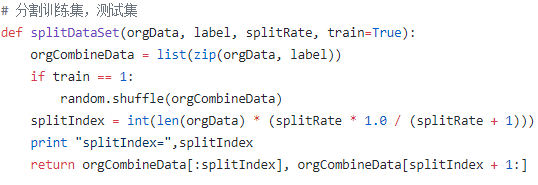


### 3.2.5 spileDataSet(orgData,label,splitRate,train=True)函数

输入：训练集，标签，划分比例，是否用于训练模型

输出：训练集，验证集

作用：orgData代表着训练集的数据，按照splitRate的划分比例对训练集进行划分，一部分作为训练集，一部分作为验证集，并且将文本和对应分类标签使用zip函数进行结合。若train为True，则将训练集的数据打乱顺序。

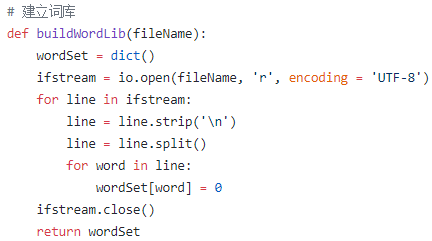


### 3.2.6 buildWordLib(fileName)函数

输入：存放评论文本的txt文件

输出：字典，其中键值为不同的词，对应的取值均设为0

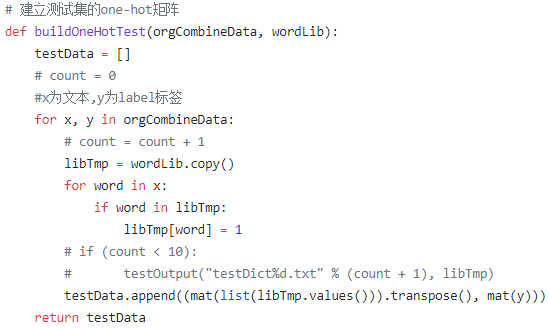
作用：将文本的词进行去重处理，并将所有词汇加入到字典当中



### 3.2.7 buildOneHotTest(orgCombineData,wordLib)函数

输入：文本和对应标签已经结合后的数据

输出：形如[(matrix([0],[0],[1],[1]),matrix([1])), (matrix([0],[1],[0],[0]),matrix([1])),……]，其中matrix([0],[0],[1],[1])表示词库有4个词，0或1代表该评论是否含有该词。matrix([1])则为对应评论的分类标签。



### 3.2.8 trainStep(train,test,preData,wordLib,net)函数

输入：训练集，验证集，预测集，词库，神经网络

输出：模型训练时间

作用：调用神经网络的SGD方法进行模型的训练

### 3.2.9 tf\_idf(infile,outfile,num)函数

输入：需要读取的文本的txt文件，需要写入处理后的数据的文件名，词库大小

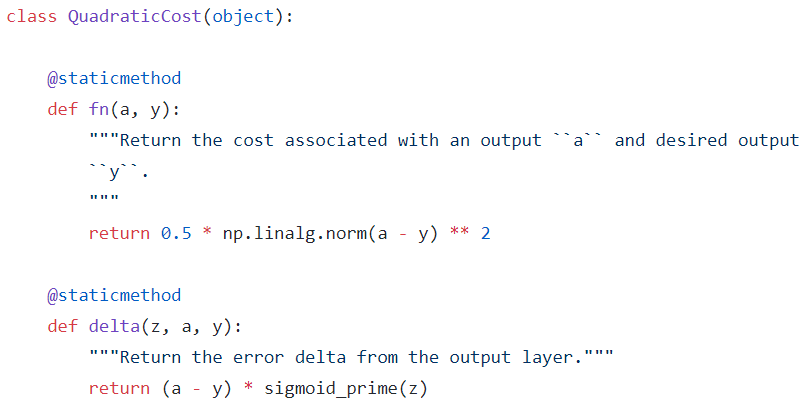
输出：无

作用：读取infile指定的文本文件，对里面的全部词汇计算tf-idf值，并排序，根据num的值取排序后的前num的词作为词库。按照已经获得的词库，对文本进行处理，仅保留出现在词库中的词，并且把处理后的文本写入outfile指定的txt文件当中。

## 3.3 神经网络主要函数说明

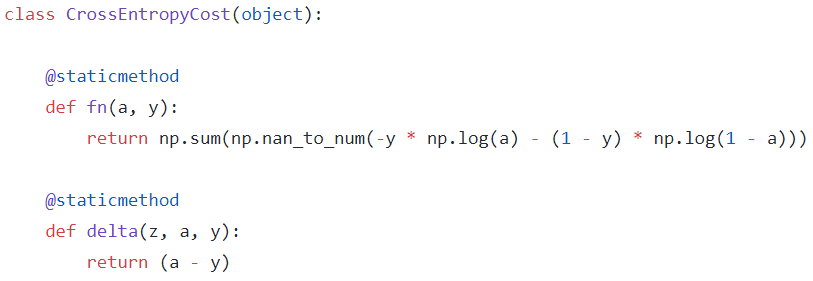
### 3.3.1 均方误差损失函数

a为输出层预测值，y为真实值，z为激活前的值。其中fn函数计算损失值，delta函数用于计算最后一层的，以用于后续的反向传播。



### 3.3.2 交叉熵损失函数

a为输出层预测值，y为真实值，z为激活前的值。其中fn函数计算损失值，delta函数用于计算最后一层的，以用于后续的反向传播。



### 3.3.3 神经网络结构定义和初始化

深度神经网络具有以下参数

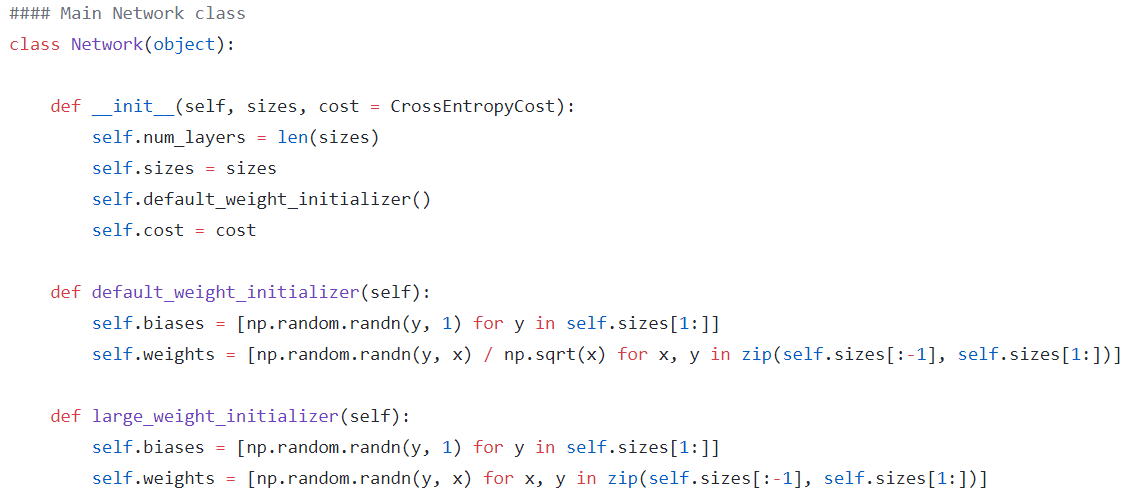
1)num\_layer：深度神经网络的层数

2)sizes：形如[wordSetLen,64,64,64,labelNum],其中wordSetLen为输入神经元个数，即词库的大小；三个64则表示有三个隐含层，每个隐含层有64个神经元；labelNum则表示标签由几个元素表示。

3)cost：使用的损失函数的名称

4)weights：神经网络的权重

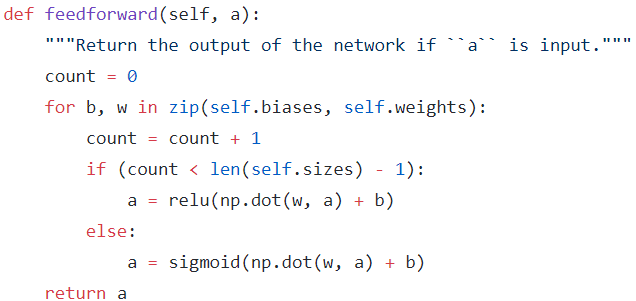
5)biases：神经网络的偏移量



default\_weight\_initializer和large\_weight\_initializer为神经网络权重weights和偏移量biases的两种初始化方法。

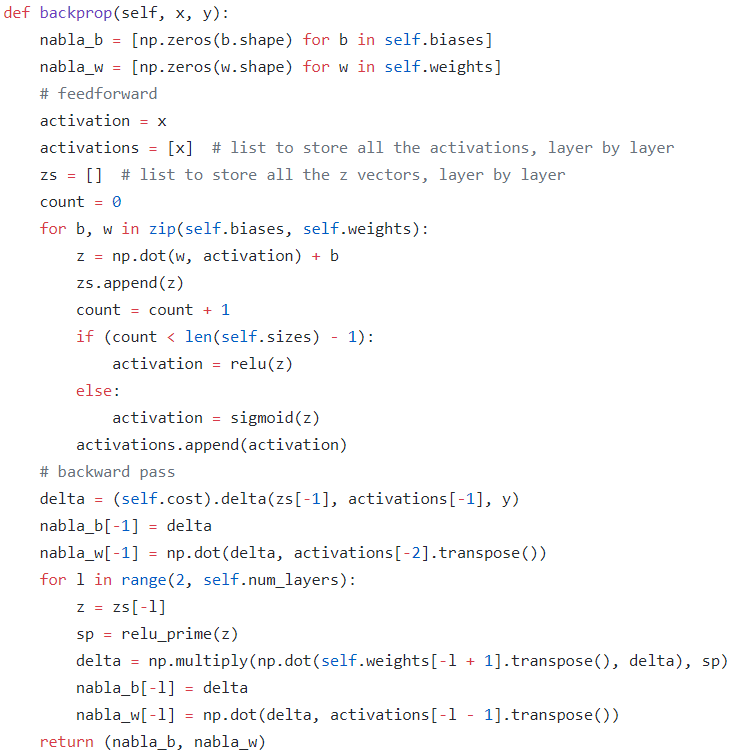
### 3.3.4向前传播

a为神经网络的输入值，隐含层的激活函数使用relu函数，输出层的激活函数使用sigmod函数，向前传播的过程与第二章的理论分析中向前传播的算法相同。



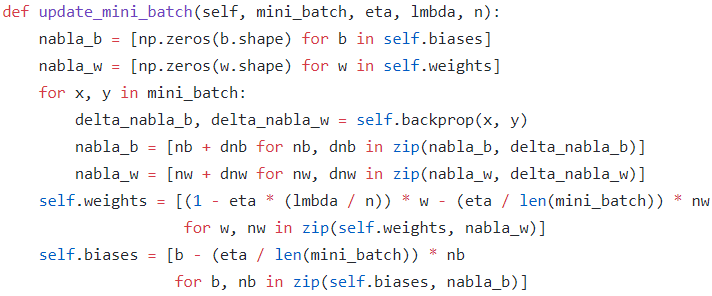
### 3.3.5反向传播

x为输入值，y为真实值，先按照向前传播的算法计算出输出层的预测值。先计算出输出层的，再根据和计算出输出层的和,然后根据递推式反向计算出前一层的的值。如此递推，即可算出每一层的和



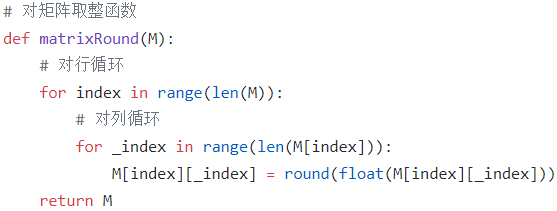
### 3.3.6 随机梯度下降更新参数

对于每一批样本，按照公式对神经网络的参数进行更新操作，权重W的计算使用了L2正则项的导数



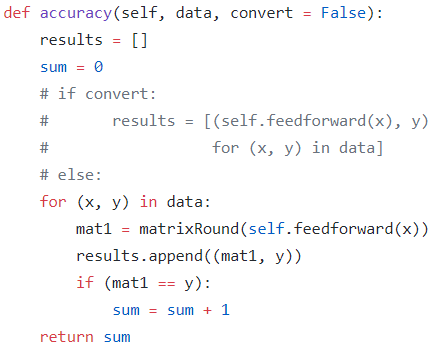
### 3.3.7 取整函数

由于输出层的激活函数使用了sigmod函数，范围在0~1之间，因此，我们使用round()函数将0~0.5之间的值归为0，0.5~1之间的值归为1，即将输出进行了分类。



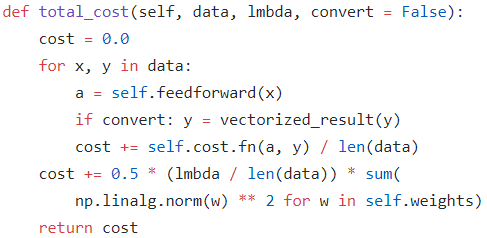
### 3.3.8 准确值计算

函数的输入参数data中包含了输出层的预测值x和对应的真实值y，调用取整函数matrixRound根据输出值x进行了分类，再和标签进行比较，最终得出这一批数据有多少个数据预测值和真实值一样，即有多少个数据分类正确。



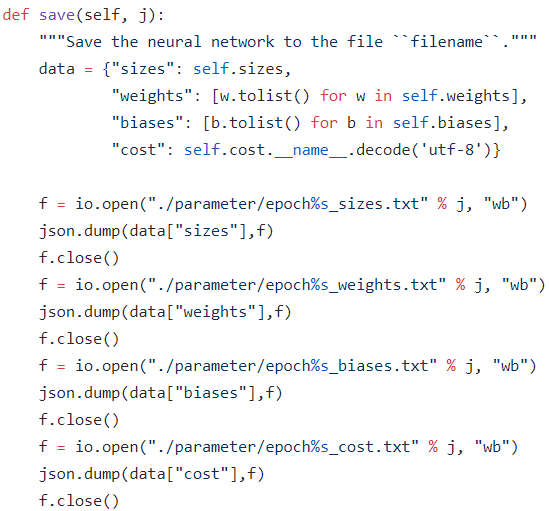
### 3.3.9 损失值计算

对于一批样本和标签，将样本输入神经网络，向前传播得到预测值，再根据以下公式计算这一批样本的损失值



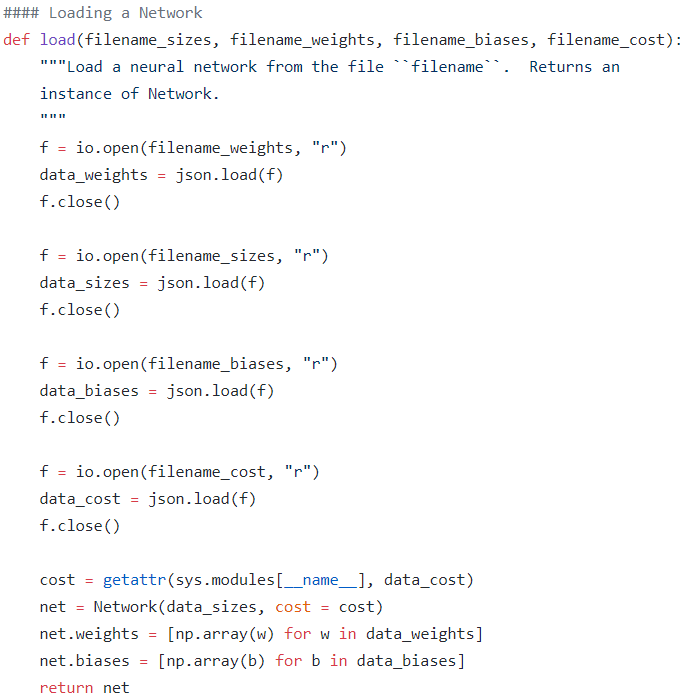
### 3.3.10 保存网络参数

将神经网络的权重weights、偏移量biases、结构大小sizes、损失函数类型cost以json的格式存放到txt文件当中。



### 3.3.11 读取网络参数

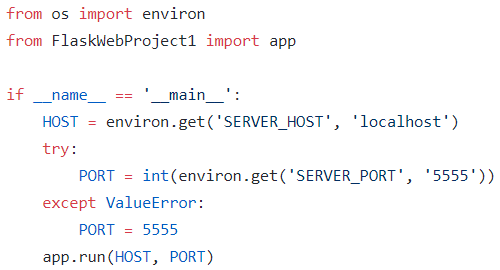
将神经网络的权重weights、偏移量biases、结构大小sizes、损失函数类型cost以json的格式从txt文件当中读出。



## 3.4 Flask框架实现B/S架构

### 3.4.1 服务器的启动

指定应用程序的运行端口为5555，并且将此runserver.py文件运行即可启动服务器



### 3.4.2 在线判别业务

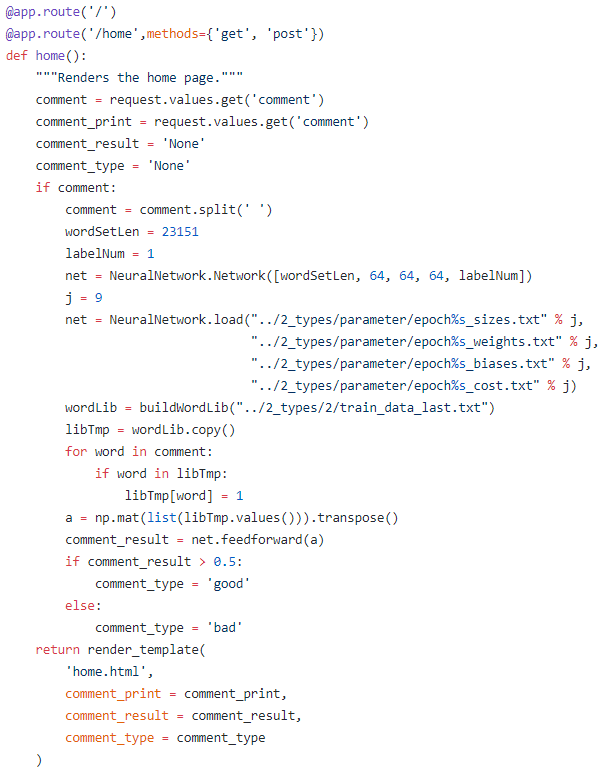
1）在线判别网页提供GET和POST方法，其中用户输入的评论以POST方法传入后端的comment变量中，为字符串类型。

2）按照前面训练的深度神经网络结构，使用NeuralNetwork.NetWork函数初始化一个深度神经网络。

3）对于初始化后的神经网络，使用NeuralNetwork.load函数读入已保存的参数。

4）对于评论comment，根据已建立的词库，构造one-hot矩阵，使用net.feedforward方法进行向前传播并根据结果分类。

5）后端得到评论的输出值comment\_result和预测类别comment\_type后，将这些值传到前端html进行渲染并返回给用户。



# 第四章 结果分析

## 4.1 模型训练最好结果

神经网络参数如下：

输入：one-hot(23151维)

架构：23151(输入)\*64\*64\*64(隐层)\*1(输出)

批(batch)：128

学习率：0.01

迭代次数:40次

训练集：验证集=23500 : 499

准确率最高的一次训练如下:验证集准确率为88.17%(在程序运行记录.txt当中可以进行查看)

Epoch 9 training now

Epoch 9 training complete

Cost on training data: 0.236970892156

Accuracy on training data: 21326 / 23500

Cost on evaluation data: 0.311158956432

Accuracy on evaluation data: 440 / 499

## 4.2 模型训练当中出现的现象

以下为每次迭代的准确率

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 迭代次数 | 训练集损失 | 训练集准确率 | 验证集损失 | 验证集准确率 |
| 1 | 0.58 | 11806/23500 | 0.68 | 251/499 |
| 5 | 0.33 | 20307/23500 | 0.36 | 427/499 |
| 9 | 0.23 | 21326/23500 | 0.31 | 440/499 |
| 13 | 0.18 | 21918/23500 | 0.30 | 436/499 |
| 17 | 0.14 | 22331/23500 | 0.32 | 435/499 |
| 21 | 0.12 | 22430/23500 | 0.36 | 434/499 |
| 25 | 0.08 | 23019/23500 | 0.37 | 434/499 |
| 29 | 0.06 | 23183/23500 | 0.41 | 435/499 |
| 33 | 0.04 | 23334/23500 | 0.46 | 434/499 |
| 37 | 0.03 | 23362/23500 | 0.52 | 430/499 |
| 39 | 0.02 | 23429/23500 | 0.53 | 431/499 |

由上表我们可以知道，随着迭代次数的增加，训练集准确率逐步增加，最终训练集达到99.70%的正确率，但验证集的正确率确是先增后减。

由此可见，深度神经网络学习功能的确强大，在第9次迭代的时候验证集就已经达到最高准确率，但并不是迭代次数越多神经网络就越好，因为迭代次数过大会导致过拟合的现象。说明深度神经网络的迭代次数和真实数据集的准确率类似一个开口向下的函数，会达到一个最大值。

# 第五章 全文总结与展望

## 5.1全文总结

本文以深度神经网络为基础，并在文本处理时使用了tf-idf方法建立词库，使用随机梯度下降法进行神经网络的训练，构建了电影评论的情感分析模型，并搭建了在线的C/S架构的电影情感分析网站。

## 5.2后续工作展望

本文仍然有很多可以改善的地方，具体如下：

1）如果文本的标签不只是二分类，而是多类判别，那么该如何处理？

2）能否使用更加完善的神经网络如CNN，RNN，LSTM来构建模型？

3）本文one-hot矩阵维度很大，能否使用词向量的方法进行降维以降低计算量？