毕业设计报告

目录

1.问题的定义

1.1 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

自然语言处理是一门融语言学，计算机科学，数学于一体的科学，通过计算机和自然语言进行通信，特别是机器学习算法，在大量样本训练基础上可以实现文档自动分类，看图说话，人机对话，语言翻译等

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，提高工作效率

数据集：

数据集来自 20 Newsgroups 官网,是由 Ken Lang 收集，为了他的论文 *Newsweeder: Learning to filter netnews[1]*使用到的数据集，该数据集由 20 个新闻组数据，20000 个新闻文档，且每个组有 1000 个左右的新闻，这个数据集已经成为机器学习文本应用试验的流行的数据集，特别是文本分类，因此使用这个数据集是很合理的，数据集下载链接地址：20news-19997.tar.gz

1.2 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，采用监督学习来解决分类问题

策略：

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，统计文档单词的词频，统计文档的词云，统计文档的单词数量

第三步：根据单词数量过滤，去掉单词数量极少和极多的文档，过滤规则：Q1=上四分之一位 Q3=上四分之三位 [Q1-0.7(Q3-Q1),Q3+0.7(Q3-Q1)],不在这个区间的文档全部过滤掉

第四步：将剩下的文档打乱顺序，将文档按 0.7:03 的比例分成训练集和测试集，将训练集按 0.8:0.2 分成训练集合验证集，将训练集的数据通过 word2vec 训练得到词向量（只使用训练数据，不是将训练测试数据统一拿过来进行训练，因为测试数据对于训练模型来讲是未知的，可能存在很多未知单词，如果模型泛化能力不强就会出现明显过拟合现象），将文档的所有单词的词向量进行叠加取平均得到新闻对应的文档向量（如果从这个单词的词向量不存在，则忽略这个单词）

第五步：将文档向量使用全连接层神经网络进行训练，采用的是 300 个输入层->100 个神经元的隐藏层->200 个神经元的隐藏层->300 个神经元的隐藏层->20 个神经元的输出层，输入层和隐藏层之间使用的是 relu 作为激活函数，隐藏层和输出层之间使用的是 softmax 作为激活函数

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

1.3 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

2.分析

2.1 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：

文档单词数量统计

	docs	wordCount
0	data/alt.atheism/49960	1257
1	data/alt.atheism/51060	2985
2	data/alt.atheism/51119	472
3	data/alt.atheism/51120	220
4	data/alt.atheism/51121	179

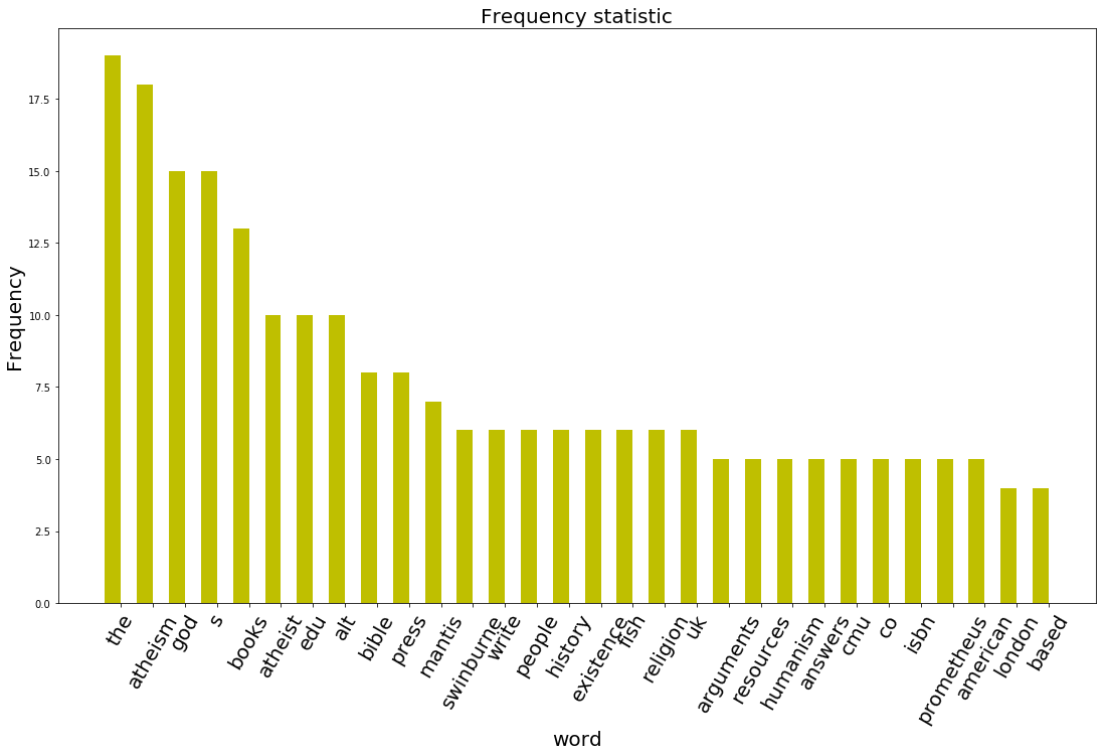
文档单词平均数的统计

	wordCount
count	19997.000000
mean	260.238786
std	589.159527
min	45.000000
25%	140.000000
50%	188.000000
75%	258.000000
max	37786.000000

2.2 探索性可视化

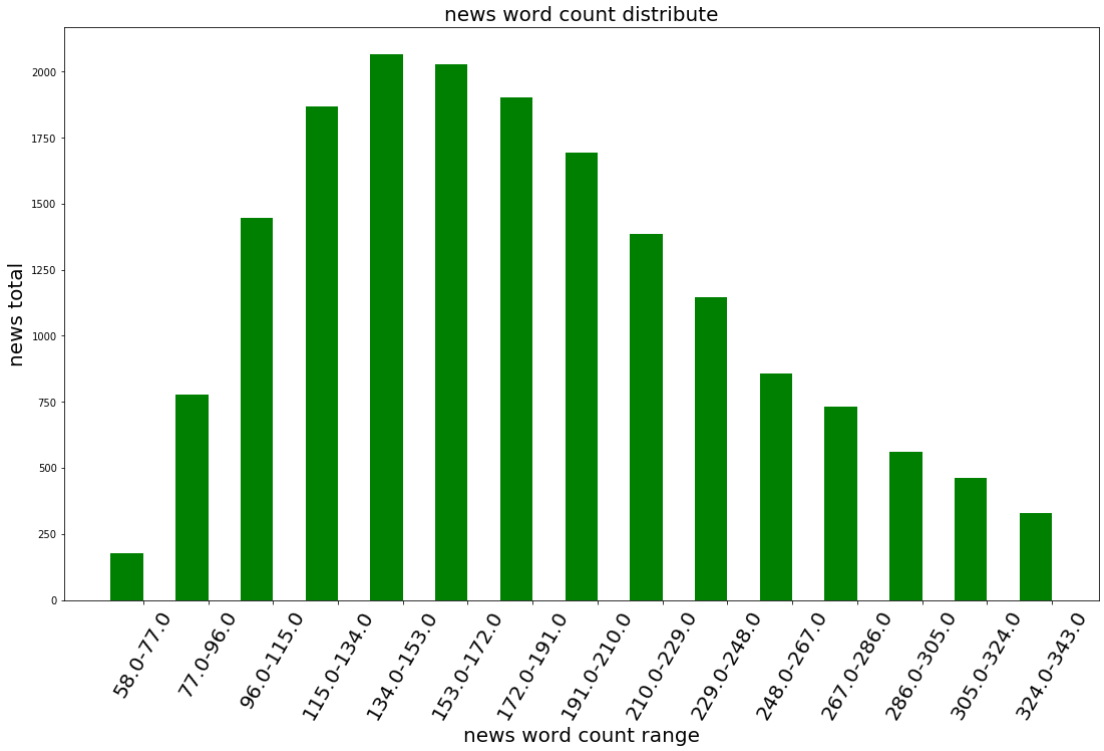
新闻词频统计：对新闻 data/alt.atheism/ 49960 进行词频统计，单词 the，atheism 等出现的次数较多

文档单词词频的统计



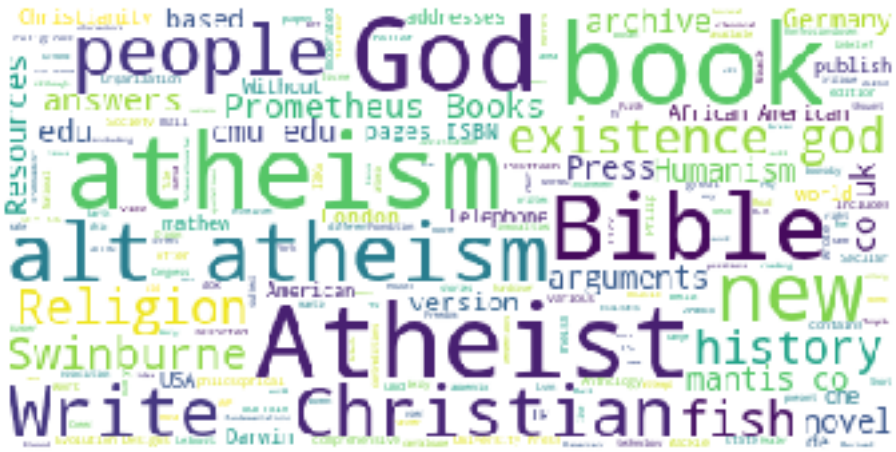
新闻单词数量分布情况：对所有文档的单词数进行统计，单词数量的分布属于偏正太分布，单词数量在 134-153 之前新闻数量最多

文档单词数量分布统计



新闻词云：（单词词频越大，单词字体越大），对新闻 data/alt.atheism/ 49960 生成词云，发现单词 alt，athesim 和新闻分类相关的单词出现的次数较多，能够体现文档的分类特点

文档单词词云统计



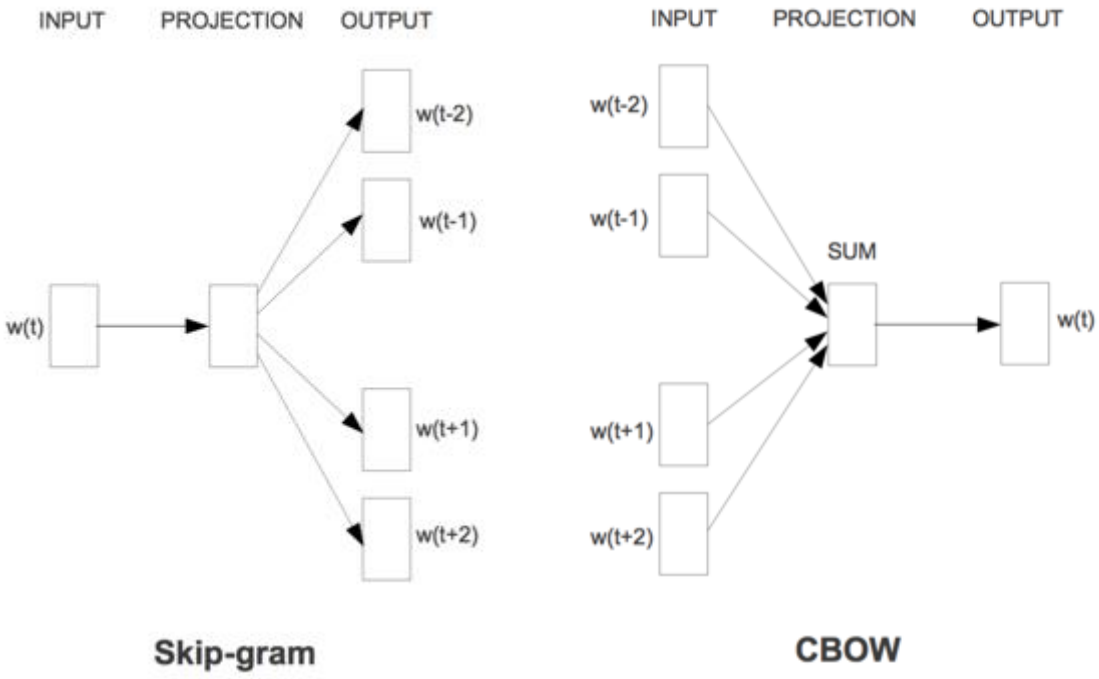
2.3 算法和技术

使用的算法：随机梯度下降，BP 神经网络，循环神经网络，卷积神经网络，word2Vec 算法

随机梯度下降算法：SGD 是用于最小化损失函数的一种常用算法，具体推导过程参考：[3]

word2vec 算法：

word2vec 是 Google 于 2013 年开源推出的一个用于获取 word vector 的工具包，十分简单，高效，使得自然语言的处理变得更加简单
包含两套框架：分别是 hierarchica softmax 和 negative sampling
包含两种模型：CBOW 和 Skip-gram



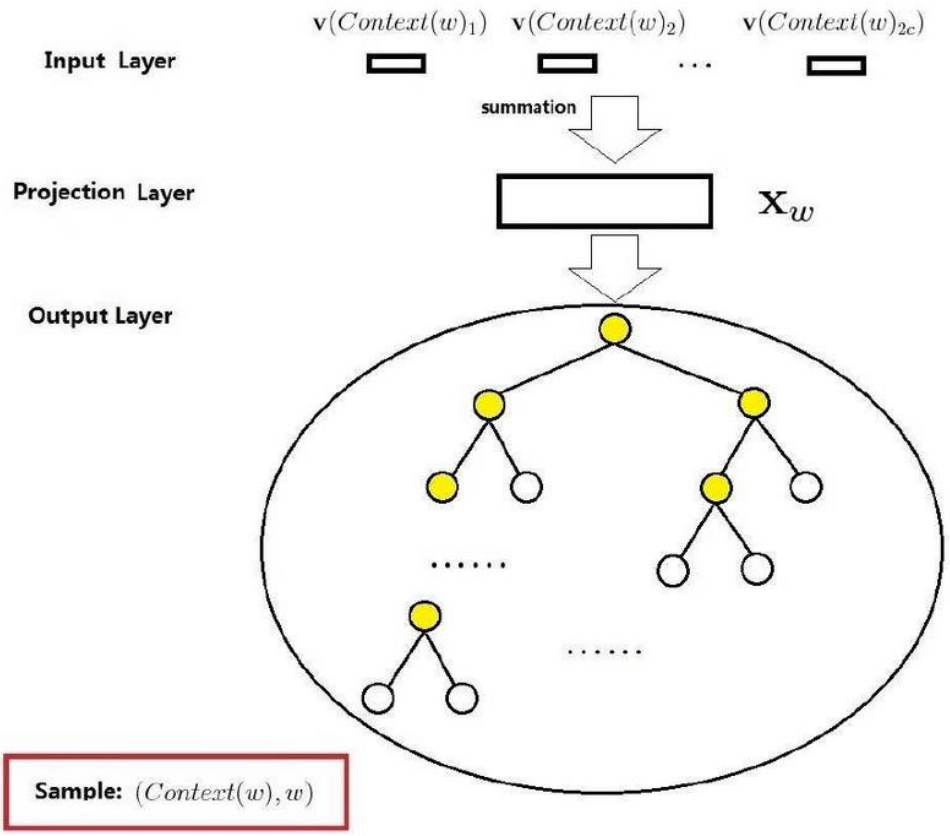
两种模型：

CBOW：Continuous Bag-of-Words Model
模型包含三个层：输入层，投影层，输出层，模型是已知上下文 Wt-2，Wt-1，Wt+1，Wt+2 的前提下预测当前词 Wt

Skip-gram：（Continuous Skip-gram Model）
模型包含三个层：输入层，投影层，输出层，模型和 CBOW 刚好相反，是已知当前词 Wt，预测其上下文 Wt-2，Wt-1，Wt+1，Wt+2

两种框架：

hierarchica softmax：是基于哈夫曼树实现的，语料中的每个词就是该树中的一个叶子节点，并按词在语料中词频作为树的权重

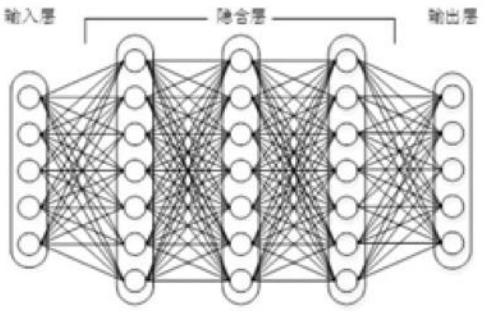


negative sampling：是基于随机负采样实现的
模型中已知 W 上下文，对于给定的上下文，词 W 就是一个正样本，其他词就是负样本

使用 word2vec 算法首先可以得到每一个词的词向量，取文档频率最高的词的词向量叠加求平均得到文档的文档向量，得到了文档向量，接着使用不同的神经网络模型对文档进行分类

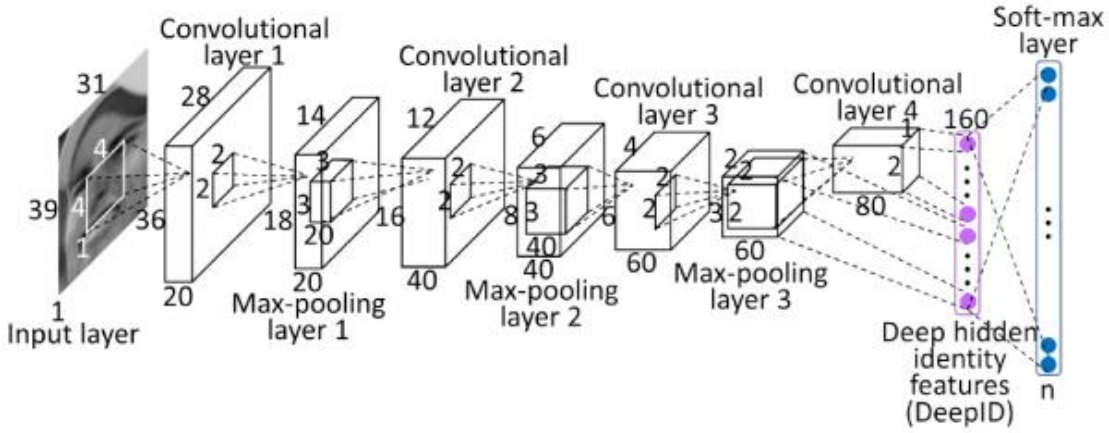
BP 神经网络：

网络结构包括输入层，多个隐层，输出层，输入的特征向　量通过隐含层变换达到输出层，在输出层得到分类结果，随着神经网络层数　的增加，容易陷入局部最优解和梯度消失的问题，具体推导过程参考：[4]



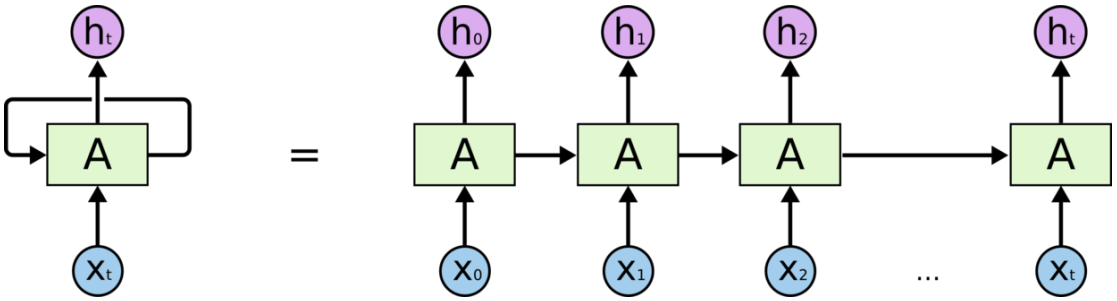
卷积神经网络：

全连接 DNN 的结构里下层神经元和所有上层神经元都能够形成连接，带来的潜在问题是参数数量的膨胀，卷积神经网络，通过“卷积核”作为中介。同一个卷积核权重是共享的，图像通过卷积操作后仍然保留原先的位置关系，这样可以很好的解决参数数量膨胀的问题，网络结构：卷积层，max_pooling 层，dropout 层，全连接层，参考[5]



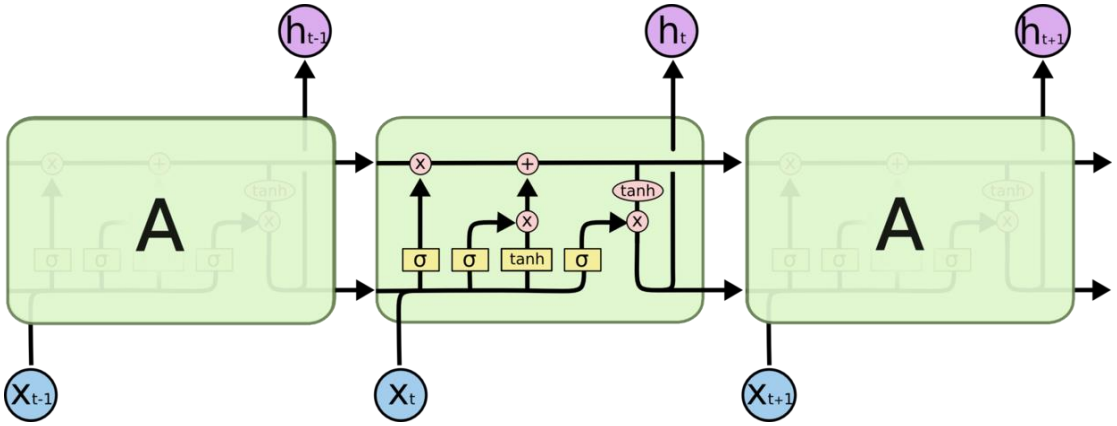
循环神经网络：

全连接神经网络，无法对时间序列的变化进行建模，对于存在时间序列的问题可以使用循环神经网络，循环神经网络可以将上一个状态作为下一个神经元的输入的一部分，网络结构：多层循环网络，参考[6]



LSTM（Long Short Term）：

通过输入门，输出门，遗忘门来控制 Cell 的状态，通过门的开关实现时间上记忆功能



网络结构	适用场景	优点	缺点
全连接神经网络	分类	分类的准确度高 并行分布处理能力强, 分布存储及学习能力强	神经网络需要大量的参数，如网络拓扑结构、 权值和阈值的初始值 随着网络层数增加，易梯度消失
卷积神经网络	图片处理	权重共享可以减少网络的训练参数，使神经网络结构变得更简单，适应性更强 特征提取和模式分类同时进行，并同时在训练中产生	结构复杂，训练时间较长
循环神经网络	序列问题	通过们的开关可实现时间上的记忆功能	并行计算难 易梯度爆炸和消失

合理性：

使用最高词频的词代表一个文档，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 不断下降，准确率不断提高，最后得到一个最好的模型

2.4 基准模型

使用 word2vec 训练词向量，采用的是 Negative sampling 和 CBOW 模型，window=5（表示当前词与预测词在一个句子中的最大距离是多少），min_count=2（可以对字典做截断. 词频少于 min_count 次数的单词会被丢弃掉, 默认值为 5），size=300（是指特征向量的维度，默认为 100），得到词向量后用 4 层的全连接神经网络作为算法的基准模型
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：基准模型的测试集准确率 85%
性能评估标准：
准确率 > 85%（基准模型测试集准确率） 则性能较好

3.方法

3.1 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
第五步：对于单词数量极少或者极多的文档作为异常文档，作丢弃处理

3.2 执行过程

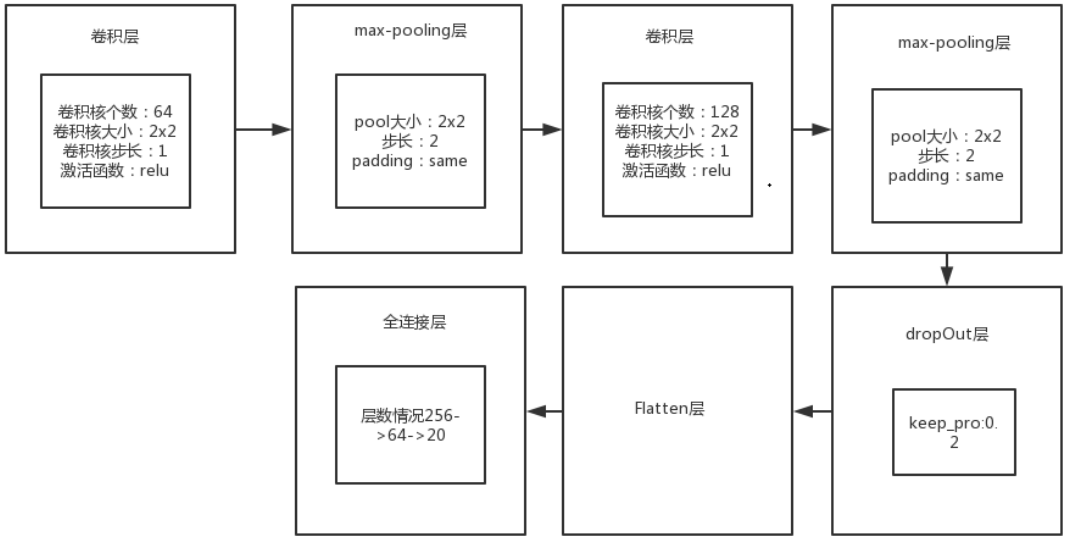
全连接神经网络结构：

测试尝试各种组合测试集准确率变化表格

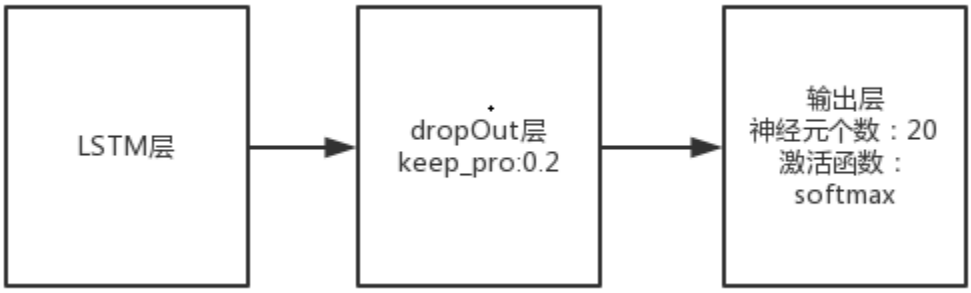
框架	模型	窗口大小	min_count	测试集准确率
Negative sampling	CBOW	5	1	81.20%
Negative sampling	CBOW	5	2	85.00%
Negative sampling	CBOW	5	3	83.00%
Negative sampling	CBOW	5	5	79.00%
Negative sampling	CBOW	6	2	84.20%
Negative sampling	CBOW	7	2	85.30%
Negative sampling	CBOW	8	2	85.70%
Negative sampling	CBOW	9	2	85.20%
Negative sampling	skip-gram	5	2	84.70%
Hierarchica softmax	CBOW	5	2	88.70%
hierarchica softmax	skip-gram	5	2	90.20%

词向量训练的方式不同，测试集准确率也不同，表格 1-4 行控制其他变量不变，改变 min_count,当 min_count=2 时准确率最高，表格 5-8 行，改变窗口大小，发现提升不大，表格第九行，使用 skip-gram 模型，测试准确率没有多大变化，第 10 行改变框架，使用基于哈夫曼树的框架，CBOW 模型，测试集准确率提升较大。第 11 行，使用基于哈夫曼树的框架，skip-gram 模型，准确率达到了 90.2%。

卷积神经网络结构：



使用上述结构的卷积神经网络，得到的测试准确率 86.75%
循环神经网络结构：

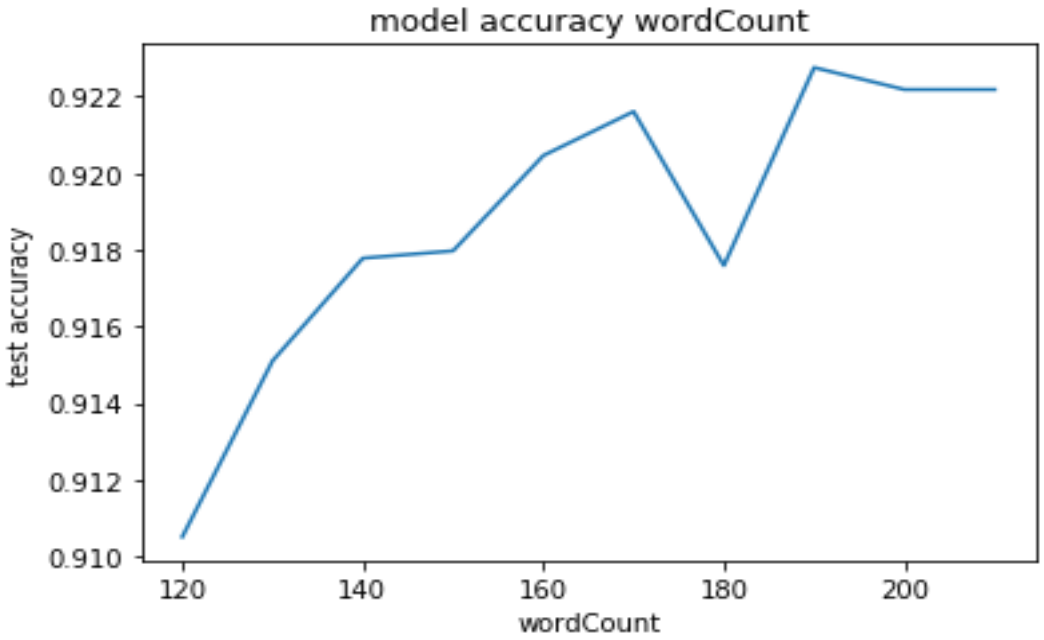


使用上述结构的循环神经网络，得到的测试准确率：85.48%

3.3 完善

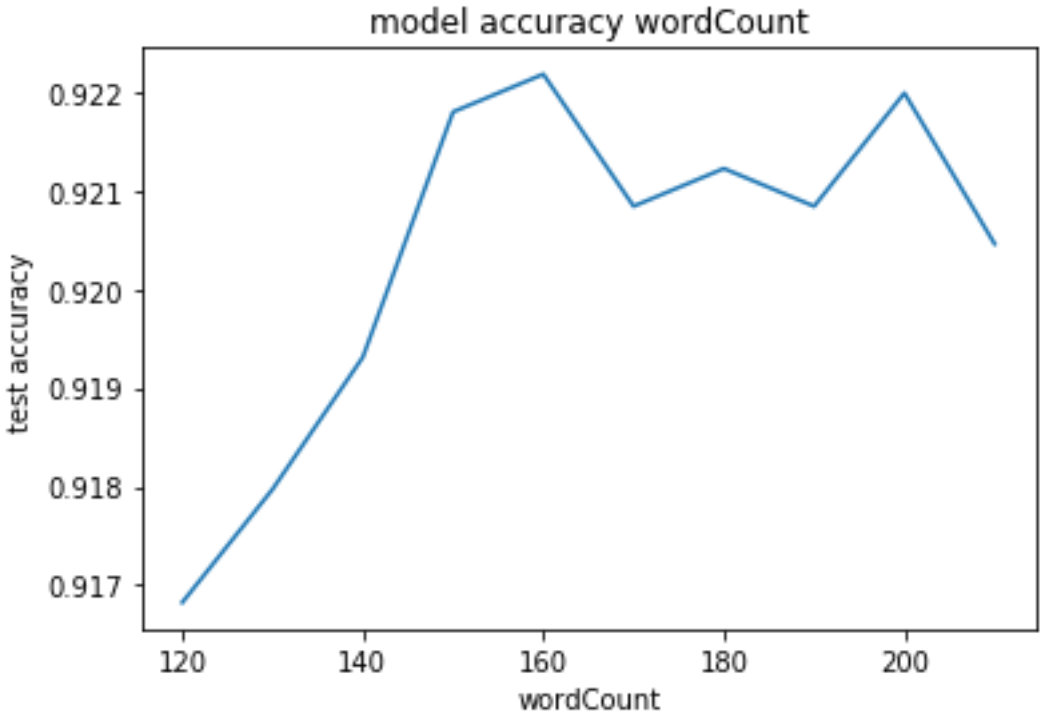
针对执行过程中得到准确率 90.2%的模型（word2vec 采用 hierarchica softmax，skip-gram，min_count=2，window=5）进行微调
第一步：使用 counter 对词频进行排序，取词频最高的 count 个单词作为文档的向量，其余的单词进行忽略，count 分别取[120,130,140,150,160,170,180,190,200,210],取词频最高的 190 个单词作为文档向量，得到的效果最佳，测试准确率最高

单词词频数量模型测试集准确率的变化



第二步：上一轮只是过滤掉了词频较低的词，并没有对单个词的词向量进行加权，对每个单词词向量进行累加之前乘以一个词频，得到的测试准确率89.87%，乘以词频影响比较大（有的词频较大），尝试使用乘以词频的对数，得到的准确率 78%，因为词频为 1 的对数得到的是 0，相当于忽略了词频为 1 的单词，做了一个判断如果对数<1，词向量不变，如果大于 1 词向量*词频的对数，得到的准确率：

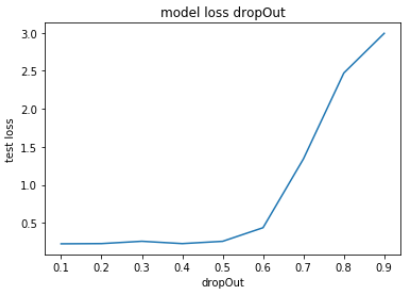
单词词频数量模型训练集准确率的变化

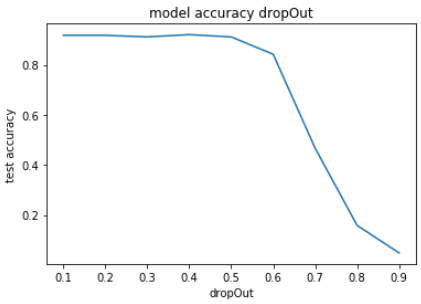


4.结果

4.1 模型的评价与验证

对参数 dropout 进行调优，分别取 0.1-0.9，统计结果如下图所示：随着参数的增大，模型越容易过拟合





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

4.2 合理性分析

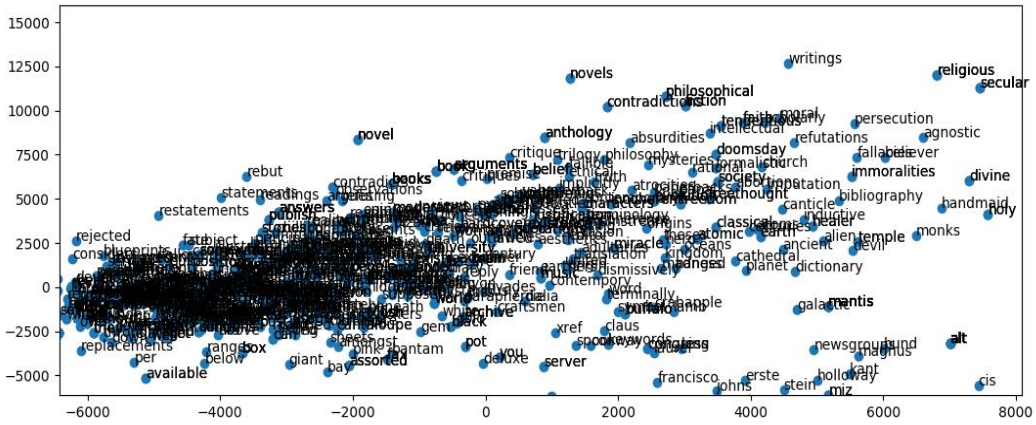
最终模型比基准模型表现的更好，准确率达到比基准模型准确率 85%高一些，损失率比基准模型更高一些，，通过这个模型，可以通过机器学习的方式解决新闻分类问题，真正解决了实际的问题

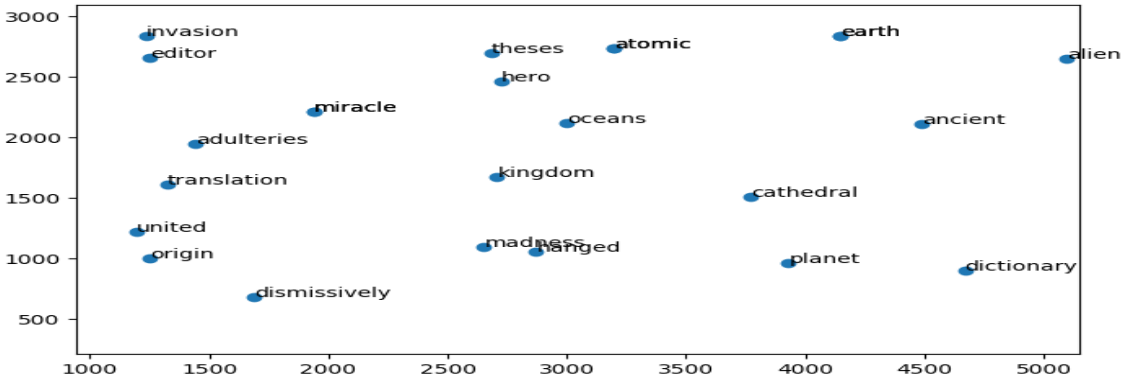
5.项目结论

5.1 结果可视化

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

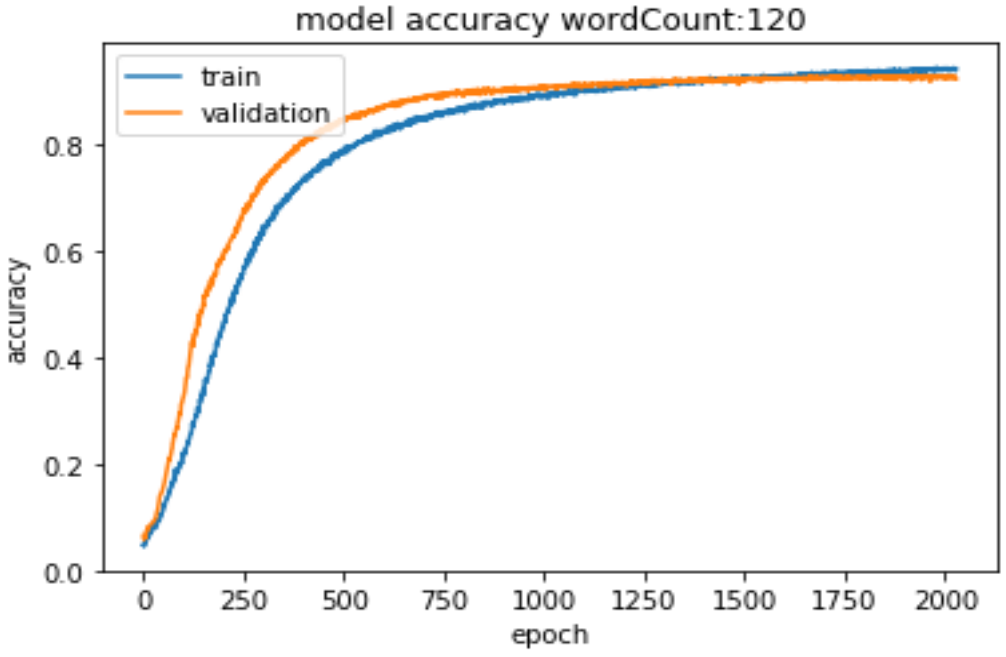
词向量图



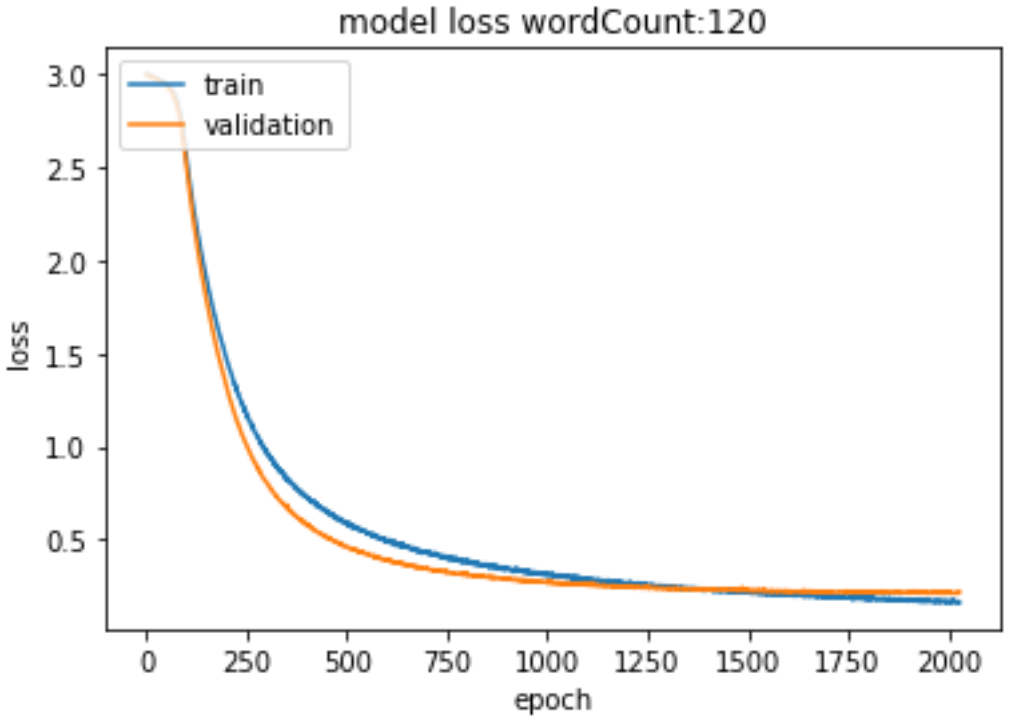


使用 120 个最高词频的单词向量叠加求平均作为文档向量，下图是准确率和损失函数的变化情况：测试准确率 91.05%

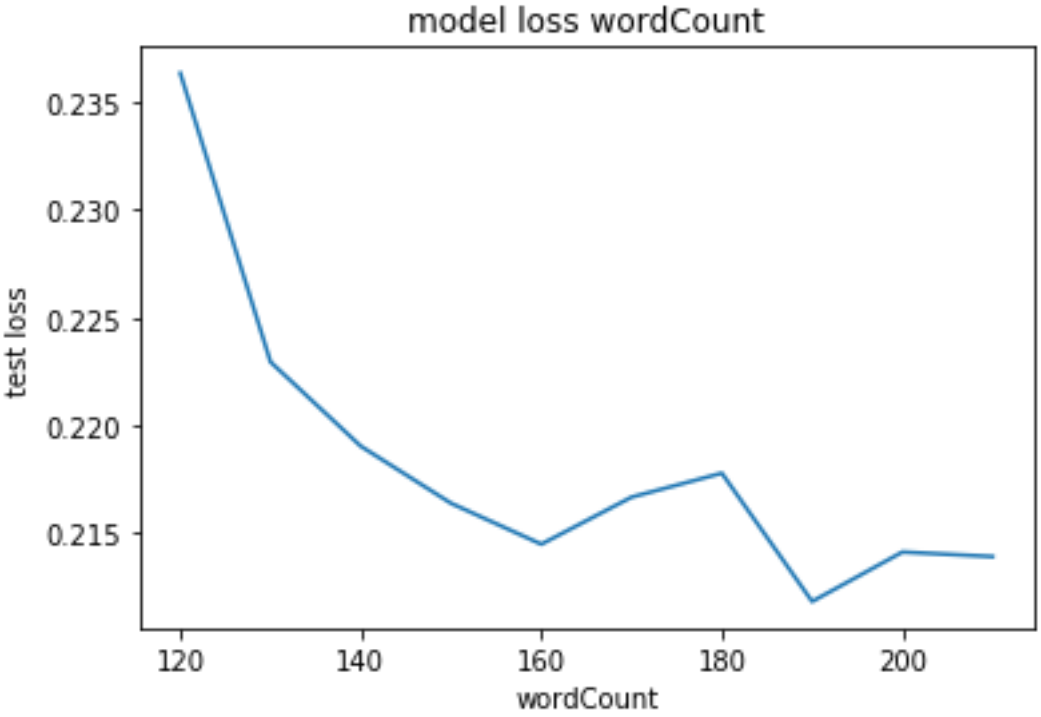
全连接神经网络训练集准确率随着训练轮数增加的变化



全连接神经网络训练集损失率随着训练轮数增加的变化

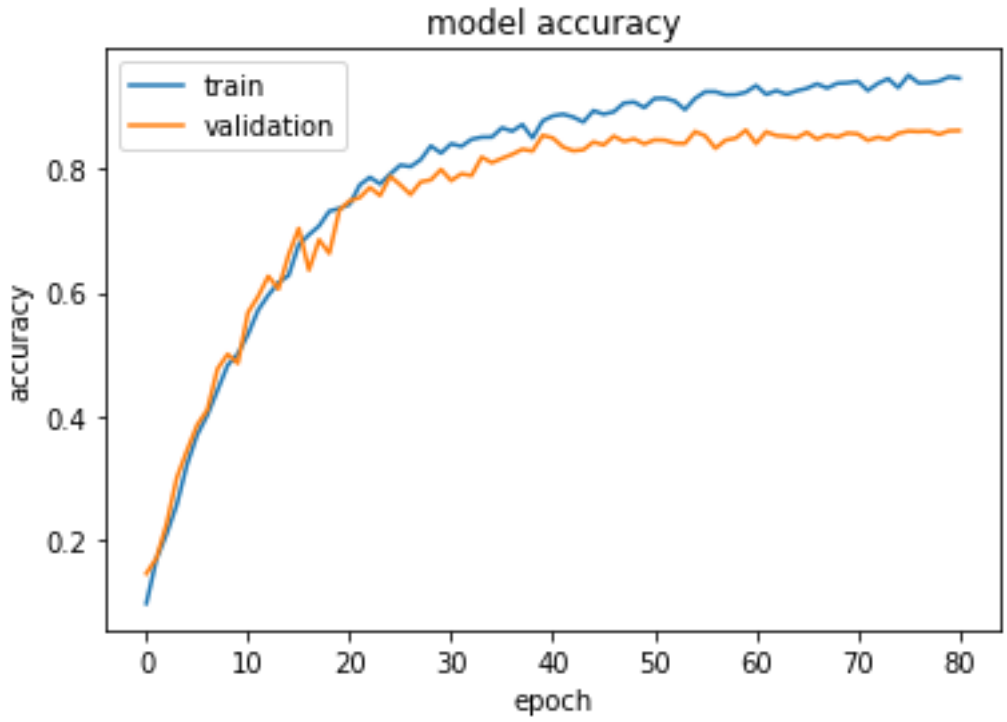


下图是分别取不同最高词频的单词数量，得到测试文档准确率和损失函数的变化：

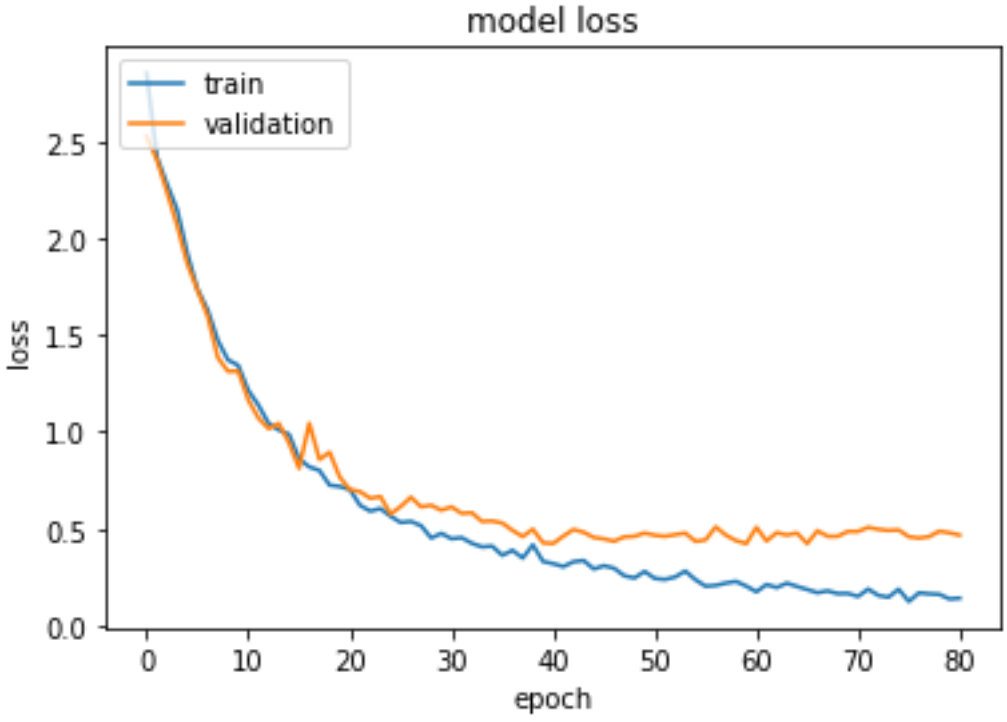


下图是使用 lstm 训练模型，准确率和损失函数的变化，测试准确率：85.49%

LSTM 模型随着训练轮数增加的变化

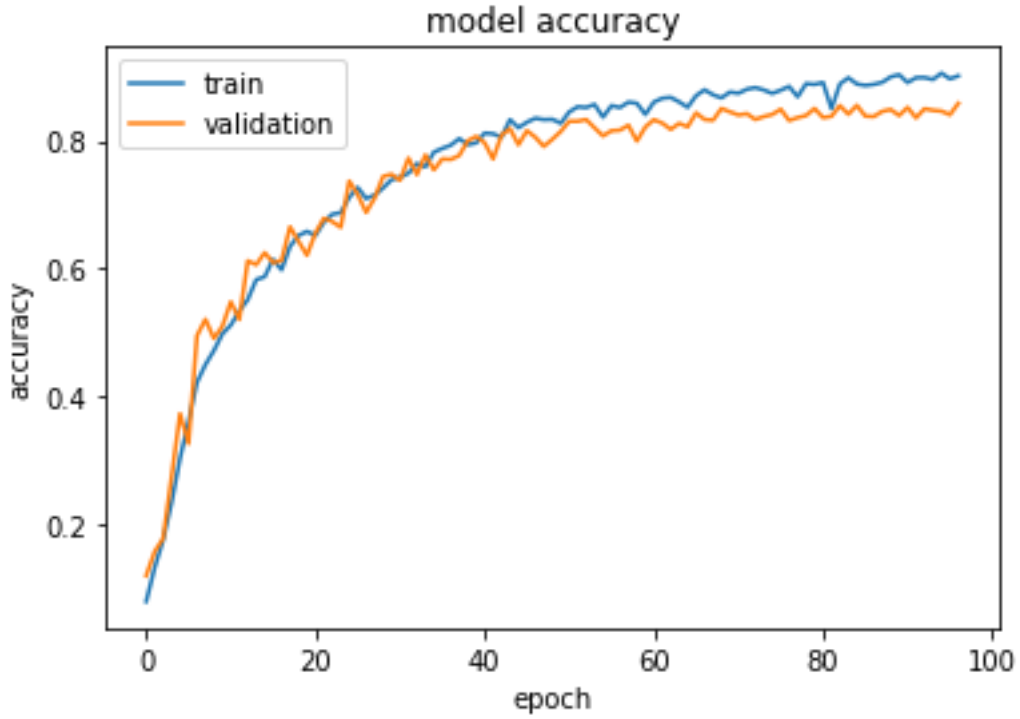


LSTM 模型随着训练轮数损失率的变化

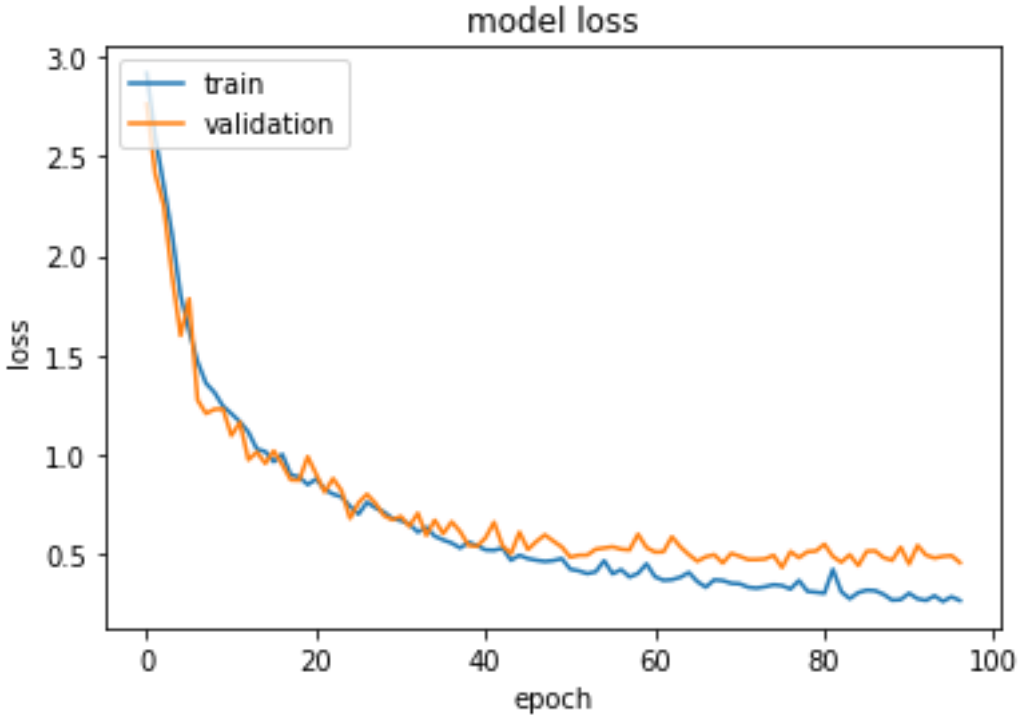


下图是使用 CNN 训练模型，准确率和损失函数的变化，测试准确率：86.75%

CNN 模型随着训练轮数增加准确率的变化



CNN 模型随着训练轮数增加损失率的变化



5.2 对项目的思考

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，祛除了极少和极多单词数量的文档
4. 特征工程，很重要的环节，我的尝试：使用 word2vec，分别对 Negative sampling ，hierarchica·softmax 和 CBOW，skip-gram 两两组合
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. 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，
7. 得到最终结果

最终模型：使用 word2vec，hierarchica·softmax 和 skip-gram 进行组合，并取词频最高的 190 个词的词向量进行叠加求平均作为文档的向量，使用全连接神经网络作为算法模型：模型结构如下：输入层 300 个神经元，3 层隐藏层，神经元个数分别是：100,200,300 ，激活函数都是使用的 relu，每层后加一个 Dropout 层，防止模型过拟合，输出层是 20 个神经元，使用的激活函数是 softmax，优化器使用的是 SGD，学习率=0.03，训练过程使用回调函数进行提前终止，当验证集的 loss，在 100 轮训中没有下降提前终止，或者验证集在 100 轮训练中准确率没有上升，提前终止

整个过程中看到模型准确率的提升是很让人兴奋的，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。

5.3 需要作出的改进

算法模型简单，现在使用的是四层神经网络，可以尝试更复杂的模型，增加隐藏层的层数和调整神经元的个数
此模型只限于英文文档的分类，语料太小，可以增加更多的语料，如搜狗，维基百科的中文语料
如果使用最终模型作为新的基准：最终模型使用的是 300 维的词向量，可以通过尝试使用 100,200,400,500,600 维的词向量，寻找一个最合适的维度，去找到一个更好的模型

6.参考文献

[1] http://www.qwone.com/~jason/20Newsgroups/lang95.bib（数据集链接）

[2] http://www.qwone.com/~jason/20Newsgroups/（20newgroups 官网）

[3] https://en.wikipedia.org/wiki/Stochastic_gradient_descent（随机梯度推导）

[4] http://blog.csdn.net/zhongkejingwang/article/details/44514073（BP 神经网络）

[5] http://cs231n.github.io/neural-networks-2/#init（卷积神经网络）

[6] http://www.jianshu.com/p/9dc9f41f0b29（循环神经网络）

[7] http://blog.csdn.net/itplus/article/details/37969979（word2Vec）

[8] https://www.zhihu.com/question/34681168（神经网络对比）