机器学习工程师纳米学位

毕业项目：文档分类

2017.05.11

1. 项目背景

1.1 项目概述

自然语言处理是计算机科学领域与人工智能领域中的一个重要方向。在自然语言处理处理面临很多挑战，其中一项任重的任务就是文本分类。文本分类一般包括了文本的表达、 分类器的选择与训练、 分类结果的评价等过程。本项目的研究任务是探究如何使用合理的自然语言处理技术进行文档分类。

文本分类诸多有着广泛的应用。信息检索里的许多任务都可以归结为文本分类问题，包括搜索引擎对网页的相关性排序、邮件的过滤、文档的组织。Google公司和微软公司在网页检索方面已不再满足简单的单词匹配来给出与用户查询相关的网页，越来越多地将信息检索和文本分类的技术引入以更好地理解用户的搜索需求，提供给用户更优的信息处理服务。

本项目将利用20新闻组数据[1]，最早是由Ken Lang收集，在他的论文*Newsweeder: Learning to filter netnews*[2]中使用。 其中包含来自20个不同的新闻组的20000篇文档，将探索不同的文本表示方式，其中包括词袋子模型[3]，利用词向量模型（如word2vec[4]）构建对文本的表示。并尝试使用不同的机器学习模型，如逻辑回归、SVM和神经网络等，利用上述不同的文本表示对文档进行分类。

1.2 问题描述

20新闻组文档分类问题是一个有监督的分类问题。本项目的目标是对文档提取出合适的特征并基于这些特征构造一个有效的模型对文档进行分类。

我将采用的不同的方法对文档进行表示，包括词袋模型、word2vev、glove等词向量方法，并探讨如何通过词向量构造整篇文档的表示方法，并以此为基础采用不同的机器学习算法进行训练，对文档进行分类。

另外，我将数据集分成训练集和测试集，并在测试集上进行训练，测试集用于评估算法的效果。

1.3 评价指标

作为一个多标签的分类问题，我将使用分类准确率作为评估指标：

$$acc = \frac{\sum_{i=1}^{n} I(y_i = \hat{y}_i)}{n}$$

2. 分析

2.1 数据的探索

该项目使用使用经典的20类新闻包，里面大约有20000条新闻，比较均衡地分成了20类，是比较常用的文本数据之一。该数据可利用*sklearn*工具包[5]下载，并且sklearn已根据日期将数据集分为训练集和测试集。其中训练集共11314篇文档、测试集共7532篇文档。共有2308880个单词，有72904个不同的单词（词典的大小）。随机选择其中一篇文档作为实例，其的内容大致如下：

```
"From: guykuo@carson.u.washington.edu (Guy Kuo)\nSubject: SI Clock Poll - Final
Call\nSummary: Final call for SI clock reports\nKeywords:
SI,acceleration,clock,upgrade\nArticle-I.D.:
shelley.1qvfo9INNc3s\nOrganization: University of Washington\nLines: 11\nNNTP-
Posting-Host: carson.u.washington.edu\n\nA fair number of brave souls who
upgraded their SI clock oscillator have\nshared their experiences for this
poll. Please send a brief message detailing\nyour experiences with the
procedure. Top speed attained, CPU rated speed,\nadd on cards and adapters,
heat sinks, hour of usage per day, floppy disk\nfunctionality with 800 and 1.4
m floppies are especially requested.\n\nI will be summarizing in the next two
days, so please add to the network\nknowledge base if you have done the clock
upgrade and haven't answered this\npoll. Thanks.\n\nGuy Kuo
<guykuo@u.washington.edu>\n"
```

通过比对多篇新闻文档可以发现新闻文档的原始数据具有以下特征：

- 几乎所有的组别可以通过标题和发行商容易区别开
- 另一个重要特征涉及发送者是否如其标题或其签名所指示的隶属于大学。
- 基于先前的引用，“article”这个词是一个重要的特征

- 其他特征也与当时发布的特定人物的姓名和电子邮件地址相匹配。

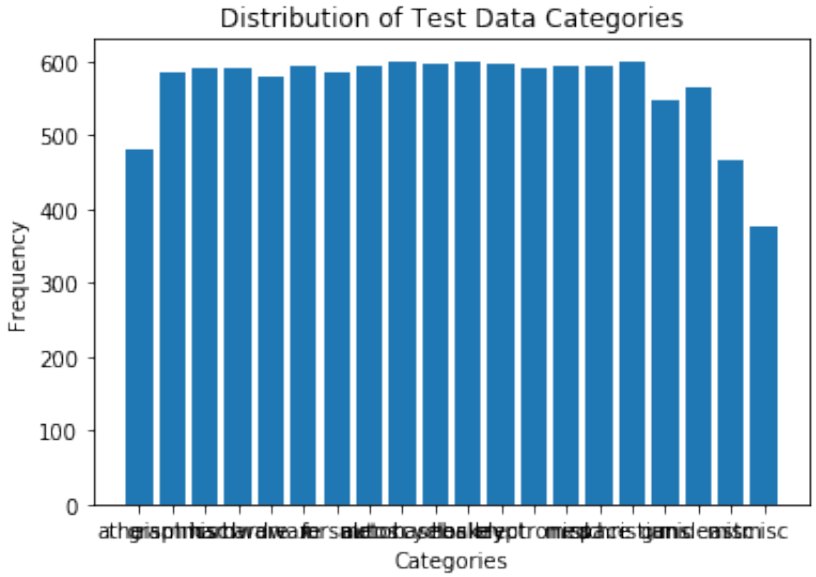
有了如此丰富的线索来区分新闻组，分类器可以不需要从文本中识别主题。为此我将使用*sklearn*工具包[5]对新闻文档进行第一阶段的预处理，去除每篇文档的标题、注脚和引用。去除后的文档如下：

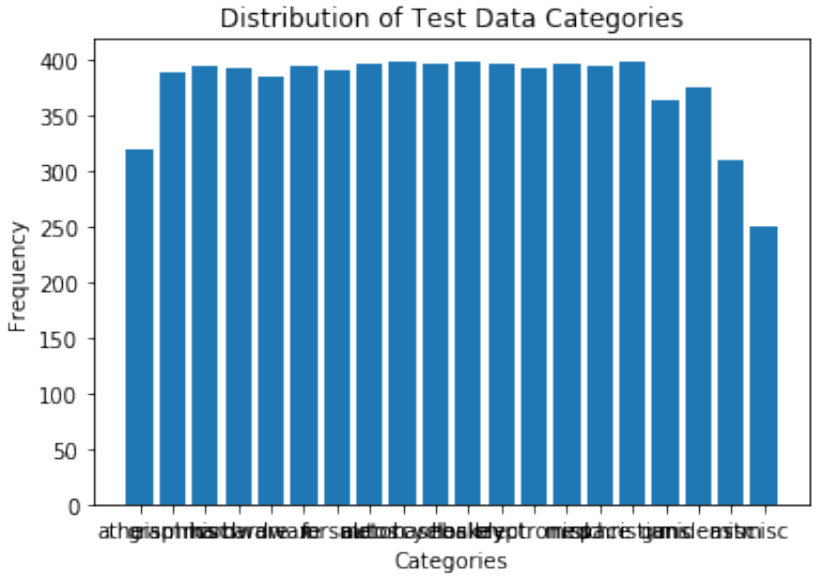
```
"A fair number of brave souls who upgraded their SI clock oscillator
have\nshared their experiences for this poll. Please send a brief message
detailing\nyour experiences with the procedure. Top speed attained, CPU
rated speed,\nadd on cards and adapters, heat sinks, hour of usage per
day, floppy disk\nfunctionality with 800 and 1.4 m floppies are especially
requested.\n\nI will be summarizing in the next two days, so please add to
the network\nknowledge base if you have done the clock upgrade and haven't
answered this\npoll. Thanks."
```

可以看到，去除后的文档简明清晰。同时我还将记录与每篇文档对应的类别。例如上面所示文档的类别为该文档所属类别为'comp.graphics'.由于是新闻文档数据，书写规范、格式合理，我将不再进行纠正拼写等预处理步骤。

2.2 探索性可视化

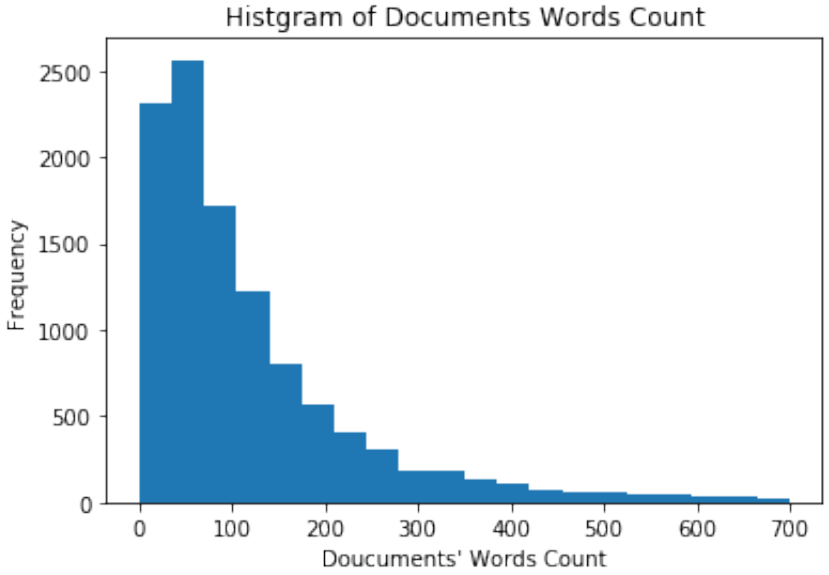
接下来将对数据进行一些可视化的探索。首先分别查看训练集和测试集中不同类别新闻组的文档数的分布。





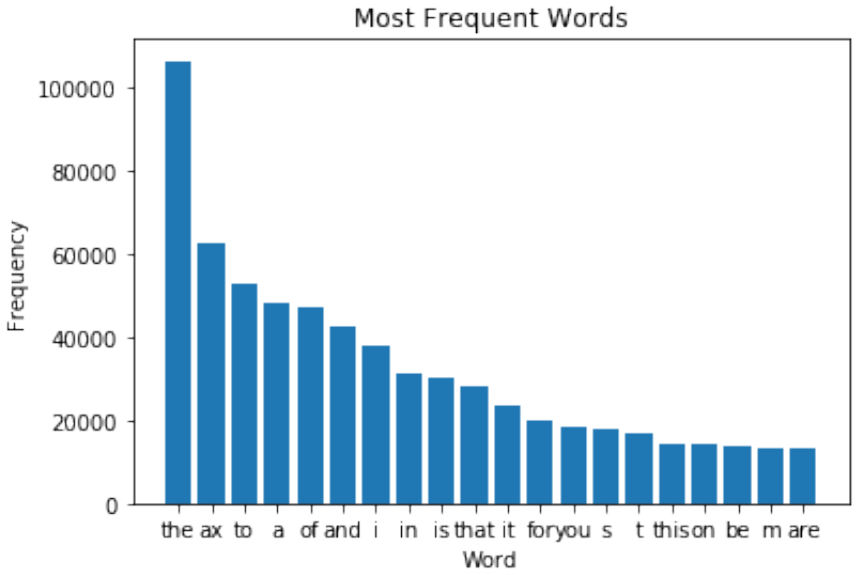
从上图可以看出每个新闻组所包含的文档数据大致相当，在进行分类时无需再考虑样本类别有偏的情况。

接下来利用直方图对每篇文档的字数做一个统计。



可以看到大部分文档的字数在0~300之间。文档字数对序列模型（LSTM）的使用存在影响。所以在LSTM进行文档分类时，为了便于统一处理，我将通过补零或阶段等操作将每篇文档的字数限制在200个字，从上图可以看出，200个字的选择是合理的。

进行最后我们统计一下每个单词出现的次数，这里只展示几个最常见的词。



可以看到出现频率高的单词都是对分类没有实际意义的停词。因此在构建文本的表示需要减少这些单词的影响。如在构建词袋子模型模型时，可以使用tf-idf值（后面会进一步解释）来表示每个单词的权重。

2.3 算法和技术

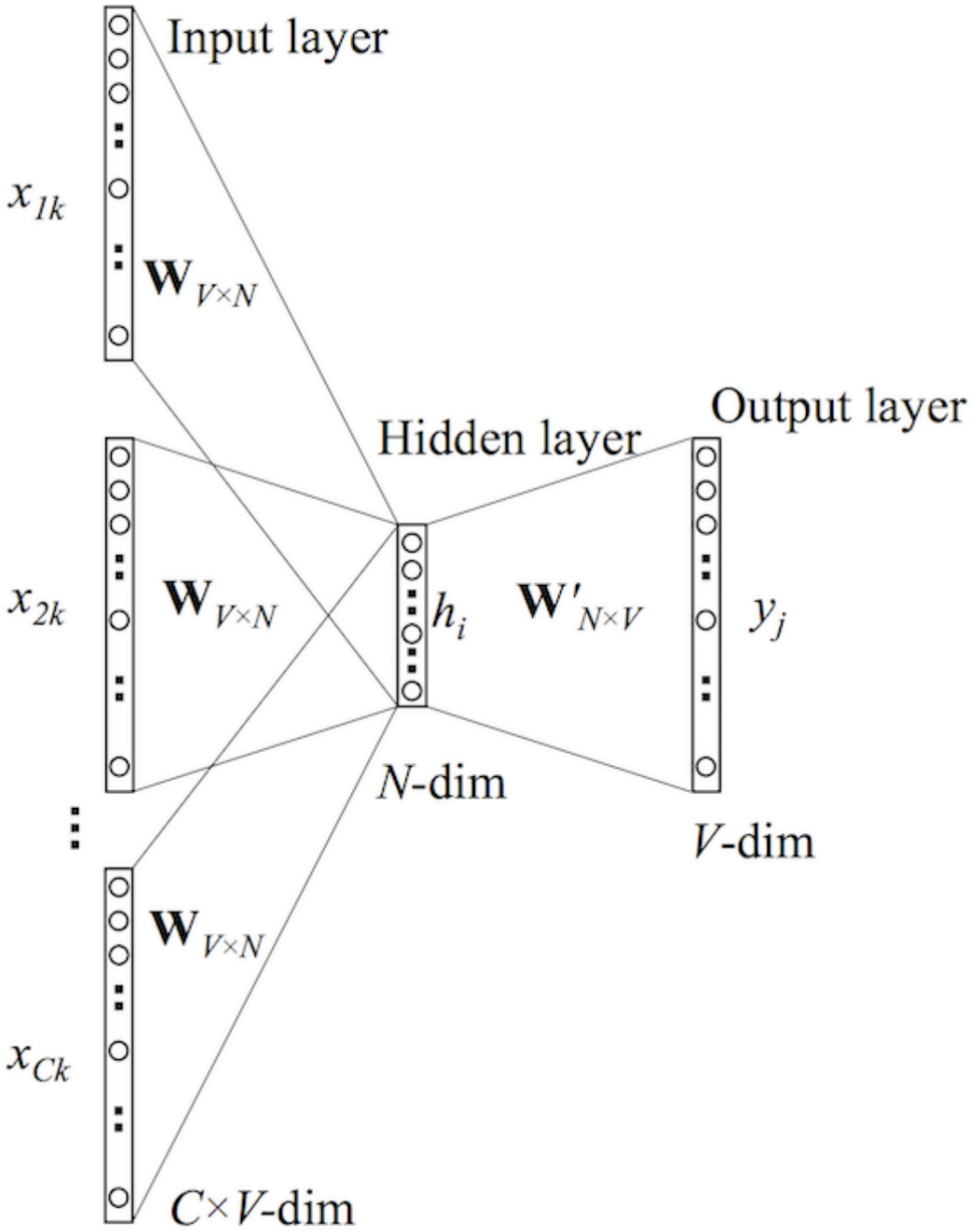
本项目将采用词袋子模型和词向量模型两种方式对文档进行表示。

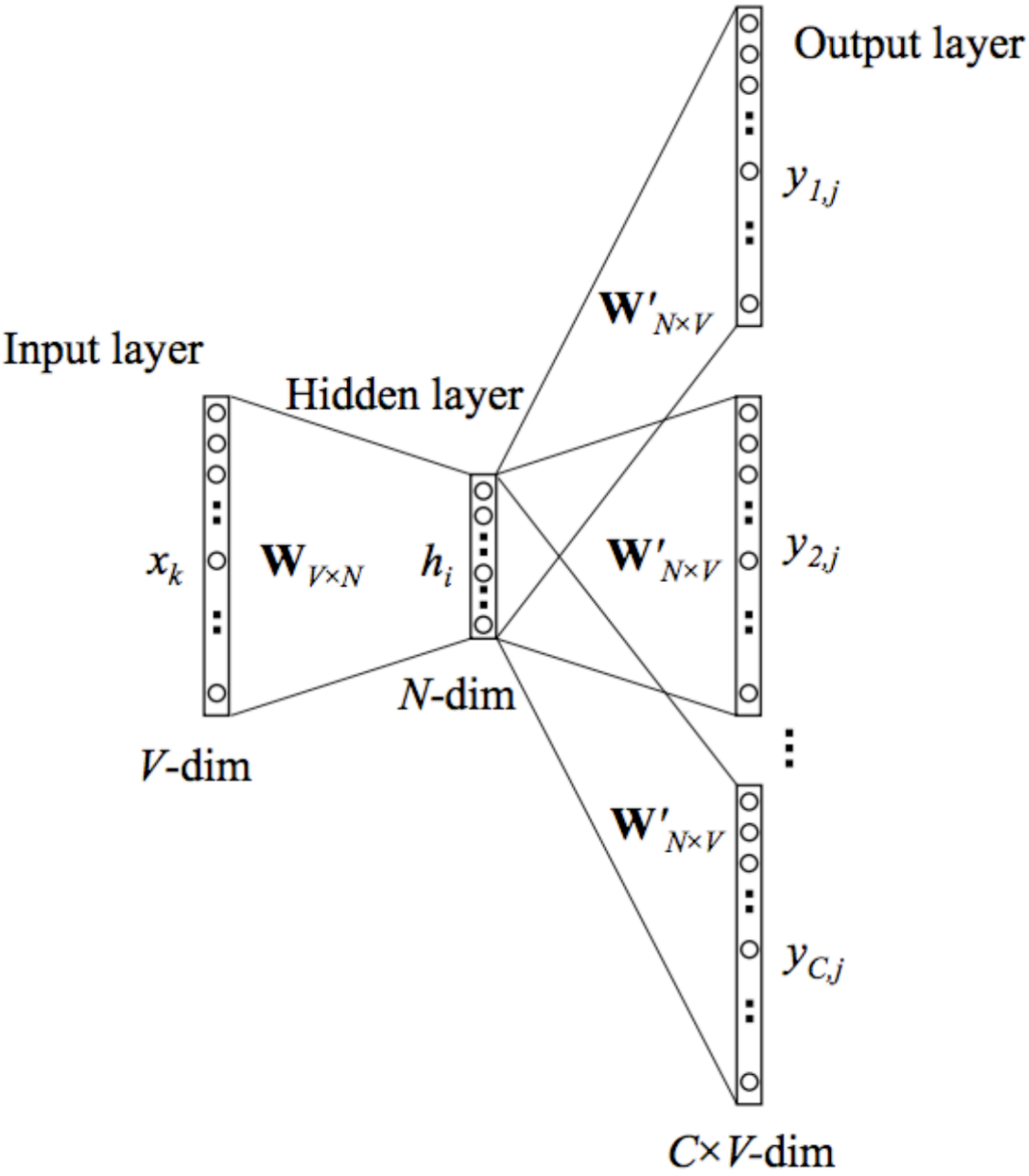
- 使用词袋子模型来表示每篇文档，常见的一种思路是首先将文本进行分词，也就是将一个文本文件分成单词的集合，建立词典。这里不打算尝试多元组构成词库，因为使用多元组构建词典会使得词典非常大，构建文本的特征表示时向量维数非常大不利用学习。构建词典之后，将每篇文档表示成长度为词典大小的一维的向量，其中词典的每个词对应向量中的一个位置，其数值使用该词对应的tf-idf值。文本中某个特征词的属性值的计算与下列因素有关。一个是特征词在该文本中出现的频率，特征词出现的频率越高，则权重值越大。另一个是特征词的文档频率，即包含该特征词的文本的个数，含有特征词的文本数目越多，则该特征词越普通，它对类别的区分作用越小，给它分配的权重也就越小。最后，考虑到文本的长短不一，要将文本长度归一化，这样同一个特征词在不同长度的文本中的权重才具有可比性。基于以上因素考虑，特征词的权重函数经常采用如下方式计算:

$$a_{ij} = \frac{tf(w_j,d_i)log(\frac{N}{df(w_j)})}{\sqrt{\sum_{k=1}^{M}[tf(w_k,d_i)log(\frac{N}{df(w_k)})]^2}}$$

  其中$tf(w_j,d_i)$表示特征词w_j 在文本d_i中出现的频率，$df(w_j)$表示了特征词的文档频率，N 表示文档总数，M表示特征总数，a_{ij} 表示了第 i 个样本矢量的第 j个分量值。这样能有效减少常见词的权重，增加特异词的权重。从而可以得到文档的特征表示。接下来，就可以方便利用机器学习分类模型进行训练。

- 利用Word2Vec方式即词向量模型表示每篇文档，这里面包含两部分主要工作：
  - 利用文本数据使用skip-gram的方法[3]进行训练，将每个单词表示成向量形式。文献[3]提出了两种算法，都是利用上下文的关系来训练出单词的向量表示。其中COBW是通过上下文来预测一个文字，而skip-gram是通过一个文字来预测上下文。可通过图示来清楚的表示：


Input layer
Output layer
Hidden layer
x_{1k}
$\mathbf{W}_{V\times N}$
x_{2k}
$\mathbf{W}_{V\times N}$
h_i
$\mathbf{W}'_{N\times V}$
y_j
N-dim
V-dim
$\mathbf{W}_{V\times N}$
x_{Ck}
$C\times V$-dim



本项目将主要采用skip-gram模型训练词向量。因为两种方法原理上类似，但查阅相关文献发现在数据量充足的条件下，skip-gram的效果要好。

词向量训练后需要进行简单评测，比如检验一些单词之间相似性是否符合逻辑等。为此将是用T-SNE技术将词投影在二维平面进行可视化。观察相邻词之间是否意思相近。

此外，针对上述提出的词向量模型的训练也需要大量数据，20类新闻数据样本量可能不足以训练出较好的词向量模型，为此我们将可以采用Mikolov曾经使用过的text8[6]数据包进行训练词向量模型。由于text8数据集较大，可以认为新闻组中的单词基本包含在text8中，所以讲只采用text8进行训练。

- 如何将词向量转化成整个文本的表示需要进一步探讨。这里将采用两种方式。
  - 将文档中的词向量进行平均得到整个文档的向量表示。同样地，使用机器学习分类模型进行训练。
  - 直接将文本表示成词向量序列，利用神经网络模型（如LSTM）处理词向量序列，进行分类。

在获得文本的表示方式之后，我们将利用机器学习模型训练数据进行分类。采用的机器学习模型有：逻辑回归、SVM、随机森林和神经网络。这些模型的优缺点如下：

- 逻辑回归
  - 优点：线性模型，简单，训练速度快，方差小，鲁棒性较好。
  - 缺点：过于简单，偏差较大，难以发现数据中的复杂模式。
- SVM
  - 优点：通过将数据投影到高维空间进行拟合，模型复杂，在数据量充足时，表现良好。
  - 缺点：训练速度慢，容易发生过拟合。
- 随机森林
  - 优点：决策树构成的集成模型。模型灵活，能拟合复杂的数据。
  - 缺点：超参数多难以调节。容易过拟合。
- 神经网络
  - 优点：模型特别复杂，灵活性大，能拟合非常复杂的数据。
  - 缺点：调参困难，训练时间长。

2.4 基准模型

基准模型将使用词袋子模型构建文档表示，并使用多分类问题的logistic regression作为分类模型。该基准模型能够清晰的定义并容易实现，可作为后续复杂模型的参照。该模型在测试集上的分类准确率为0.678.

3. 方法

3.1 数据预处理

我们使用利用*sklearn*工具包[5]下载20类新闻组，并且sklearn已经按照标准做法根据日期将数据集分为训练集和测试集了。对于新闻数据，文本具有清晰的结构和规范的语法，我们使用sklearn中的函数自动过滤掉标题、注脚和引用（由于这些内容与分类问题具有较强的关联性，而本项目的目的在于关注文档内容对分类的影响）。同时还将使用标准的文本预处理步骤，如分词、去除标点符号、转换成小写、去除不常用词、词性转换等。

3.2 执行过程

- 首先我采用词袋子模型对文本进行表示。其中每个特征的值为该位置对应单词的tf-idf值。这样能有效减少常见词的权重，增加特异词的权重。同时每篇文档有固定长度的特征表示，为101631维。随后采用多分类的逻辑回归、svm和随机森林进行分类。这里我使用scikit-learn工具包中的相应函数进行训练和预测。经过训练后三种模型在在训练集上的分类准确率分别是**0.896**、**0.954和0.971**，在测试集上的分类准确率分别是**0.678**、**0.663和0.396**. 需要说明的是这里svm使用的核函数为线性核函数。由于这里维数较大，使用径向基核函数会导致模型效率极低。
- 其次我尝试的模型为word2vec. 这种词向量的模型好处在于能够通过词向量之间的代数关系反应出单词之间的语义关系。我采用gensim[7]工具包对该新闻语料进行训练，这里每个单词被表示成100维的数值向量。训练后各单词之间的关系可以通过向量之间的相似性来表示。下面简单的举两个例子，观察词向量的训练效果。如与'woman'相似的词按相似性由大到小进行排列为

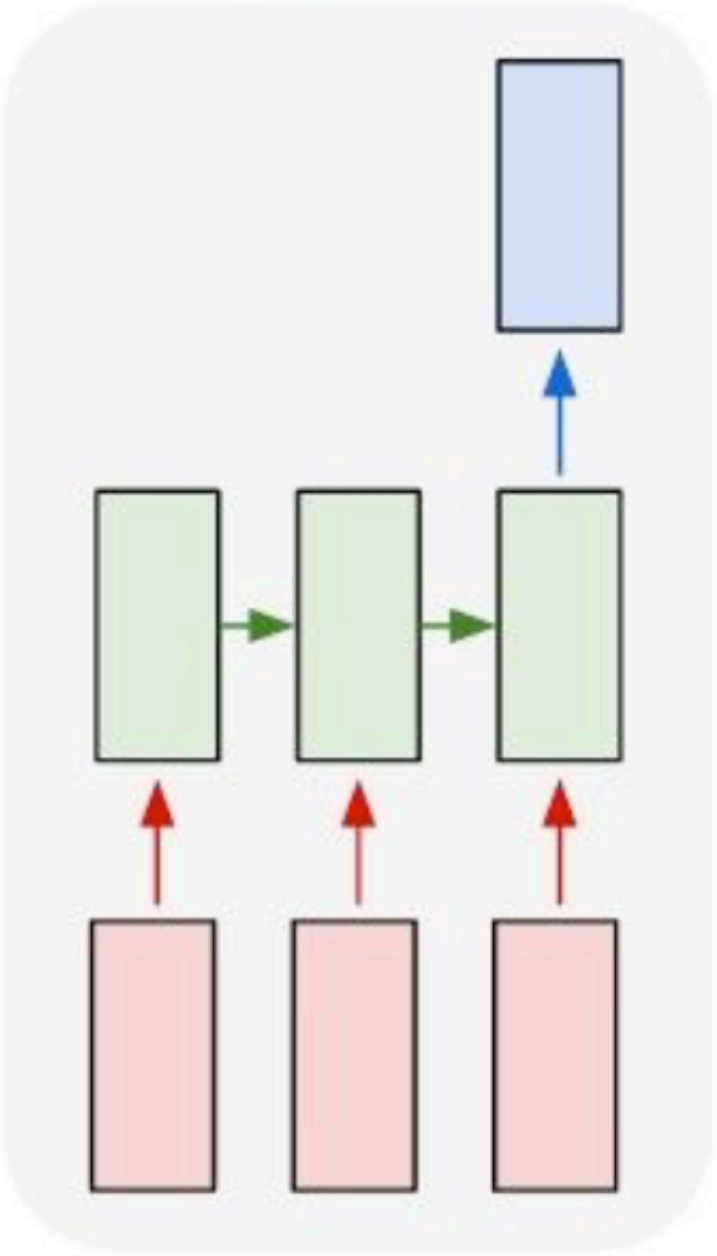
```
('man', 0.6217696070671082),
('her', 0.5826433897018433),
('she', 0.5783253908157349),
('person', 0.534965991973877),
('dog', 0.5204246044158936),
('women', 0.5027643442153931),
('girl', 0.4961436688899994),
('mother', 0.4921339750289917),
('children', 0.45686399936676025),
('friend', 0.45095932483673096)
```

可以看到最为相似的词为'man'，符合词义之间的联系。与'car'相似的词按相似性由大到小进行排列为

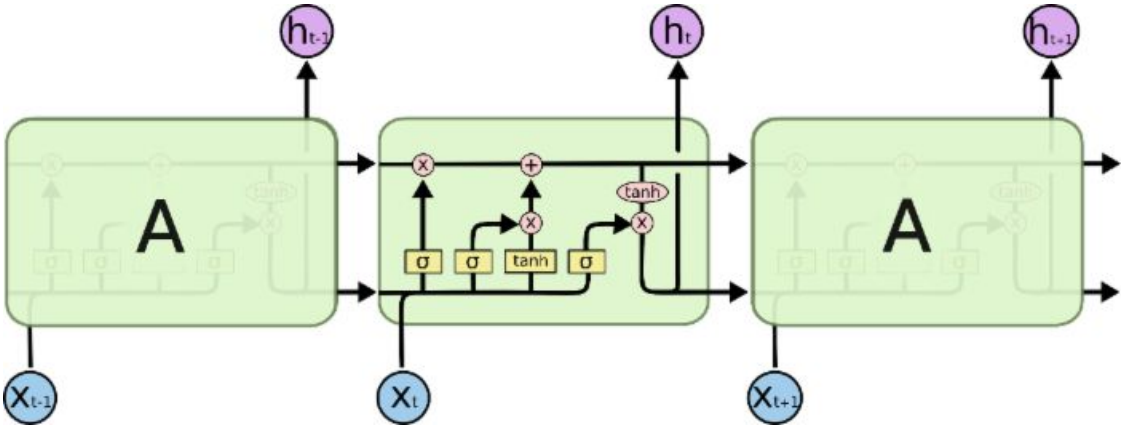
```
('bike', 0.7111523151397705),
('cars', 0.6475207805633545),
('engine', 0.5363672375679016),
('bikes', 0.5203389525413513),
('honda', 0.5200111865997314),
('tires', 0.5190979242324829),
('jacket', 0.5108001232147217),
('dealer', 0.5054090023040771),
('volvo', 0.4970071613788605),
('odometer', 0.49022600054740906)
```

可以看到最为相似的词为'man'，符合词义之间的联系。获取词向量后，使用两种方式对文本进行表示。

- 对任意一篇文本，将其包含的所有词的词向量进行平均，可以得这篇文档的数值向量。相比于词袋模型，该文档表示的维数大大降低，当时损失了不少的文本信息。之后仍然采用上述三种模型进行训练并预测。在测试集上的分类准确率分别是**0.591**、**0.422**和**0.289**。相比于之前测试，效果均有不同程度的降低。
- 对任意一篇文本，将其表示为词向量序列。随后采用能对序列数据建模的递归神经网络（RNN）进行训练。这里我将采用RNN的一个变种LSTM进行训练，LSTM中的记忆单词能有效解决RNN训练过程存在梯度消失的问题。为了便于训练，我将所有的新闻样本固定长度为200，采用了从头部截断或者头部添零的技术。之所以选择200是因为大部分文本长度集中0~300之间，选择200个单词可以保留文本的大部分信息，同时也能让LSTM进行有效的训练。文本过长会导致网络训练速度慢，使得训练不充分。这里我尝试过单层的LSTM、双层的LSTM和三层的LSTM进行训练，其中所有隐藏层的节点大小均为128，训练时固定词向量。其中单层LSTM的结构如下：



红色为输入序列，每个方块代表一个输入元素，绿色为隐藏层（双层lstm有两层绿色序列），每个方块表示一个cell，其结构如下：



由遗忘门、读入门、写出门控制信息的流动，解决RNN中梯度消失的问题。蓝色方块为输出，在该问题中表示输入每个类别的概率。最终双层隐藏层的LSTM模型的测试准确率较高，为**0.522**.

3.3 完善

基于上述的复杂模型的测试准确率均不如基准模型的表现效果。我采取以下两种方式进行改进。

- 对于复杂模型，其超参数的设置对其模型表现效果影响很大，所以我将采用网格搜索法和交叉验证来调试支持向量机、随机森林的超参数。通过设置合理的搜索范围，经过试验可以发现模型的效果有明显的提升。其中svm和随机森林最好的模型在测试集上的分类准确率为**0.663和0.631**. 其中随机森林提升较大。
- 为了探究文本表示的对分类结果的影响，我将对词向量模型进行改进。上述词向量模型效果一般的可能原因在于新闻组的语料大小有限，为此我将可以采用Mikolov曾经使用过的text8数据包进行训练词向量模型。训练后的词向量在二维平面的投影参见后面的模块。仍然使用上述的RNN进行训练，在测试集上的分类准确率**0.553**，相比于新闻组语料训练处的词向量模型效果略有提升。

4. 结果

4.1 模型的评价与验证

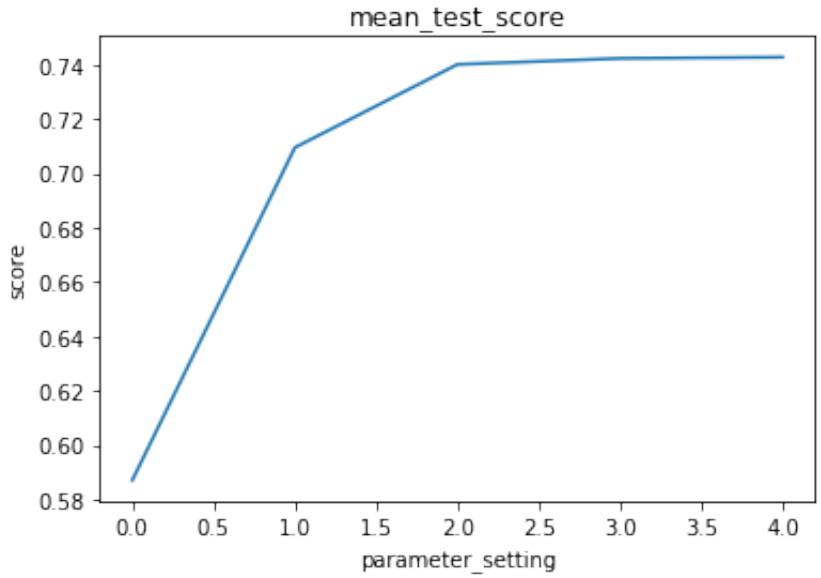
在不同的文档表示方式和不同的机器学习模型（经过超参数网格搜索优化之后）之下，训练集和测试集（括号中）上的分类准确率如下：

	词袋模型	**词向量-平均模型(新闻组)**	**词向量-平均模型(text8)**	**词向量-序列模型(新闻组)**	**词向量-序列模型(text8)**
逻辑回归	0.896 (**0.678**)	0.670 (0.591)	0.601 (0.537)	*NULL*	*NULL*
随机森林	0.974 (0.631)	0.970 (0.390)	0.970 (0.313)	*NULL*	*NULL*
SVM（linear）	0.954 (0.663)	0.750 (0.573)	0.668 (0.527)	*NULL*	*NULL*
RNN	*NULL*	*NULL*	NULL	0.717 (0.522)	0.690 (**0.553**)

从上述结果对比看出，可以看到基准模型的效果最为出色。为此我们通过对参数添加正则化($l_2 - norm$)，进一步衡量模型的稳定性。参数设定如下：

```
{'C': 0.1},
{'C': 1.0},
{'C': 10.0},
{'C': 50.0},
{'C': 100.0}
```

而交叉验证下的预测平均准确率如下图所示：

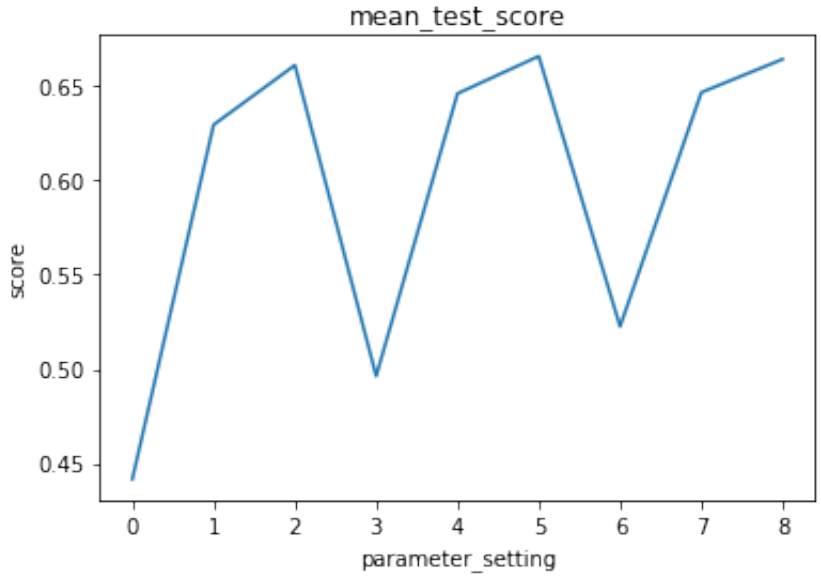


可以看到，在$C \geq 1.0$时模型的表现都很稳定。说明基准对超参数的设定不敏感。具有很强的鲁棒性。

另外可以随机森林和SVM都存在一定的过拟合的情况。相比于基准模型，这两种模型在测试集上的表现已经大致相当。为了进一步分析模型的稳定，这里我将考察随机森林在在不同超参数设置下的表现。这里的参数设定如下：

```
{'min_samples_split': 2, 'n_estimators': 10},
{'min_samples_split': 2, 'n_estimators': 100},
{'min_samples_split': 2, 'n_estimators': 500},
{'min_samples_split': 10, 'n_estimators': 10},
{'min_samples_split': 10, 'n_estimators': 100},
{'min_samples_split': 10, 'n_estimators': 500},
{'min_samples_split': 50, 'n_estimators': 10},
{'min_samples_split': 50, 'n_estimators': 100},
{'min_samples_split': 50, 'n_estimators': 500}
```

而交叉验证下的预测平均准确率如下图所示：



可以看到随机森林的波动较大，对超参数的设定比较敏感，不稳定。而且可以看到主要依赖于n_estimators的设定，n_estimators=500时模型效果较好。综合考虑模型的稳定性和训练速度，在使用词袋模型时，我认为使用逻辑回归进行分类效果要好。由于词袋模型的维度太高，这里使用基准模型作为标准模型是合理的，稳定性较高。

词向量平均模型会使得词义信息混淆，同时也不具有文档的语义信息，但是分类结果相比词袋模型只有略微的下降。这得益于两个主要因素：一是词向量模型能够挖掘词义之间的信息，使得这种粗糙的表示方法也能够传递文本内容的信息；二是词向量模型能大大降低文本表示的维度，提升模型的稳定性。从这里可以看出词向量模型有其特有的优势。

相反的两种神经网络模型的训练准确都不高，存在模型欠拟合的情况，测试集上的表现相比其他机器学习模型也要逊色。另外，实际情况是这两个模型训练时间均为十几个小时，说明神经网络在使用过程中极度依赖硬件资源，否则训练时间极长，影响模型表现。

综合以上多方面分析，该文本分类项目表现最为出色的为词袋模型和逻辑回归的结合。在20类分类问题中最终达到了67.8%的分类准确率。

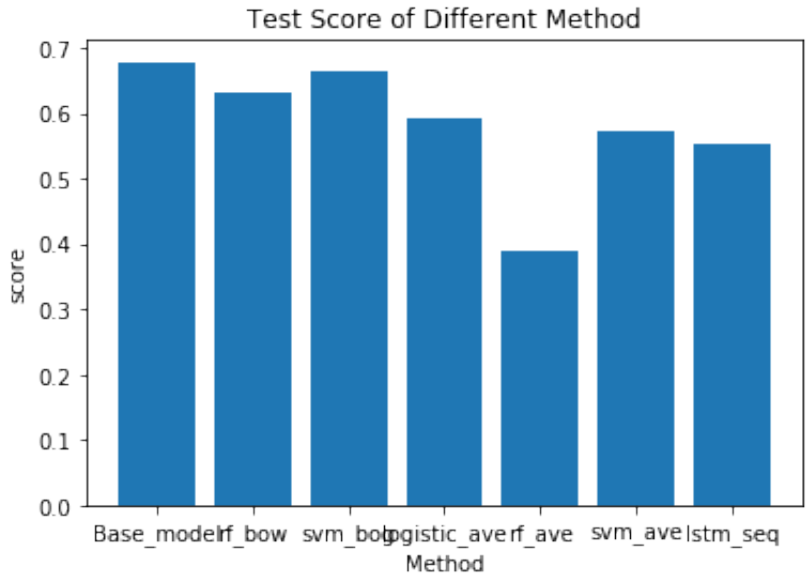
4.2 合理性分析

从上述模型的效果可以看到，虽然词向量模型能够刻画语义之间的关系，但是如何利用它对文本进行合理的表示是处理文本分类问题时的难点。仅仅进行词向量平均的话会损失并混淆很多信息，导致分类效果不佳。但在利用神经网络模型处理词向量序列是又会面临着网络结构的选择，超参数的调试和训练时间长等问题，这些因素制约了模型的效果。综合这些因素，可以看到即使词袋子模型有着维数高、表示系数、不考虑词序等诸多缺点，但是在配合简单的逻辑回归时，仍然能够得到很好的效果。

5. 项目结论

5.1 结果可视化

我将各个模型的表现用图表进行汇总表示如下：



可以看到其他复杂的表现效果均不如基准模型，诸多原因上面也进行了分析，兼顾到模型训练的时间，我认为基准模型非常适用于该文本分类的问题。在20类新闻组的预测中测试准确接近70%，是一个很不错的成绩，了解诸多模型的利弊，达到了我预期的目的。

5.2 对项目的思考

从前面的结论可以看出，词向量是一种有效的表示手段，但是如何通过词向量对整篇文档进行表示，使得文档的分类准确率得到提高，是一种难点。从目前尝试的方法来看，利用RNN对词向量序列进行分类是具有潜力的。关键在于如何设计出有效的模型和采用有效的训练方式以获得较好的泛化能力。

文档分类是一种古老但是仍旧活跃的问题，如何新的技术手段提高对文档分类的准确率是一个需要不断尝试和改进的过程。

在本项目中，我使用了20类新闻包数据，希望建立一种行之有效的模型，能够对文本数据进行分类。为了达到这个目标，我尝试了不同的文档表示方法，有直接通过词袋构建文本表示的方法，有通过词向量模型间接构建文本表示的两种不同尝试，尝试不同的机器学习算法并进行了相应的参数优化。最后确定了词袋模型和逻辑回归的组合模型能够快速的有效的完成文本分类的认为。但我仍然认为，随着数据量的增加、算法的不断完善和计算能力的提高，文本分类在复杂模型的作用取得更大的进步。通过本项目我并认为词向量模型结合神经网络模型在设计出合理的模型结构和合理训练的条件下应该是具有潜力的。但由于计算条件的显示，使得在本项目中该方法为得到充分的训练、参数也没能进一步优化。我会在后续的工作进一步尝试和探究。

5.3 需要作出的改进

实际上，句子、段落以及文章也可以引入向量的概念进行表达，称之为Doc2Vec，Mikolov的论文[7]有相应介绍。在后续的工作我将尝试直接对文本进行向量化表示并分类。

参考文献

[1]20新闻组: http://www.qwone.com/%7Ejason/20Newsgroups/

[2]NewsWeeder: Learning to Filter Netnews:http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.22.6286

[3]词袋子模型: http://www.cnblogs.com/platero/archive/2012/12/03/2800251.html

[4]Word2Vec:http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

[5]*sklearn*工具包:http://scikit-learn.org/stable/datasets/twenty_newsgroups.html

[6]text8:http://mattmahoney%5B.NET%5D(http://lib.csdn.net/base/dotnet/dc/text8.zip

[7]Distributed Representations of Sentences and Documents: https://arxiv.org/pdf/1405.4053v2.pdf