毕业设计报告

I. 问题的定义：

1. 项目概述：

解决的问题：

将新闻归类，可以给用户提供用户关注的分类的相关新闻，从而提高用户的体验，此次项目要解决的问题是通过机器学习的方式将新闻进行分类

涉及的领域：

自然语言处理，神经网络，机器学习，NLTK，Word2Vec

出发点：

使用机器学习的方式将新闻分类，替代人工的方式，从而减少网站运营人员的工作量，通过机器学习的方式还有以下优点： 1.可以更精准，2.可以减少误差（不同人对文档分类可能不同）

数据集：

www.qwone.com/%7Ejason/20Newsgroups/20news-19997.tar.gz
里面大约有 20000 条新闻，比较均衡地分成了 20 类

2. 问题陈述：

需要解决的问题：

将 20000 条新闻，通过机器学习的方式将其分类

策略（1）：

第一步：对数据进行清洗，使用 nltk 将新闻切换成句子，去掉句子中的特殊字符，去掉停用词（如 me，i，he，she 等），然后将句子进行切分成一个个单词
第二步：对得到的新闻的单词列表进行统计，提取出词频最大的 300 个词
第三步：使用 google 训练好的模型：GoogleNews-vectors-negative300.bin 或者使用 glove 模型，因为上述训练的模型使用的语料较大，通过这个模型得到的词向量更好一些
第四步：通过上面得到的词向量，词频最好的 300 个词的词向量，进行叠加取平均得到新闻对应的文档向量
第五步：将新闻的分类进行 oneHot 编码得到新闻的标签，然后将 20000 条数据按比例分成训练集和测试集，使用神经网络进行训练得到一个最好的模型

策略（2）：

第一步：同策略 1
第二步：同策略 1
第三步：通过 doc2Vec，对文档进行训练，直接得到新闻的文档向量
第四步：同策略 1
第五步：同策略 1

期望结果：

通过对新闻数据的训练可以得到一个最佳模型，对于任何一个文档，可以将此文档划分到这个 20 分类中最能代表这篇新闻的分类

3. 评价指标：

模型性能评价指标：

测试集的准确率作为模型性能评价的标准，准确率越高，模型性能越好

合理性：

只有测试集的准确率，才能体现模型的泛化能力，才能更准确的预测模型未见过的数据，当然这也是在数据集,分类数据相对平衡的情况下，如果分类数据不平衡,就不能使用准确率来衡量一个模型的好坏,这时候可以使用 F-score，R2 评分等去评价一个模型的好坏

II. 分析

1. 数据的探索：

使用了 20000 条新闻数据， 均衡地分成了 20 类

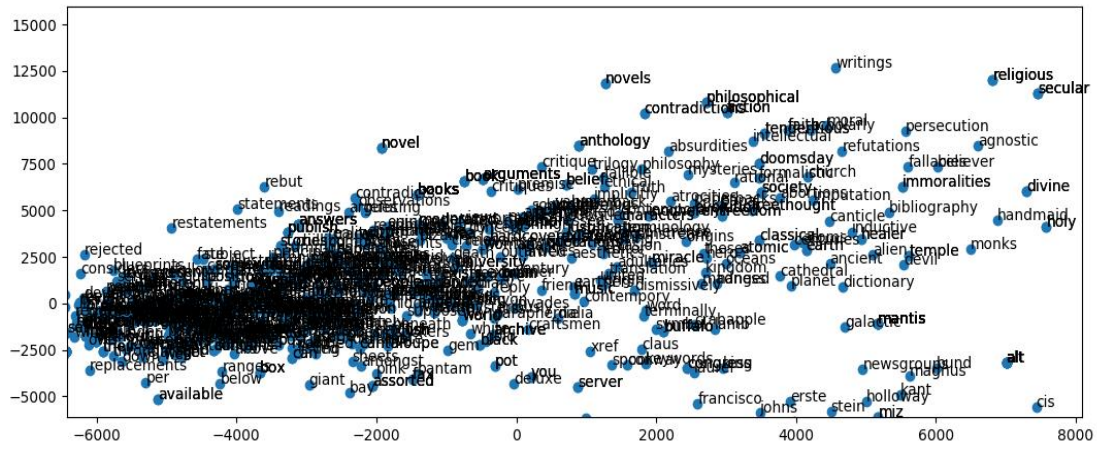
每条新闻含有成千上万的词句，首先对新闻进行了分句分词，发现数据有大小写问题，接着对所有单词统一转换成小写，新闻中还还有一些特殊符号如标点符号，以及数字，将这些词全部去掉，然后通过 Counter 对新闻的单词进行统计，发现很多词频较大的词，都是没有含义的指示代词如（me，he，she），通过使用 ntlk stopwords 将这些词过滤掉，最后得到以下的统计数据：

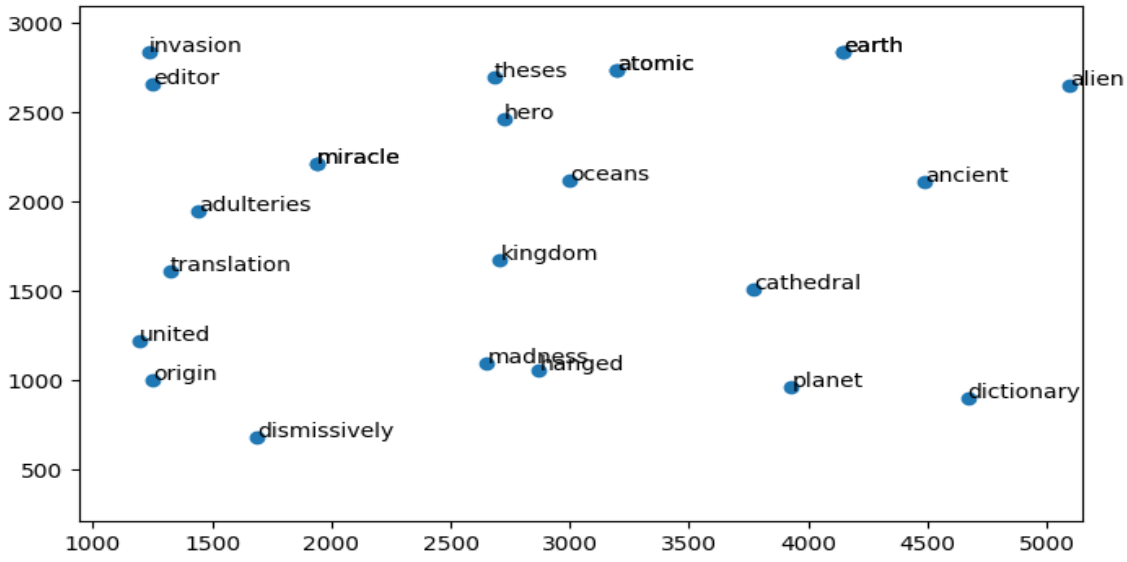
[('atheism', 18), ('s', 15), ('god', 15), ('books', 13), ('alt', 10), ('edu', 10), ('atheist', 10), ('press', 8), ('bible', 8), ('mantis', 7), ('uk', 6), ('write', 6), ('religion', 6), ('people', 6), ('swinburne', 6), ('fish', 6), ('existence', 6), ('history', 6), ('cmu', 5), ('answers', 5), ('co', 5), ('prometheus', 5), ('american', 5), ('humanism', 5), ('isbn', 5), ('arguments', 5), ('resources', 5), ('news', 4), ('addresses', 4), ('germany', 4), ('mathew', 4), ('r', 4), ('london', 4), ('book', 4), ('publish', 4), ('usa', 4), ('new', 4), ('p', 4), ('darwin', 4), ('archive', 4), ('world', 4), ('telephone', 4), ('christianity', 4),

.....]

2. 探索性可视化：

通过 word2Vec 对每一个词得到一个 300 维的向量，通过 PCA 降维，变成一个二维的向量，使用 mataplib 展示，第一幅图是文档所有的词，第二幅图是放大的图，可以发现 hero，kingdom 意义相近的词距离较近

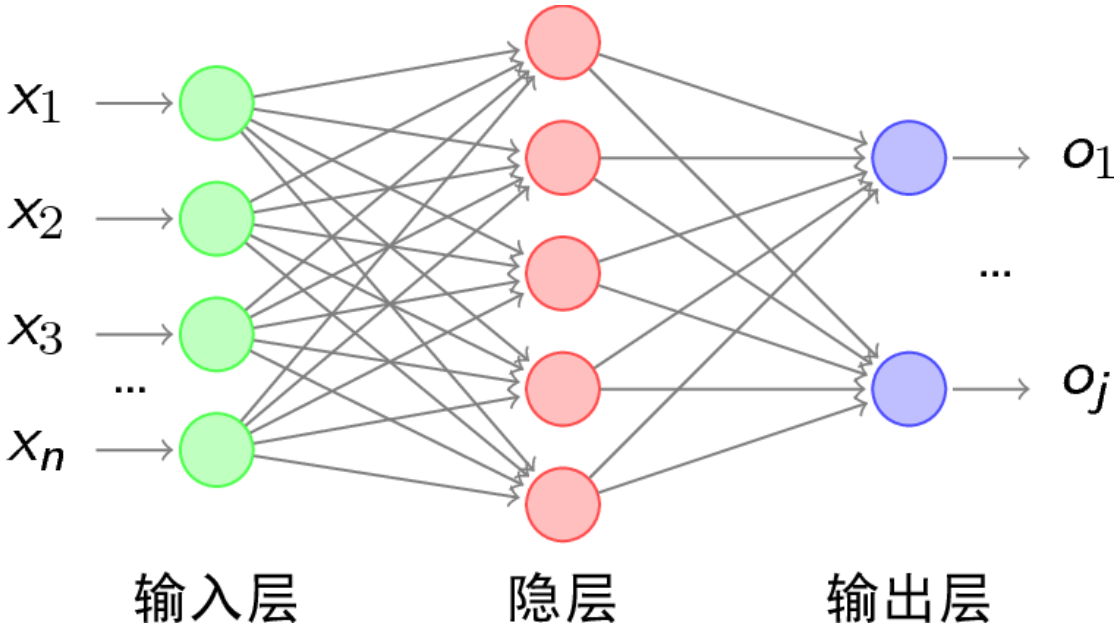




3. 算法和技术：

使用的算法： BP 反向传播算法，word2Vec，随机梯度下降

BP 反向传播算法：



为了得到权向量，我们通过最小化损失函数来不断调整权向量。此方法也适用于此处求解权向量，首先我们需要定义损失函数，由于网络的输出层有多个输出结点，我们需要将输出层每个输出结点的差值平方求和。于是得到每一个训练样例的损失函数为

$$E(\vec{w}) = \frac{1}{2} \sum_{k \in outputs} (t_k - o_k)^2$$

根据实例的输入，从前向后依次计算，得到输出层每个单元的输出。然后从输出层开始反向计算每一层的每个单元的误差项。

对于输出层的每个单元 k，计算它的误差项：

$$\delta_k = o_k(1 - o_k)(t_k - o_k)$$

对于网络中每个隐藏单元 h，计算它的误差项：

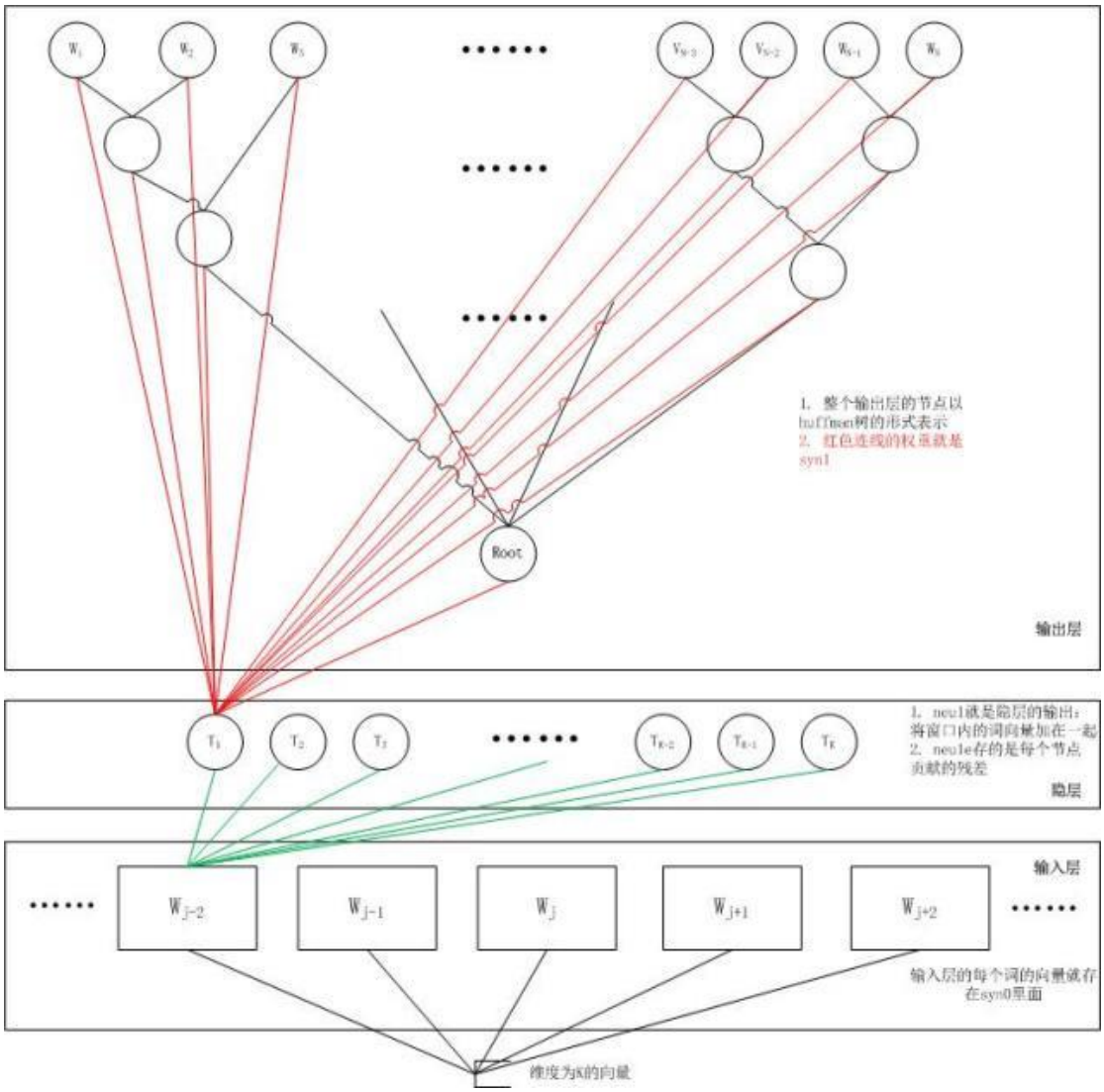
$$\delta_h = o_h(1 - o_h) \sum_{k \in outputs} w_{kh}\delta_h$$

更新每个权值：

$$w_{ji} = w_{ji} + \eta\delta_j x_{ji}$$

$$\Delta w_{ji} = \eta\delta_j x_{ji}$$ 被称为权值更新法则

Word2Vec 算法：



输入层读入窗口内的词，将它们的向量（K 维，初始随机）加和在一起，形成隐藏层 K 个节点。输出层是一个巨大的二叉树，叶节点代表语料里所有的词（语料含有 V 个独立的词，则二叉树有|V|个叶节点）。而这整颗二叉树构建的算法就是 Huffman 树。这样，对于叶节点的每一个词，就会有一个全局唯一的编码，形如”010011”。我们可以记左子树为 1，右子树为 0。接下来，隐层的每一个节点都会跟二叉树的内节点有连边，于是对于二叉树的每一个内节点都会有 K 条连边，每条边上也会有权值。

在训练阶段，当给定一个上下文，要预测后面的词(Wn)的时候（word2vec 的 CBOW 和 Skip-gram 都不是预测后面的词，都是在中间的词上做文章，但是本文这么写并不影响理解），实际上我们知道要的是哪个词(Wn)，而 Wn 是肯定存在于二叉树的叶子节点的，因此它必然有一个二进制编号，如”010011”，那么接下来我们就从二叉树的根节点一个个地去便利，而这里的目标就是预测这个词的二进制编号的每一位！即对于给定的上下文，我们的目标是使得预测词的二进制编码概率最大。形象地说，我们希望在根节点，词向量和与根节点相连经过 logistic 计算得到的概率尽量接近 0（即预测目标是 bit=1）；在第二层，希望其 bit 是 1，即概率尽量接近 1……这么一直下去，我们把一路上计算得到的概率相乘，即得到目标词 Wn 在当前网络下的概率(P(Wn))，那

么对于当前这个 sample 的残差就是 1-P(Wn)。于是就可以 SGD 优化各种权值了

参数说明：

word_count = 300
使用多少个最高词频的词，作为文档向量
keep_pro = 0.2
为了防止过拟合会丢弃一些数据特征
learning_rate = 0.03
梯度下降算法的学习率，过大容易错过最优解，过小收敛的过慢
epochs = 5000
训练的轮数，因为我设置了提前终止，所以我这个参数设置的比较大，提前终止的规则是 100 轮里验证集的准确率没有上升，或者验证集的 loss 没有下降，会提前终止训练
batch_size = 256
每次传入的数据大小

合理性：

使用最高词频的词能够比较好的代表一个文档，然后通过 word2Vec 得到每个词的词向量，进行叠加求平均得到文档向量，作为神经网络的输入，文档归类通过 oneHot 得到标签，作为神经网络的输出，通过随机梯度下降的方法不断的更新权重，使得 loss 不断下贱，准确率不断提高，最后得到一个较好的模型

4. 基准模型：

基准结果：测试集的准确率，测试集的准确率更能体现一个模型的泛化能力
基准模型阈值：85%（20000 份新闻，有 17000 预测正确）

性能评估标准：
准确率 < 60% 性能较差
准确率 > 60% <85% 性能中等
准确率 > 85% 性能较好

III. 方法

1. 数据预处理：

第一步：使用 nltk 进行分句分词
第二步：去掉特殊字符和数字
第三步：去掉停用词(如 he，she 等)

第四步：将所有单词转换成小写

2. 执行过程：

策略一：

(1) 执行过程中一开始使用的是新闻数据作为语料训练 word2Vec 模型，训练得到的测试准确率 62%，语料太小，训练出来的模型效果不佳
(2) 使用 google 训练好的 word2Vec 模型，准确率在 76%
(3) 取词频较高的 300 个词，准确率提高到 83%
(4) 过滤停用词之后准确率提高到 89%

策略二：

(1) Doc2vec 训练 300 轮，准确率 74%
(2) Doc2vec 训练 500 轮，准确率 80%
(3) Doc2vec 训练 2000 轮，准确率 83%
(4) Doc2vec 训练 5000 轮，准确率 85%

3. 完善：

针对取词频最多单词的个数代表文档的向量的个数分别取 50,100,200,300,400,500
对模型也做了不同的选择，google，glove
训练模型也做了优化，使用了 keras 的回调函数进行提前终止

IV. 结果

1. 模型的评价与验证：

最终模型的选择,选择测试集准确率最高的模型

最好的模型准确率达到，已经是较好的模型，最终的模型是通过很多次试验，筛选出最好的参数，模型对于新闻分类稳健可靠，训练数据输入一些微小变化不会太大影响结果，这个模型测试数据 6000 份新闻，有 90%分类正确，所以这个模型是可信的

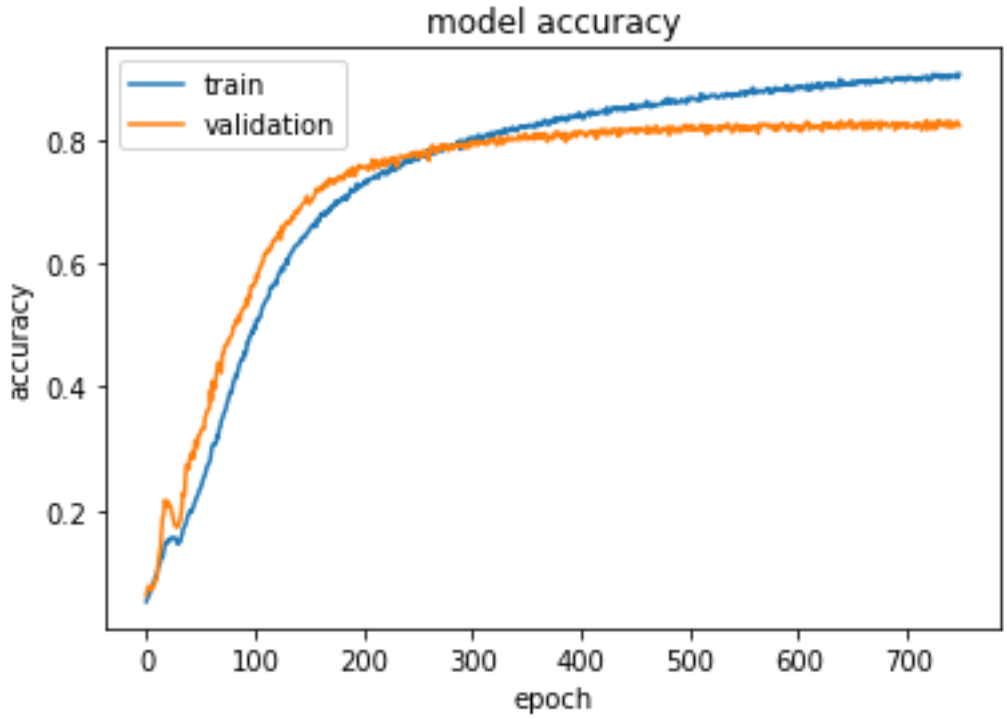
2. 合理性分析：

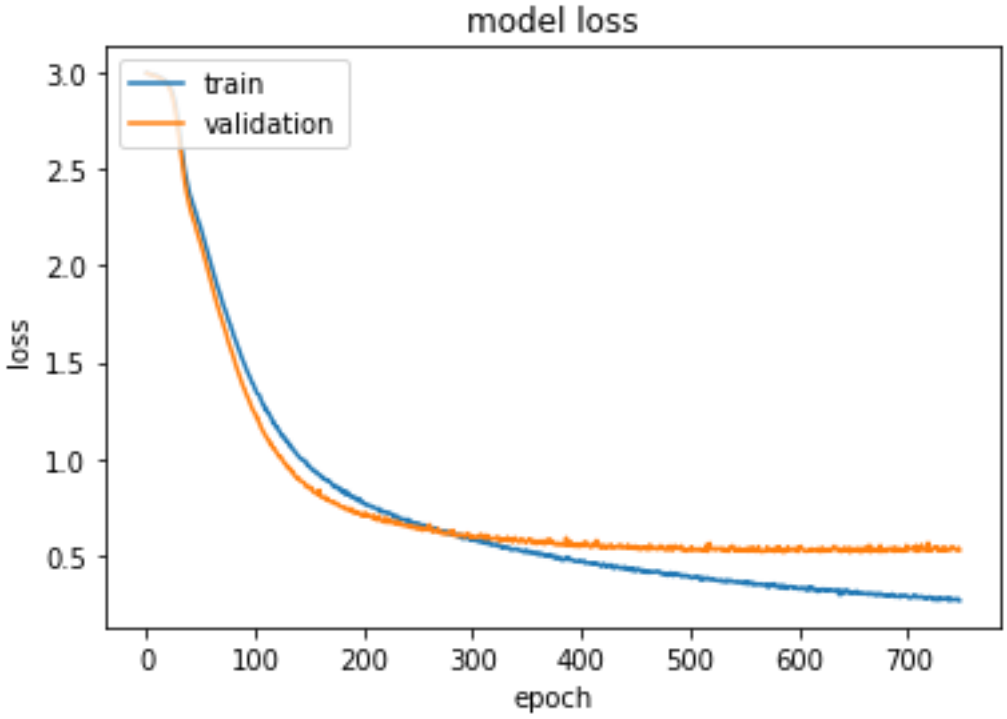
最终模型比基准模型表现的更好，准确率达到比 85%高一些，比基准模型预测的准确率更高一些，通过这个模型，可以通过机器学习的方式解决新闻分类问题，真正解决了实际的问题

Ⅴ. 项目结论

1. 结果可视化

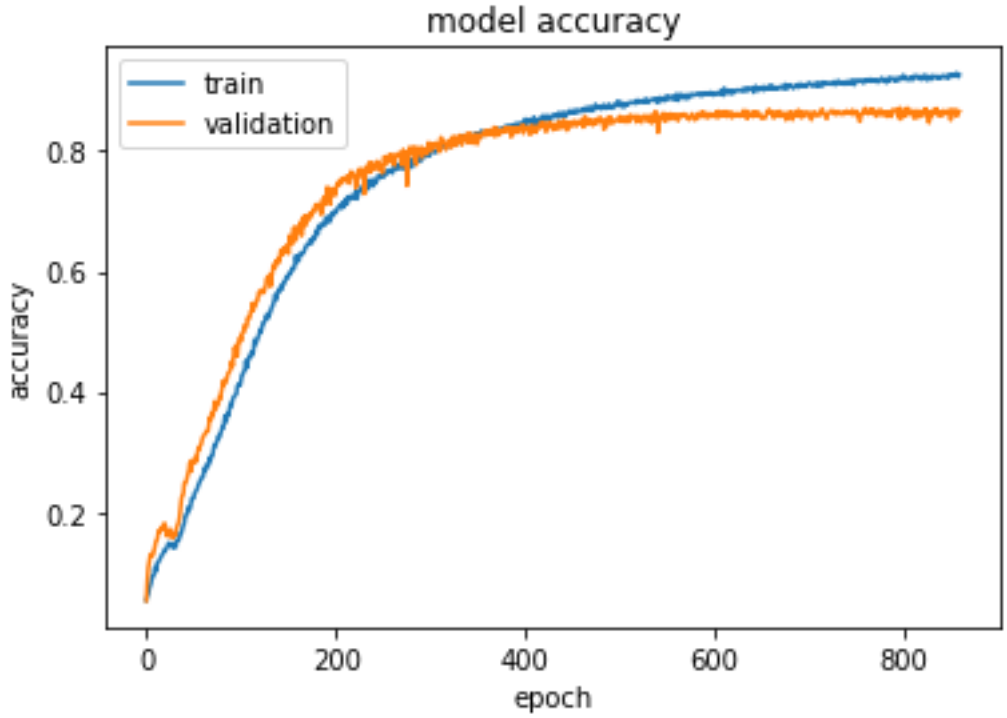
wordCount = 50

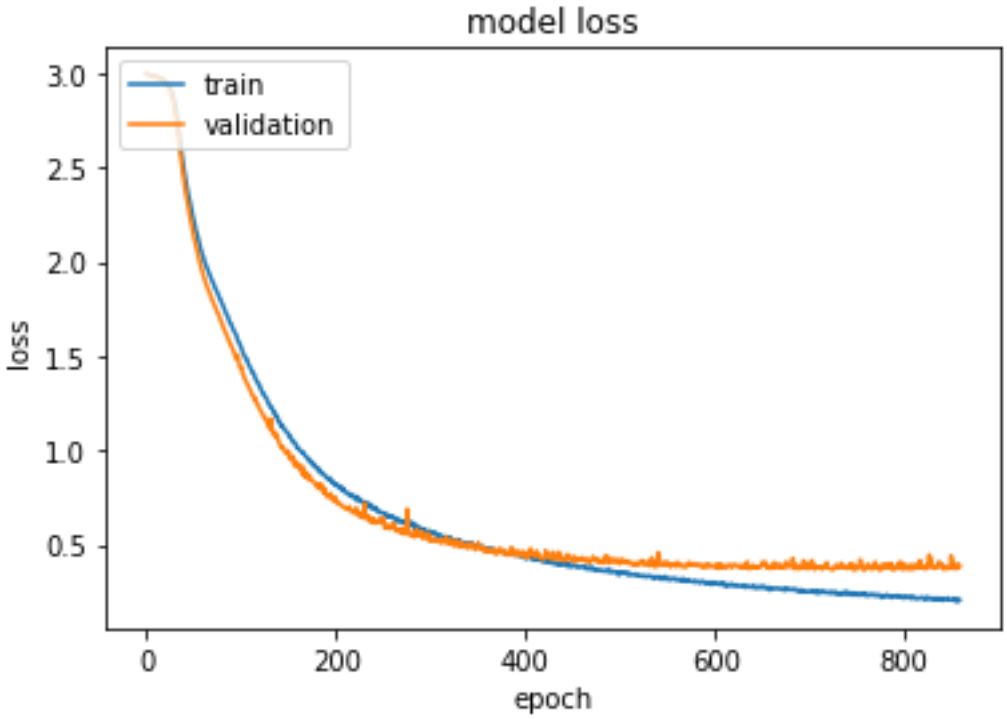




test loss: 0.524331120173 test accuracy 0.835

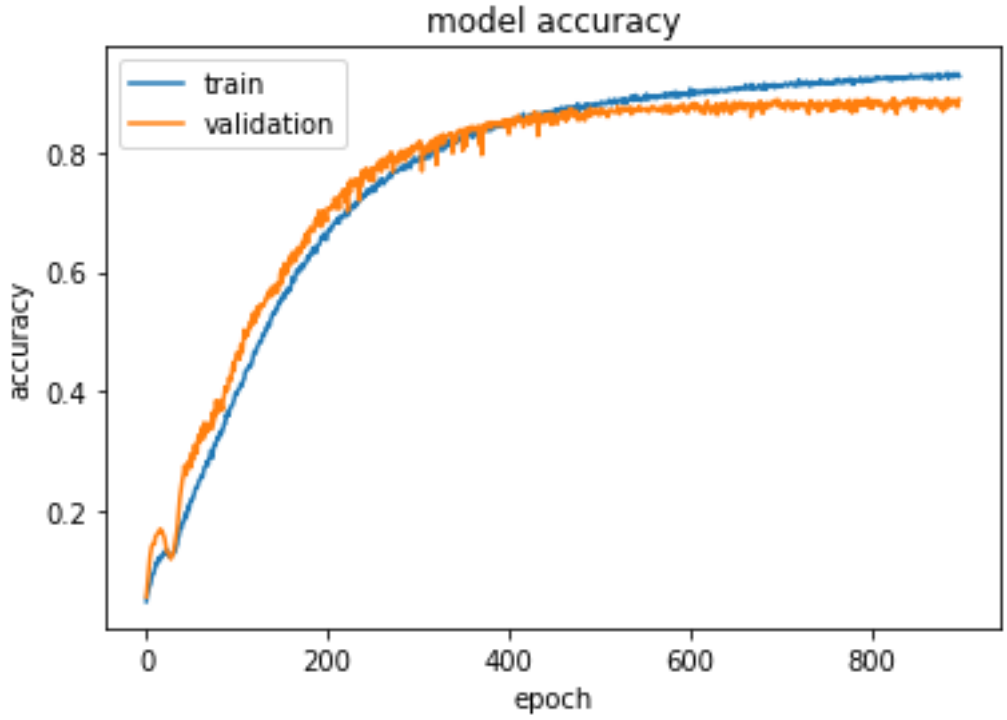
wordCount = 100

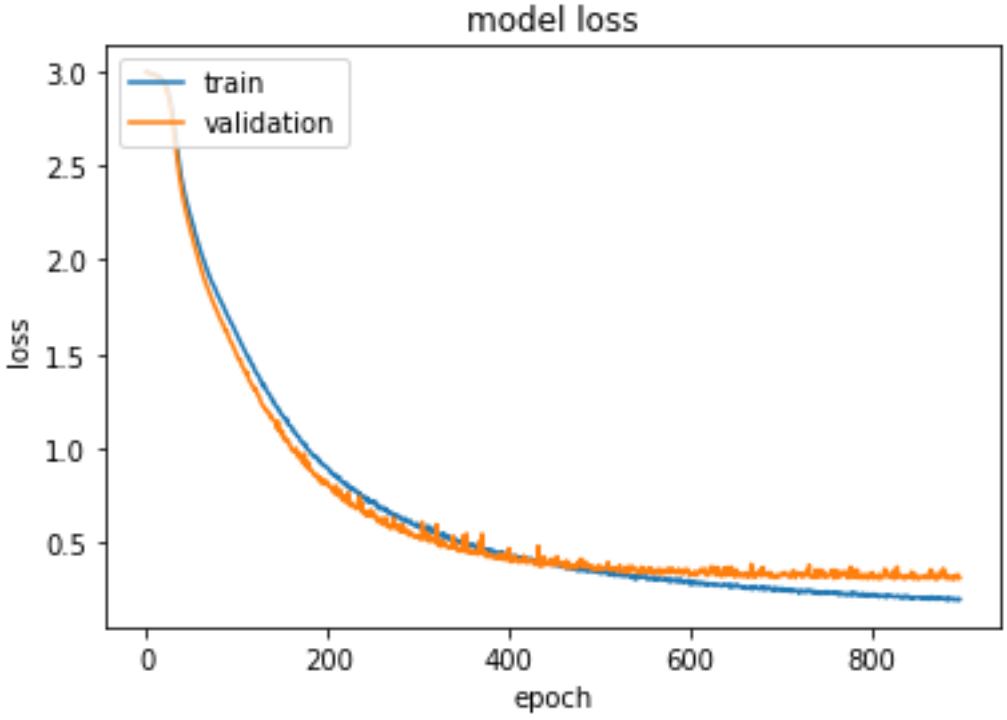




test loss: 0.365488126377 test accuracy 0.876

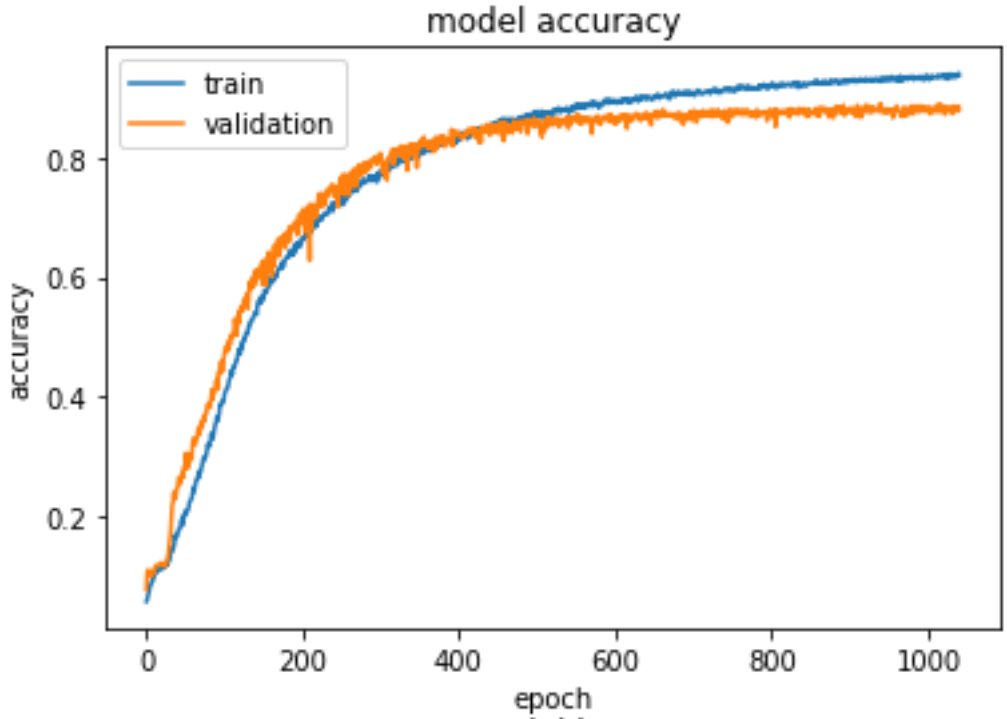
wordCount = 200

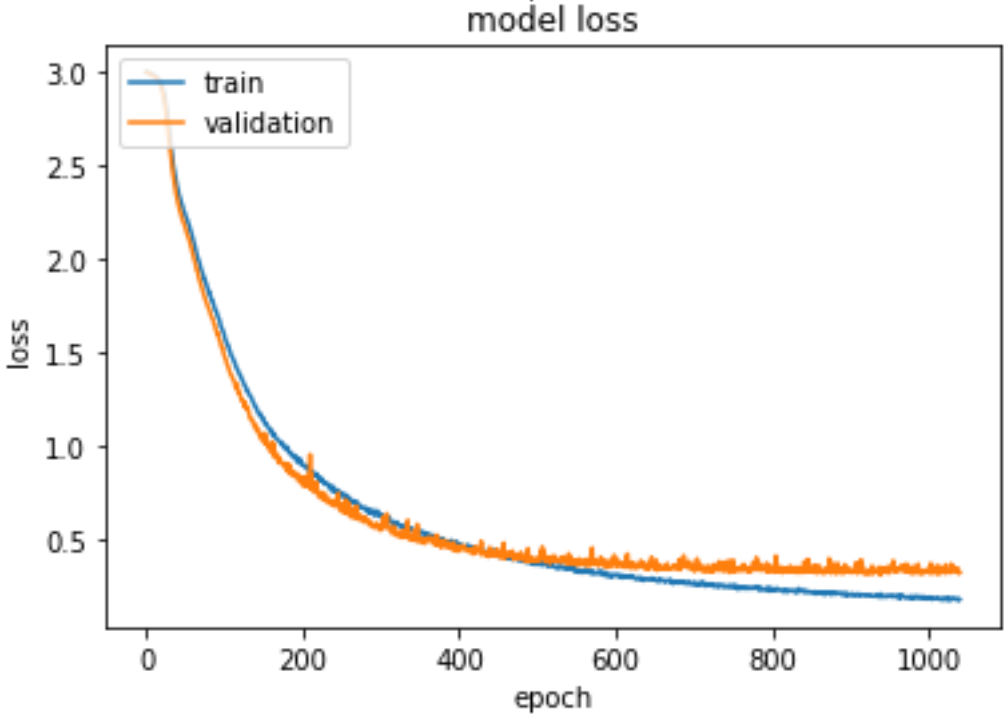




test loss: 0.323876729618 test accuracy 0.888666666667

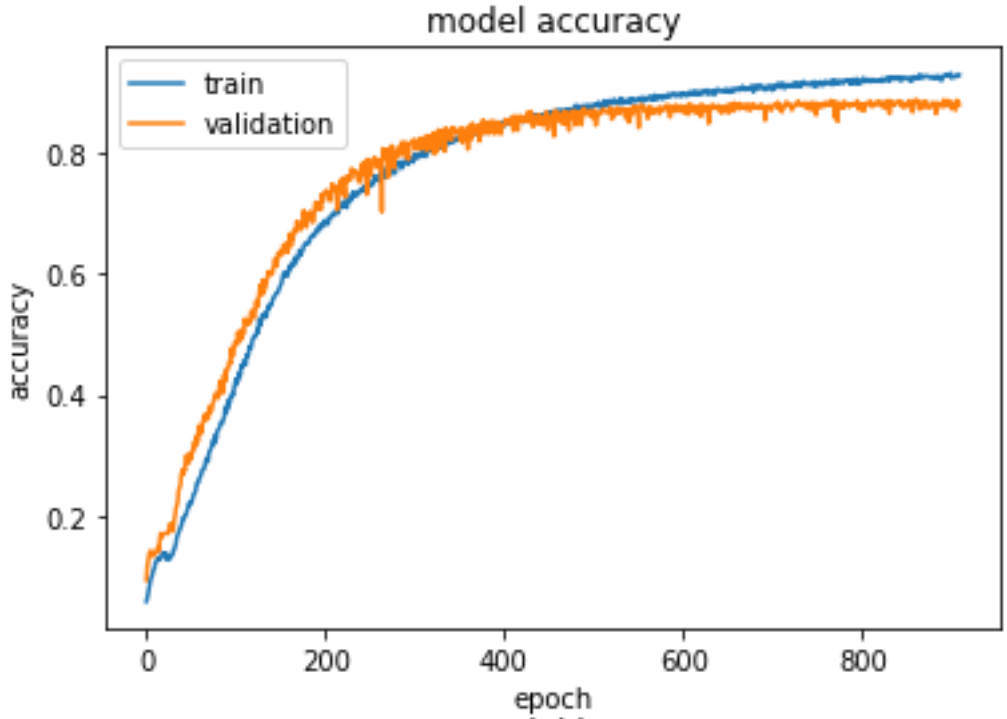
wordCount = 300

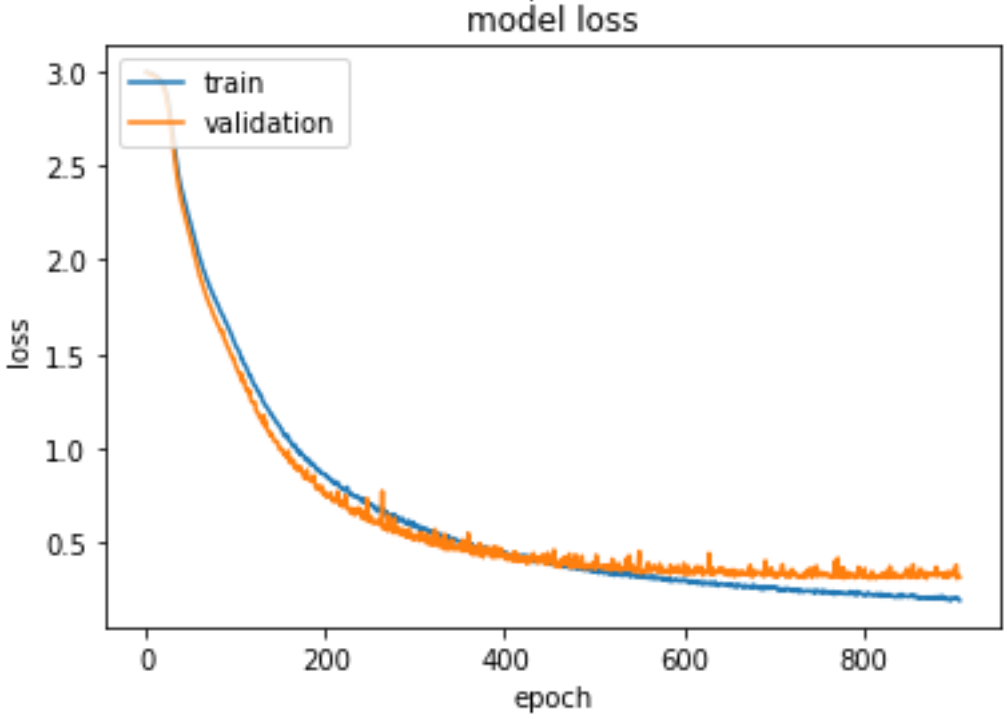




test loss: 0.286920311093 test accuracy 0.900833333333

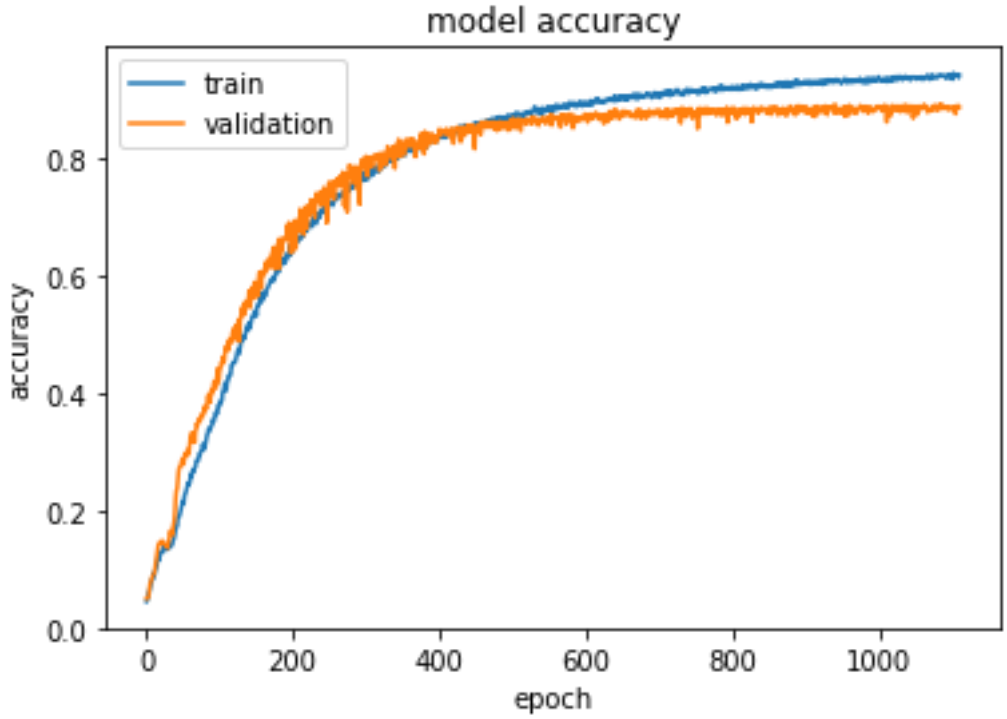
wordCount = 400

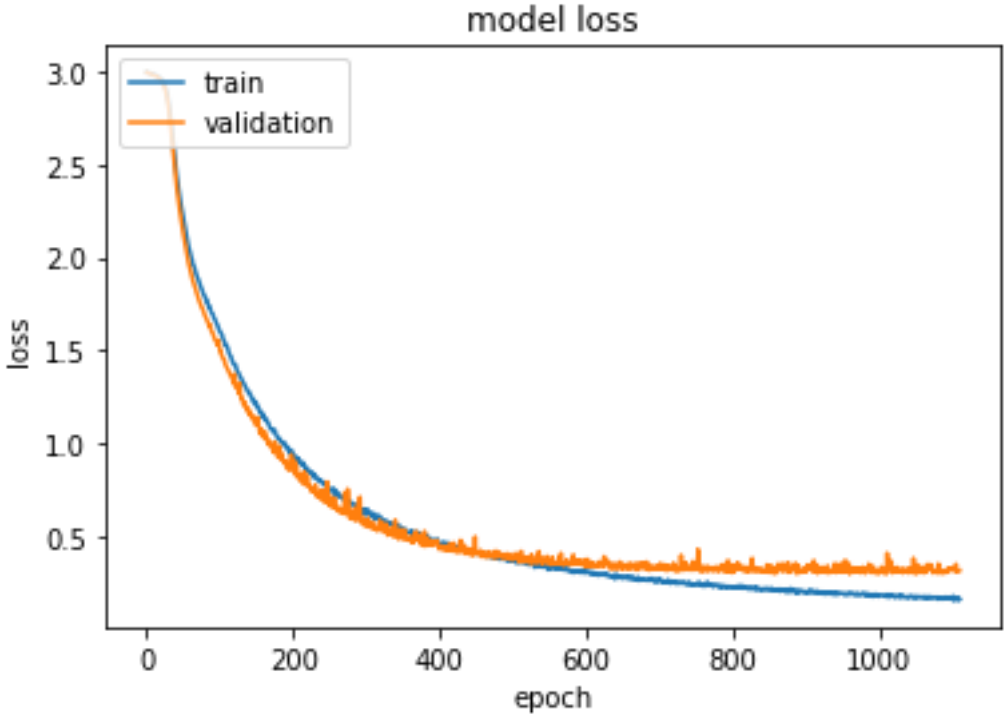




test loss: 0.311032900383 test accuracy 0.890666666667

wordCount = 500





test loss: 0.308422126452 test accuracy 0.896

对于使用不同最高词频单词的个数代表文档向量，得到的结果不同进行了如下统计：

wordCount	测试集准确率
50	83.50%
100	87.60%
200	88.86%
300	90.08%
350	88.80%
400	89.07%
500	89.60%

3. 对项目的思考

项目流程：

分析需求->获取数据->数据清洗->特征工程 ->算法模型选择->算法参数优化->最终结果

1. 分析需求，了解项目的背景与目的
2. 获取相关的数据
3. 数据清洗，提取有用的数据，去掉没用的数据，做一些归一化，标准化操作，还有祛除噪点数据等，这一步我做了停用词的过滤，特殊字符的祛除，转小写
4. 特征工程，很重要的环节，如果做得好，使用的不同的算法，都能达到很好的效果，如果这一步做得不好，算法再好也很难得到一个好的效果，

我的尝试：使用 word2vec 对数据集进行训练（效果不佳），使用 google 训练好的 word2vec 模型，使用 glove 模型，使用 doc2vec 直接训练数据得到文档向量

5. 算法的选择，原则上是多选择一些算法，然后找到一个最理想的算法，我选择的模型：普通神经网络，卷积神经网络，循环神经网络
6. 得到最好的训练模型，经过不断的训练发现普通神经网络得到的效果最佳
7. 针对最好的模型进行参数调优（如果还是达不到预期的结果）需要不断重复返回 3-5 步，对最高词频数不断的迭代和更换，得到最佳参数 300
8. 得到最终结果

整个过程中看到模型准确率的提升是很让人兴奋的，从 60%-70%-80%-90% 优化模型的过程是曲折的，需要做大量的试验，同时也需要查找资料，不断的尝试各种方法，不断的总结经验。

4. 需要作出的改进：

(1) 算法模型有点简单，可以尝试更复杂的模型，现在使用的是四层神经网络

(2) 此模型只限于英文文档的分类，需要加入更适用于中英文的词向量模型