文档分类

夏鲁豫 2018 年 3 月 15 日

问题的定义

文本分类是自然语言处理的一个重要任务，应用场景非常广泛，比如对新闻进行自动分类，影评情感分析，垃圾邮件检测等等。

本项目目的就是利用自然语言处理技术结合所学机器学习知识对文档进行准确分类。

分析

数据探索

20 类新闻分类数据来自 http://www.qwone.com/~jason/20Newsgroups/

数据格式使用的是官网推荐的 20news-bydate.tar.gz 这个数据包，里面包含了分割好的训练集和测试集，并且是原始文本，按照类别，共有 20 类，数据总量为 18941 个，训练集和测试集分别为 60%和 40%，解压后总大小为：34.1MB

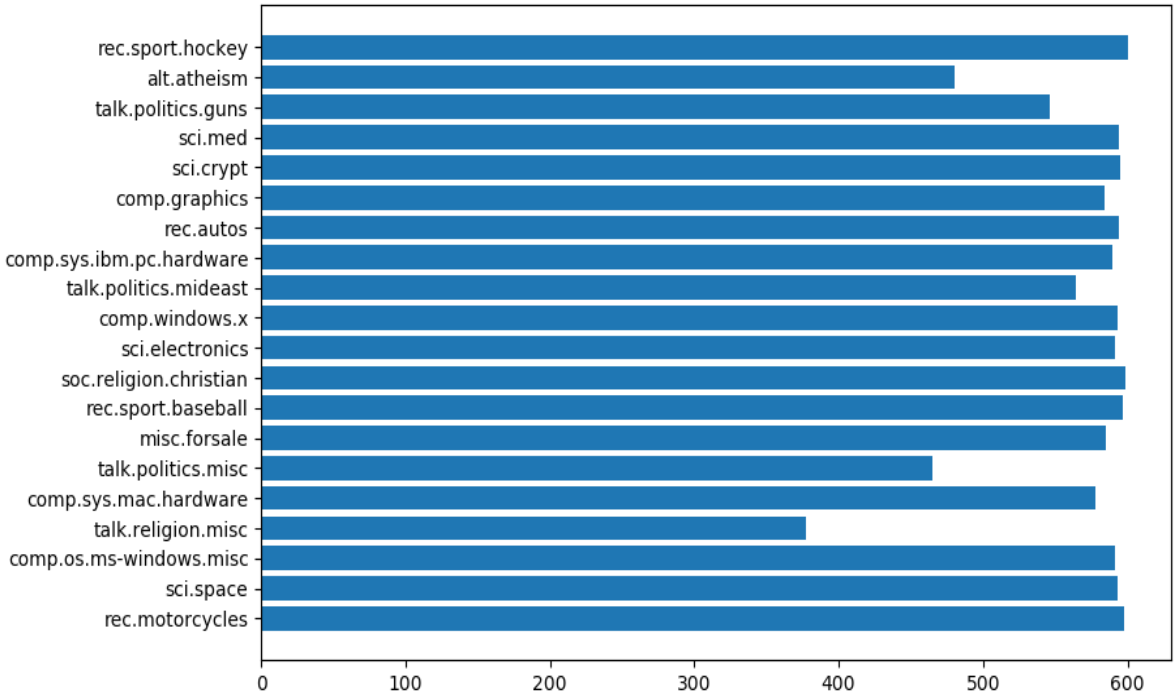
数据样本摘抄如下：

```
From: I3150101@dbstu1.rz.tu-bs.de (Benedikt Rosenau)
Subject: Re: Gospel Dating
Organization: Technical University Braunschweig, Germany
Lines: 93

In article <65974@mimsy.umd.edu>
mangoe@cs.umd.edu (Charley Wingate) writes:

>>Well, John has a quite different, not necessarily more elaborated theology.
>>There is some evidence that he must have known Luke, and that the content
>>of Q was known to him, but not in a 'canonized' form.
>
```

>This is a new argument to me. Could you elaborate a little?

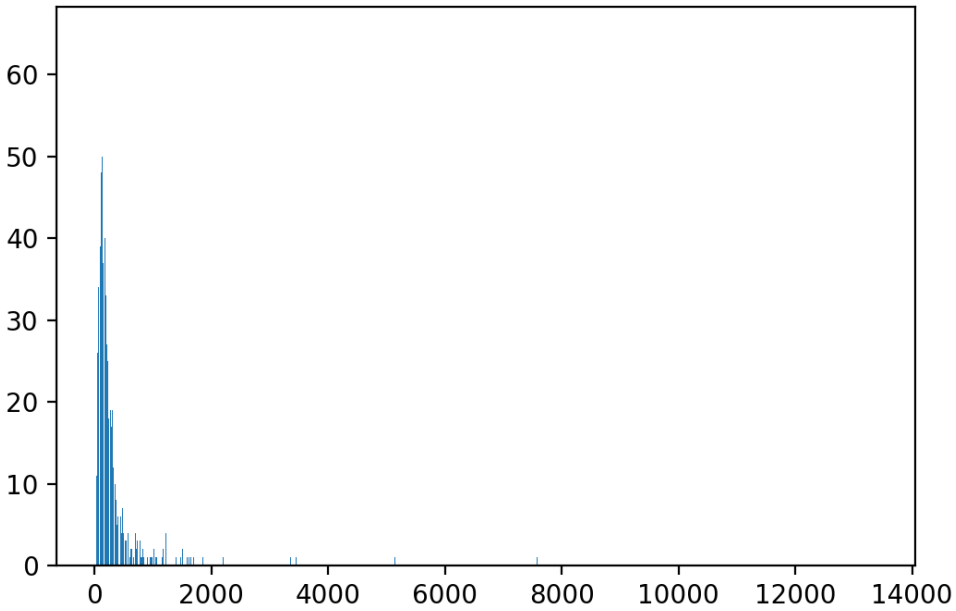
探索性可视化

总共 20 类数据，每类的分布如图所示：

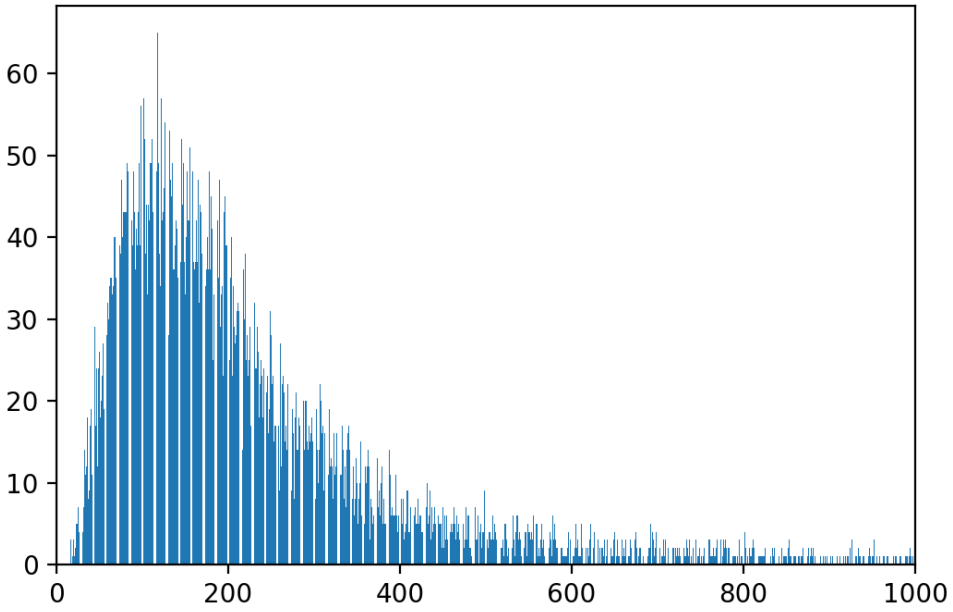


可以看出数据分布比较均衡，没有出现数据失衡的问题。

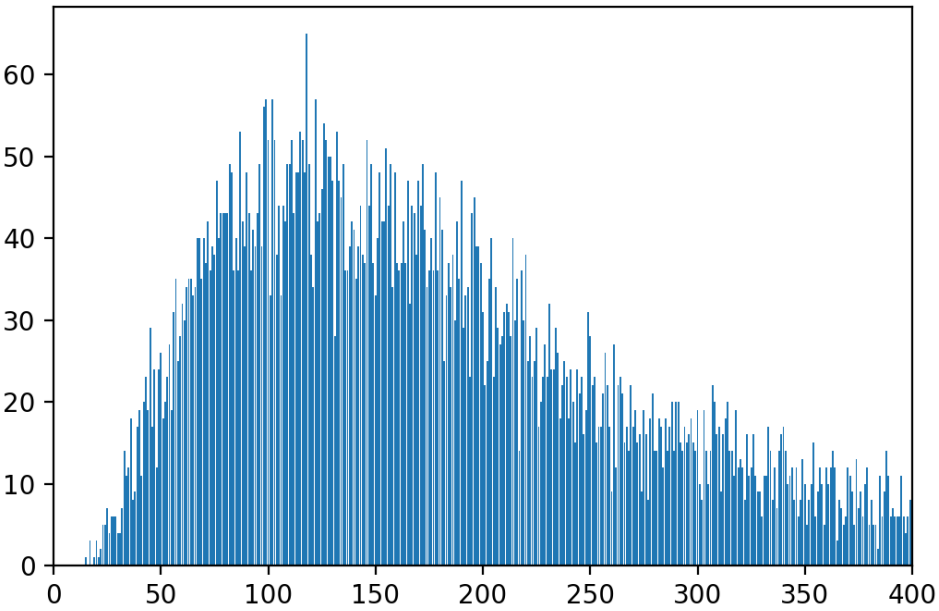
每个样本的词数是不一样的，统计一下词数分布：



从上图看出，词数分布比较偏，下面统计 0-1000 之间的分布：



可以看出词数主要集中在 0-400 之间，下面看下 0-400 之间的分布



可以从上面几幅图看出，样本中的词数主要集中在 50-400 之间

算法和技术

词向量

对文本进行分类，首先要解决一个问题就是用什么作为特征，对于这个问题，很自然的就想到用单词（对于英文来说）作为特征，因为文本是由一个一个单词组成的，但是单词却不是数字，所以需要用数字来表示这些单词，很自然的就考虑到使用 One-Hot 编码的方式对文本中的单词进行编码，但是 One-Hot 编码方式存在一些问题

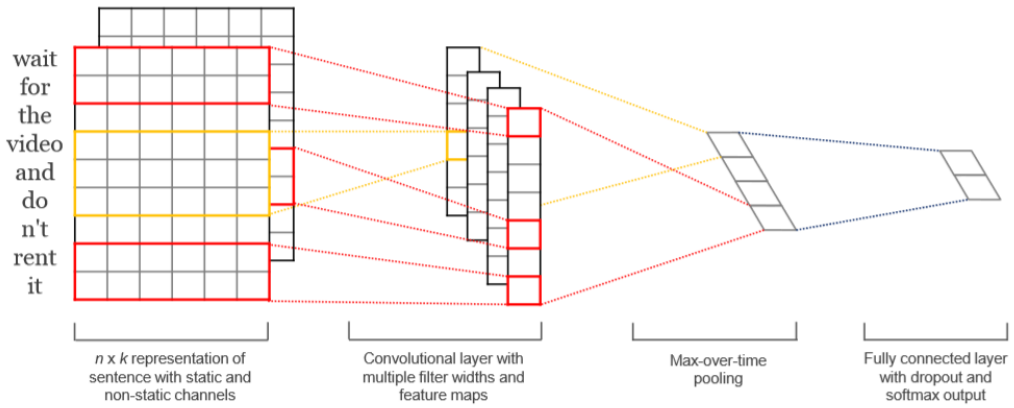
1. 当词汇量比较大时，One-Hot 编码后的数据太过庞大，对计算带来很大的困难
2. One-Hot 编码只仅仅编码了不同的单词，但是并没包含单词之间的信息，因为任何两个单词都是正交的。

对于One-Hot编码的弊端，在这里采用词向量的编码方式，词向量可以解决One-Hot 编码的两个问题，词向量是稠密矩阵，有效的压缩了数据量，词向量可以表示词与词之间的相似度，可以通过两个词向量的余弦相似度来对比两个词的相似程度。

当前主流的词向量方式有 word2vec，Glove，fasttext 等，其中 Glove 使用了词与词的共现信息，而 fasttext 模型则将一个词分成不同的子词，使用子词向量来表达整词向量，这样可以有效解决未在训练集出现过的词的向量。

TextCNN 模型

在模型选择上，除了传统的机器学习方法，比如贝叶斯，决策树和神经网络，近几年来涌现出了令人激动的深度学习，在这个任务中，我使用卷积神经网络来解决文本分类的问题，卷积神经网络已经在图像领域证明了其强大，令人意想不到的是卷积神经网络同样能够用于文本分类，主要原因是卷积核可以捕捉局部相关性，这样就可以捕捉到文本中的一些关键信息。



TextCNN 模型就是通过 CNN 网络来进行文本分类的一个网络模型，通过 CNN 网络对文本进行特征提取，然后在通过全连接层进行分类。

基准模型

基准模型采用 sklearn 数据使用页面所提供的示例代码作为基准模型，其在测试集中的准确率为 77%

在 TextCNN 模型中，目标是测试集的准确率达到 85%

评估指标

因为这是分类任务，所以采用准确率作为评估指标

$$\mathrm{Acc} = \left(\frac{1}{m} \sum_{i=1}^{m} 1\{P(i) = y\} \right) * 100\%$$

M 为样本总数，P 为模型，P(i)为模型输出预测分类

1{}为示性函数，例如：1{值为真的表达式} = 1

方法

数据预处理

1. 去除无效字符

由于原数据是原始数据，所以需要对该数据进行筛选，只需要其中的单词，特殊字符和标点符号是不需要的。

2. 截断补齐

由于使用的是 CNN 模型，所以需要对文本数据进行截断补齐，从上面分析得知，数据的单词数主要集中在 50-400，所以这里将一个样本的单词长度设为 400，不足 400 的使用字符</s>进行补齐，超过 400 的取前 400 个单词。

3. 建立词典和词向量

接下来需要建立一个词典，用来从单词到数字的映射。为了更好的表达单词，在这里使用 fasttext 词向量，而且使用预训练的词向量。

需要注意，本项目使用的是官网提供的原始数据，而文本的编码格式并不是 UTF-8，而是 windows-1252，所以需要注意读取的编码格式。

执行过程

模型结构

采用 TextCNN 网络模型，模型的网络结构和参数如下图所示。

```
HybridSequential(
  (0): Embedding(83531 -> 300, float32)
  (1): ReshapeInput(

  )
  (2): ConvConcat(
    (net1): HybridSequential(
      (0): Conv2D(1 -> 100, kernel_size=(3, 300), stride=(1, 1))
      (1): MaxPool2D(size=(398, 1), stride=(398, 1), padding=(0, 0), ceil_mode=False)
    )
    (net2): HybridSequential(
      (0): Conv2D(1 -> 100, kernel_size=(4, 300), stride=(1, 1))
      (1): MaxPool2D(size=(397, 1), stride=(397, 1), padding=(0, 0), ceil_mode=False)
    )
    (net3): HybridSequential(
      (0): Conv2D(1 -> 100, kernel_size=(5, 300), stride=(1, 1))
      (1): MaxPool2D(size=(396, 1), stride=(396, 1), padding=(0, 0), ceil_mode=False)
    )
    (net4): HybridSequential(
      (0): Conv2D(1 -> 100, kernel_size=(6, 300), stride=(1, 1))
      (1): MaxPool2D(size=(395, 1), stride=(395, 1), padding=(0, 0), ceil_mode=False)
    )
  )
  (3): Dropout(p = 0.5)
  (4): Dense(400 -> 20, linear)
)
```

通过 Embedding 层来对文本数据进行向量化，然后后面有四个卷积层和最大池化层的并列组合，四个不同大小的卷积核可以捕捉到不同尺度的特征信息，通过这四个卷积层后，输入全链接层进行分类，在卷积层和全链接层之间加入了一层 Dropout 防止过拟合。

超参数选择

优化算法采用自动优化算法 Adam，Adam 优化算法可以自动调整学习率，这样在训练前期的学习率会大一些，能够快速是收敛，而在后期需要小学习率来稳定收敛。

Bach Size 选择 8，因为太大不仅训练速度会减慢，而且收敛速度也会变慢，太小收敛会发生震荡，测试了 8，16，32 几个，发现 8 的效果最好。

使用预训练的 Fasttext 词向量

在词向量上面，我选择了使用预训练 fasttext 词向量，通过预训练的词向量编码后，每个单词会映射成 300 维的向量，通过词向量，可以对比词与词之间的相似度，例如和 school 最相似的五个词分别是'schools', 'graduation', 'education', 'postgraduate', 'student'，可以看出预训练的词向量能够正确表示出词与词之间的相似度。

在不使用词向量的情况下，模型会通过学习 Embedding 的权重向量来实现词向量，但是由于数据集有限，所以学习的词向量可能不能很好的表示词与词之间的关系。所以在与使用预训练词向量的训练方式对比中，虽然不使用模型也能快速收敛，但是测试集准确率会在 0.82 震荡，始终无法达到 0.85，而使用 fasttext 预训练词向量，模型会在 0.84 左右震荡，但最终会达到 0.85。

完善

将单词数进行截断补齐后，比使用最大长度的模型训练时间和空间效率提升 很明显，而且模型准确率也提升很高。在使用预训练词向量后，模型会比不适用预

训练词向量准确率提升约 0.1，对比如下表：

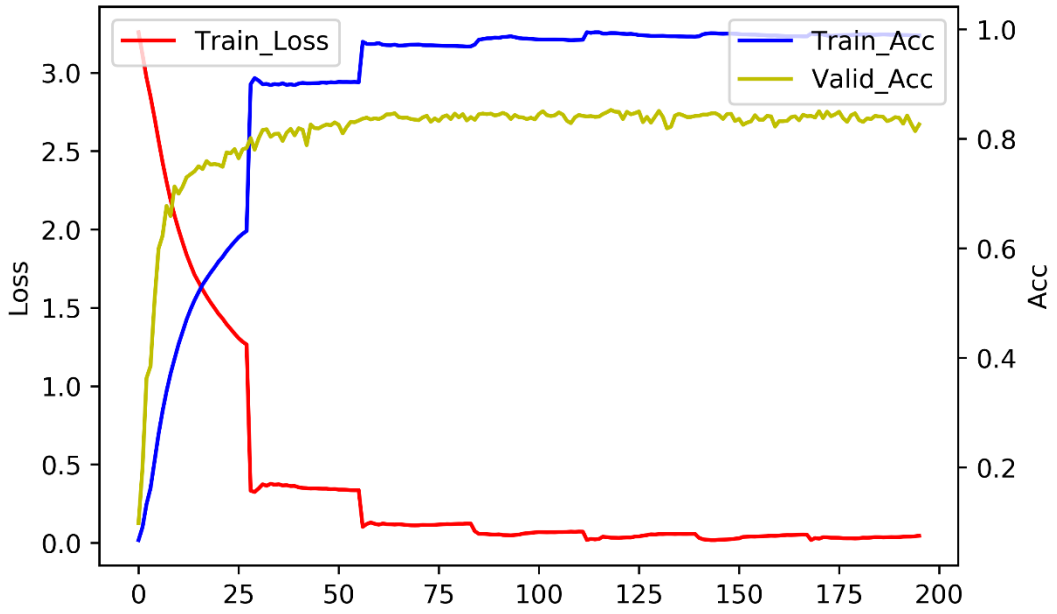
模型	准确率	训练时间
使用最大长度补齐并且不使用预训练词向量	81.6%	约 3h
使用 400 个单词长度截断补齐不使用预训练向量	84.3%	约 2h
使用 400 单词截断补齐使用预训练向量	85.2%	约 2h

结果

模型的评价与验证

最终训练的 TextCNN 模型在测试集的准确率为 0.852，已经达到预期目标

训练图如下



但是该模型的训练时间和训练空间效率上并不理想。与基准模型对比如下：

模型	测试集准确率	训练时间
TextCNN	85.2%	约 2h
基准模型	77%	<2min

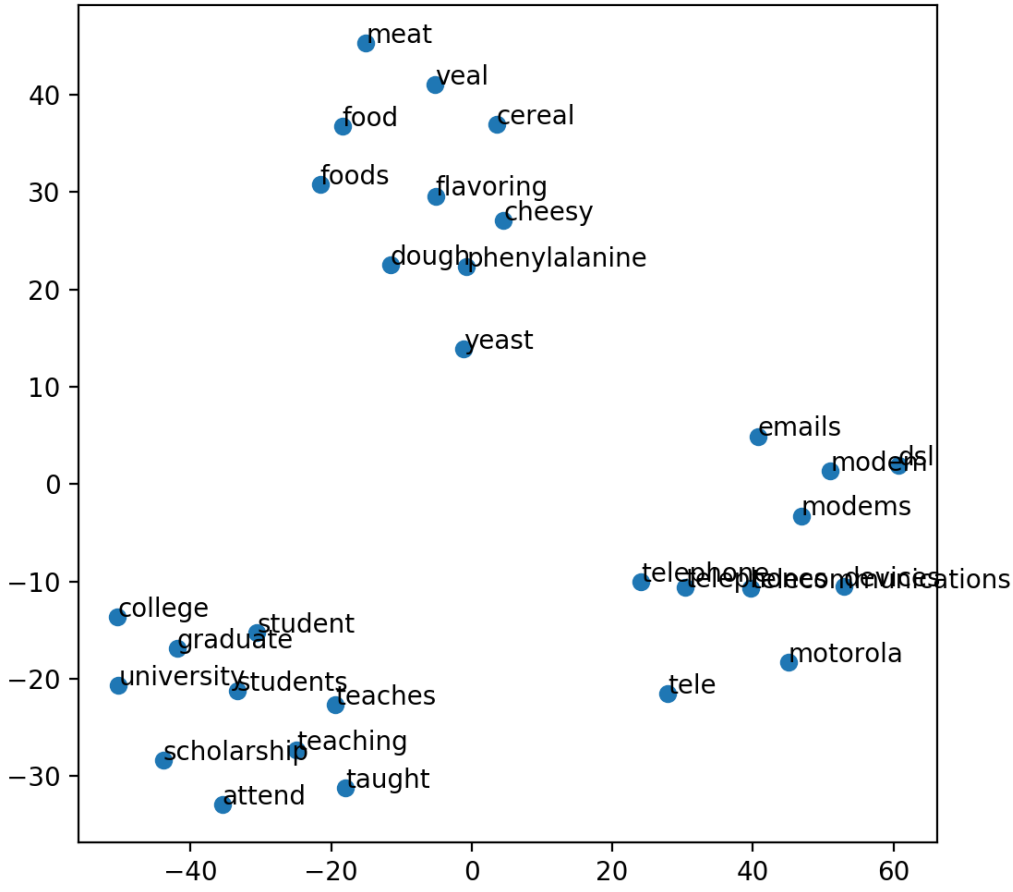
合理性分析

从模型的性能来看，模型的测试集分类准确率达到 0.85 算是及格，但是模型的训练时间比较长，所以综合来说，目前的 TextCNN 模型相较于传统的模型并无多大优势，但是在大规模文本数据中，TextCNN 借助 GPU 运算可以获得更好的训练速度和准确率。综合来说，TextCNN 模型基本满足预期。

项目结论

结果可视化

通过 t-SNE 算法对预训练词向量进行降维压缩，然后分别选择和 milk，phone，school 最接近的 10 个词可视化出来，可以明显的看出，这三组词在图上也分成了三族，说明预训练词向量能够表示词与词之间的关系。



对项目的思考

本项目主要针对自然处理中的一个分类任务，并使用深度学习的方式来解决，通

过对数据预处理，模型搭建，最后训练调优最后得到的模型在准确率上达到预期目标，但是在训练时间上却不太理想。深度学习在于可以不手工提取特征，而且适用于大规模文本数据。而词向量方式可以很好的解决 One-Hot 向量无法表达词与词之间的关系这一弊端，而且通过预训练词向量可以提高模型性能。

需要做出的改进

在这个项目中，使用的是基本的 TextCNN 模型，可以通过加入注意力机制来提升模型性能，而且可以使用更大数据集训练的预训练词向量来提升模型性能，以及可以使用 RNN 和 CNN 以及注意力机制结合的方式，来进一步提升分类准确率。在后续，会考虑使用 RNN 以及注意力机制来获得更好的性能。

参考文献

FastText: https://arxiv.org/pdf/1607.04606.pdf

TextCNN https://arxiv.org/pdf/1408.5882.pdf

Visualizing Data using t-SNE

http://prb.tudelft.nl/sites/default/files/vandermaaten08a.pdf

Adam: A Method for Stochastic Optimization https://arxiv.org/pdf/1412.6980.pdf