机器学习纳米学位毕业项目

基于20 Newsgroups的文档自动分类系统

郭晨晨 优达学城

2017年6月29日

I. 问题的定义

项目概述

随着世界节奏的加快以及信息量的爆炸性增长，人类产生的各种文档、语音和图片信息也在飞速的增长。正确的把每个文档按照需求归类在越来越多的场景需要被用到，比如“今日头条”和“google now”把新闻分类为政治、经济、军事等。对于现在每天产生的海量文档如果只靠人工甄别显然已经不能满足要求，所以文档自动分类已经是一个被广泛研究的NLP问题了，它的目的就是通过计算机可以准确快速的将文档识别到一个特定的类别中。在这过程中可能会用到文章的内容、题材、关键词以及作者等等属性。

本项目立足于算法层面，准确的说是利用机器学习的方法构建一整套算法体系实现对文档库的自动分类。机器学习领域有许多分类算法，比如朴素贝叶斯、支持向量机和神经网络，每一种都有它的适用领域和优缺点。

网络上有许多公开的已经过人工标记的语料库，比如CMU的20 newsgroup和Matt Mahoney的text8。本项目中因为考虑到硬件资源的限制，我们选取了相对比较少的前者作为实验数据集。20 newsgroup含有20个类别共20,000条新闻，它的分类是基于新闻的内容,但是它的分类颗粒度过于细致，从大类比如计算机类、科学类到子类windows、mac、电学、医药，再到子类的子类硬件类、软件类，并不完全符合现在人们在各种新闻客户端上看到的常规分类。以今日头条为例它对于新闻的分类主要有时政类、经济类、体育类、娱乐类、科学类等基本上是和*20newsgroup*中的大类对应的，并不会把体育类再自动分类为篮球、网球等等，所以为了更贴近现今真实的新闻自动分类需求，在分类算法中我们只选择其中分属不同大类的四类数据来进行验证。

问题陈述

针对20 newsgroup语料库的自动分类是一个监督学习问题。我们的目的是**设计出一个高效的算法模型，通过在测试集上的训练后可以对语料库中的四类文档（soc.religion.christian, comp.os.ms-windows.misc, rec.sport.hockey, sci.space）进行准确的分类**。之所以选择这四类是因为它们分属soc、comp、rec、sci四个不同的大类，而且这四个类别的测试集数量相对较多可以进行更为详细的测试。

我们会首先分析预料库中的文档信息，对符号等进行一定的预处理并剔除掉可能存在的异常数据。然后使用doc2vec[1]方法训练所有20类新闻文档的训练集部分，得到每个文档的一个向量表示，再把这个向量通过全连接神经网络分类算法训练出一个分类模型。doc2vec是一种针对文档的，类似于word2vec[2]的词向量计算方法。它是目前三种用词向量表示文档的方法，平均值法、词向量聚类法和doc2vec之一。根据论文《基于 Word2Vec 的一种文档向量表示》[3]该方法最好所以选用。最后再用验证集中上述已选择好的四类的文档数据验证整个自动分类算法对这四类文档的分类准确性。

评价指标

- 本项目是一个多元分类问题，针对这类问题我们决定采用最常用的分类结果准确度来衡量算法的优劣，即分类正确的文档占总测试文档的百分比

$$accuracy = \frac{\sum_{i=1}^{n}(y^i = \overline{y^i})}{n}$$

n表示测试文档的数量，y^i表示第i篇文档的分类标签真值，$\overline{y^i}$表示模型预测的第i篇文档的分类值。当预测值和真值相等则为1.

- 同时为了避免可能存在的数据偏斜严重的问题，我们还会考虑F_1值，也就是查准率和查全率的权重结果

$$F_1 = \frac{2PR}{P+R}$$

- 因为对于分类问题最重要的是准确度，所以这里我们并不把算法的运算时间作为指标。而且好的类似于深度学习算法都会训练很久，相对于更高的准确度，一次训练的时间长短并不非常重要，只要它不是类似于指数型增长的。所以我们会记录算法运行的时间。除非超过一定的阈值，否则以算法精确度为准。

II. 分析

数据的探索

问题中涉及到数据集 *20 Newsgroups*，以下探索该数据集。

2.1 数据来源

已经有学者已经将 18000 条新闻文本分为 20 类主题，并提供了多种版本的新闻包以便下载。本项目中选取了其中经过整理、分割为训练集与测试集的新闻包[4]。

2.2 数据特征

20 Newsgroups 具备如下特征：

- 每条新闻均被研究人员标注为 20 个主题中的 1 个（即任何样本的分类都是单一的）
- 总数据集包含 18846 条新闻，总共被分割为 2 个子集：
  - 训练集（占总数据 60%）
  - 测试集（占总数据 40%）
- 剔除了跨主题的新闻（即任何一份新闻都只在单一主题下），提出了新闻组相关辨认标识（如 Xref, Newsgroups, Path, Followup-To, Date）
- 对于文本处理而言仍不够干净：除了由小写字母 a-z 组成的单词、单一空格以外，还有一些标点符号，如 `@`、`.`、`*` 等。因此在最终训练前，需要对该数据进行清洗。

2.3 样本展示

如下为其中的一份训练样本：

```
From: randy@megatek.com (Randy Davis)
Subject: Re: A Miracle in California
Article-I.D.: megatek.1993Apr5.223941.11539
Reply-To: randy@megatek.com
Organization: Megatek Corporation, San Diego, California
Lines: 15

In article <1ppvof$92a@seven-up.East.Sun.COM> egreen@East.Sun.COM writes:
|Bikers wave to bikers the world over.  Whether or not Harley riders
|wave to other bikers is one of our favorite flame wars...

  I am happy to say that some Harley riders in our area are better than most
that are flamed about here: I (riding a lowly sport bike, no less) and my
girlfriend were the recipient of no less than twenty waves from a group of
at least twenty-five Harley riders.  I was leading a group of about four
sport bikes at the time (FJ1200/CBR900RR/VFR750).  I initiated *some* of the
waves, but not all.  It was a perfect day, and friendly riders despite some
brand differences made it all the better...

Randy Davis                          Email: randy@megatek.com
ZX-11 #00072 Pilot                         {uunet!ucsd}!megatek!randy
DoD #0013
```

如下为其中另的一份测试样本：

```
From: les.tom@idcbbs.com (Les Tom)
Subject: Replacement for Program M
Distribution: world
Organization: IDC BBS - Alameda, CA - 510-865-7115
Reply-To: les.tom@idcbbs.com (Les Tom)
Lines: 33

          Maw Ying yuan wrote •
 ..............................
...From: yuan@wiliki.eng.hawaii.edu (Maw Ying Yuan)
...Subject: Replacement for Program Manager and File Manager?
...Message-ID: <C68G1G.JuJ@news.Hawaii.Edu>
...Date: Thu, 29 Apr 1993 06:44:04 GMT
...
...replacements for Win3.1's Program Manager and File Manager?
...yuan@wiliki.eng.hawaii.edu :)
 ..............................
    Hi,
      I've been using Plannet Crafter's "Plug-in" for Program Manager.
      Its listed in most BBS'es as PLUGIN13.  Its an add-on which gives
      some needed features to Progman; such as the ability to better
      manage your groups, change your cursor and icons on the fly,
      constant status of RAM and resources, "Quick-menu" (a drop-down
      menu of DOS commands or app launcher), plus some other neat stuff.
      Used it, liked it, and even reg'd it <G>.

      If you can't find it anywhere, let me know and I'll zip it up and
      mail it to you (shareware version, of course <g>...)

Aloha...
---
 .. .es .. All hope abandon, ye who enter messages here.
---
 . SLMR 2.1a .

----
+--------------------------------------------------------------------------+
| IDC BBS    510-865-7115 (USR HST 14.4k)  510-814-8097 (USR DS v.32bis)    |
| Alameda, California  -   Home of KingMail, KingQWK and QWKMerge           |
+--------------------------------------------------------------------------+
```

从上面样本可以看到文档中含有一些不常用或者不规范的字符，在建立字典时如果不加以处理很可能会引发问题，数据预处理的时候需要格外的注意。并且文本中含有大小写字母，因为我们决定采用词向量模型所以也需要多大小写字母进行过滤，保证同一个词在字典中只出现一次，这样有利于减少维度和提高准确率。

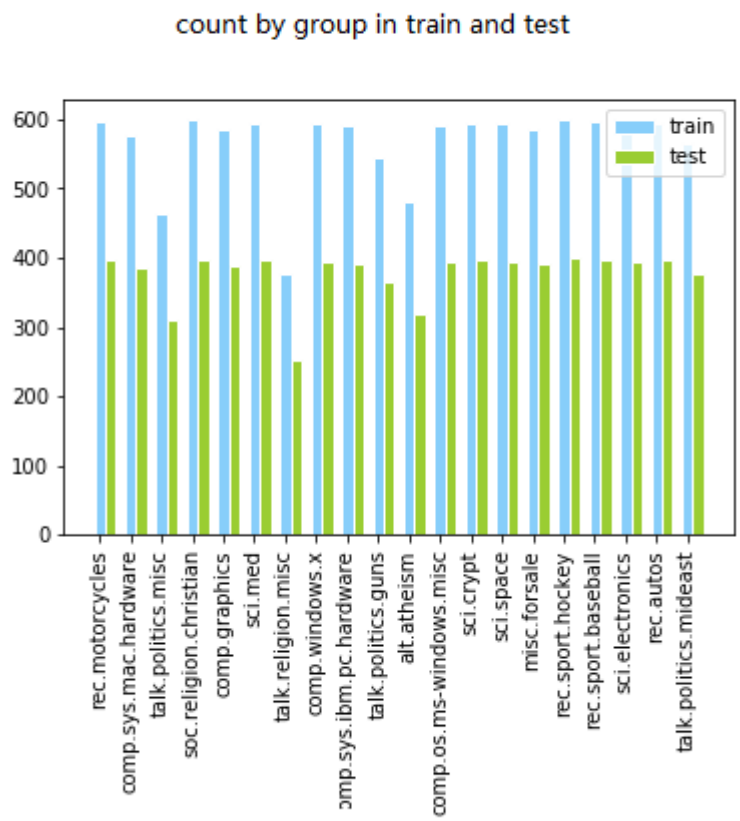
在对*20 Newsgroups*数据集进行字符过滤之后，我们获得了只含有小写字母和空格的纯净数据以用于后面的算法计算。处理后的数据各统计信息展示如下：

文档总数	**18846**	每篇文档字数（**mean**）	**324**
训练集文档数量	11314	每篇文档字数（std）	721
测试集文档数量	7532	每篇文档字数（max）	33450
数据集字数总和	6114843	每篇文档字数（min）	13
数据集单词数量	105481	每篇文档字数（25%）	128
每篇文档字数（median）	199	每篇文档字数（75%）	321

从上表可以看到全部数据集共有6114843个单词，数据量比较大，对于训练词向量我们认为是可以的。每篇文档的平均单词量是324个，符合新闻的一般文字数。75%分位的数量是321也说明大多数文档的字符数还是比较少的。对于最多的单词数达到33450，我们开始觉得可能会有异常需要处理的需要，不过在查看了该文档后发现是符合要求的，所以予以保留。

探索性可视化

因为目标是对文档分类所以我们最关心的是各类文档的数量以及测试集和训练集的分布情况，另外还有每篇文档的文字数量会在后面数据预处理的时候详细说明。这里展示文档数量分布图

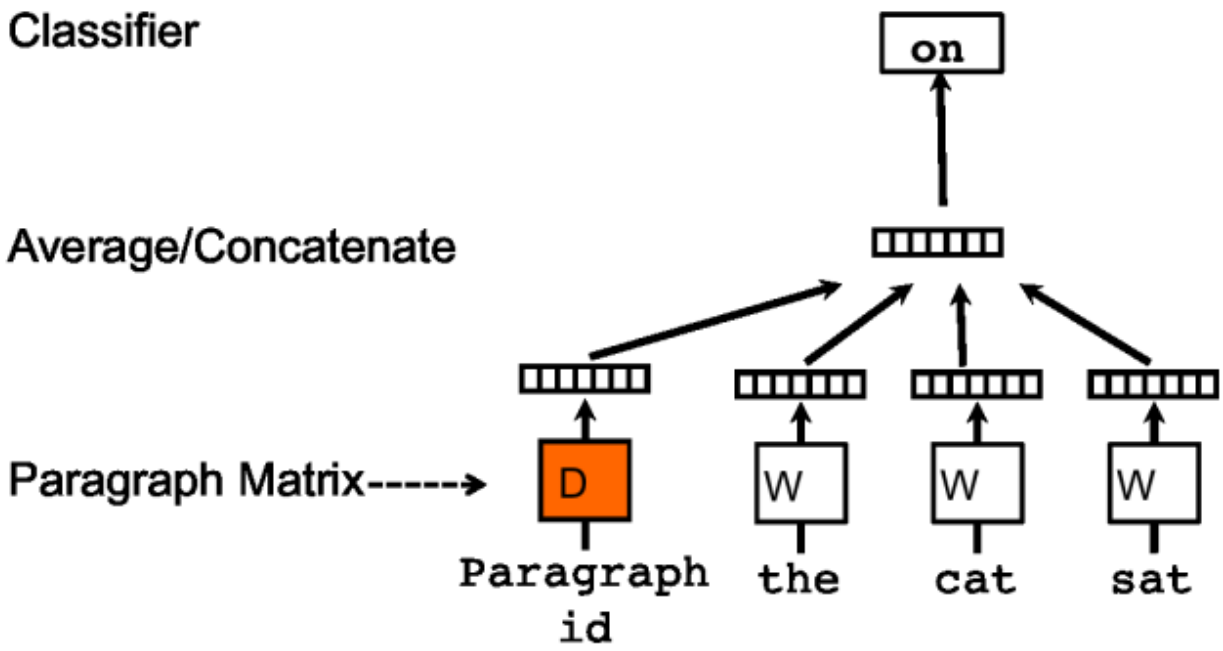


从图中很容易看出有三类文档的数量明显偏少，正常的每类文档测试集和训练集的数量分别为400篇和600篇，比例都是4/6。也就是如数据提供者所说的那样所有数据都已经被合理的分配和放置了。明显偏少的类别应该是数据提供者过滤无效文档时删除掉的，猜想原始数据这三个类别的文档质量不高，但即使如此这三类文档的训练集数量也有将近400条，符合算法要求无需特殊处理。测试集和训练集的比例为4/6，相对于一般常用的3/7比例稍微多了一些。这样可能会使得训练数据不够多，模型没有优化好。但是从另一方面如果在这样比例下模型仍然有很好的表现，则反映了它的鲁棒性很强，所以我们觉得不做处理。后续如果有需要可以考虑移动部分测试集的文档到训练集来提供更多的训练。

算法和技术

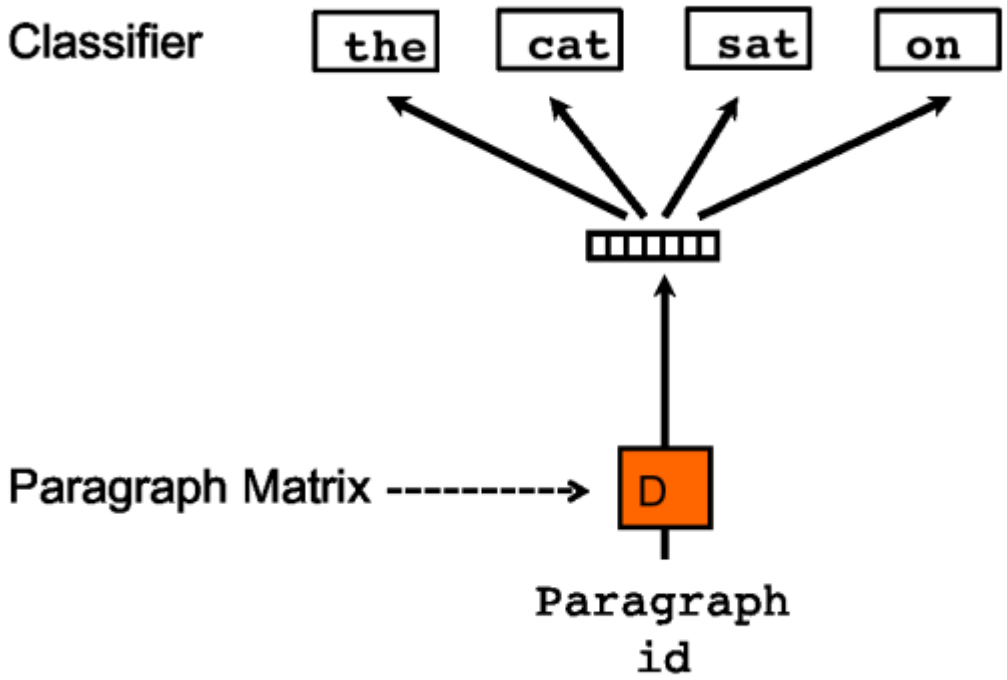
首先我们需要对*20newsgroup*的数据集进行预处理，主要是去除掉多余的符合以及统一大小写。然后我将在所有的数据的基础上使用基于word2vec的doc2vec方法对整个数据集进行建模。然后用得到的能够有效表达文档特征的100维向量来代替文档，最后再把特征向量数据和文档标签一起放到一个多维分类算法中，用监督学习的方法构建一个神经网络模型，并最终实现对文档的分类。选用doc2vec的方法是因为大多数的文档处理办法主要有两种，Tf-Idf [5]以及word2vec。Tf-Idf的核心是词频，但是它忽略了词与词之间的顺序关系，而word2vecword2vec 模型利用词的上下文信息将一个词转化成一个低维实数向量，越相似的词在向量空间中越相近。将词向量应用于自然语言处理非常成功，已经被广泛应用于中文分词、情感分类、句法依存分析等。但是它只是基于一个词周围的词，对于一个比较长的文档它也忽略了上下文不同段落间的关系。我们选用的算法为了更准确的分类希望可以更有效的保留原始文本的信息，将整个文档进行向量化。另一方面因为在本算法中文档的向量化对结果的权重非常。如果文档向量化后有比较高的辨识度那么基本上已经完成自动分类了；反之则无论选用什么样的分类算法都因为特征向量无法准确的表示文档而无法得出准确度高的模型。所以对于最后的分类算法我们考虑选择近些年比较热门且表现非常好的全连接神经网络分类算法而这也是doc2vec论文作者推荐的算法和朴素贝叶斯 [12]多维分类算法。神经网络是对人脑或自然神经网络（Natural Neural Network）若干基本特性的抽象和模拟。它的优点是分类的准确度高,并行分布处理能力强,有较强的鲁棒性和容错能力，非常适合复杂问题超大维度的特征训练，它的缺点在于参数繁多、调参困难，对于并不复杂的问题训练速度比较慢而且输出结果无法解释原因有黑盒。全连接神经网络主要包含输入层input、隐藏层hidden以及输出层output。在本项目中我们的输入层将会是doc2vec输出的文档向量，而输出层则对应文档将要被分类的四个类别，并且使用softmax方法作为输出层的激活函数来对应多维分类。朴素贝叶斯是一系列以假设特征之间强（朴素）独立下运用贝叶斯定理为基础的简单概率分类器。它的优点是对训练集数量要求很少，高扩展性而且常被用来做文档恢复的工作，缺点在于模型可能过于简单而且必须基于假设特征之间相互独立。

如何利用词向量有效地表示一篇文档是NLP领域的一个难点。目前在这方面的研究进展缓慢，常见的方法有对一篇文档所包含的所有词向量求平均值、对词向量聚类以及doc2vec。我们之所以在这三者中选择doc2vec是考虑到数据集文档普遍有一定长度（>300），单纯的求均值或聚类都会让辨识度降低，而且根据`Distributed Representations of Sentences and Documents` [6]的实验结果doc2vec的准确度更好超过传统的贝叶斯、RNN等。doc2vec主要有两种训练模型的方法，一种是利用前几个词和文档特征向量一起来预测后面一个词，非常类似于word2vec中的CBOW模型。



从上图中可以看到在这一类训练过程中会依据对于前几个词的词向量聚合的方式不同分为两种方法：PV-DMM就是对每一个词的词向量区平均值作为最终向量；PV-DMC就是对于每一个词的词向量进行联合组成一个包含所有向量的更高维度的向量来作为最后的向量。

另一种是对于一个随机的窗口段落，随机给出其中的一个词再加上文档特征向量去预测整个段落的其他单词，它非常类似与word2vec中的skip-gram。实际对应的方法就是PV-DBOW（Distributed Bag of Words version of Paragraph Vector）。



我们会分别使用这三种方法去训练模型，以求达到最好的效果。

我们使用sk-learn库中的neural_network来构建神经网络分类模型，使用gensim库中的doc2vec来实现文档向量化。

基准模型

许多研究者已经对*20newsgroup*进行过分类处理了。来自斯坦福大学的一个小组使用了斯坦福分类方法得到了准确率85%的成绩在所有20个分类上。他们使用的数据是原始的*20newsgroup*的数据集，通过自己的一个脚本进行的基本的数据预处理主要是把换行符替换成空格并将文档转化为utf8的格式，在分类算法上他们使用的是Stanford Classifer[11]，这是一种利用特征的正负标签进行含权值投票的分类算法。其他的一些研究报告里[7]，准确率也都是低于90%的。

我们利用所有数据去学习构建向量，而只用其中的四类去做最后的分类测试，而且这四类分属不同的大类理论上辨识度更高。所以我们认为我们最终的准确度会好于他们，达到90%+。

III. 方法

数据预处理

我们起先在数据集的导入上遇到了麻烦。最开始使用的是python3.5来运行数据文本读取的工作，但是发现在许多文档上报错，提示有不在utf-8的字符集出现，无法识别。在排除了windows操作系统和中文系统环境的原因后，最终发现如果在python2.7下是可以正常读取文档的。应该是python3升级之后对于字符的有更严格的要求有关。同时在文本导入之前，我们使用了在 text8 数据提供者提供的清洗 Wikipedia dump 的脚本 [7]基础上修改过的脚本，同样对 *20 Newsgroups* 数据集进行处理：清除其他所有字符，只保留由小写字母 a-z 组成的单词、单一空格（将不在 a-z 之间的字符也一律转换为空格）。这样得到的数据集更加纯净，有利于词向量的更准确的训练。后来发现sk-learn其实已经有 *20 Newsgroups*的数据集，以后要用的时候可以直接导入。

执行过程

我们使用gensim[8]库中的doc2vec方法来构建文档向量模型，然后对四个备选类别中的训练集（2383篇）和测试集（1585篇）中的所有文档用训练好的模型进行向量化，最后使用sk-learn库中的MLPClassifier方法对经过StandardScaler标准化后的向量和标签进行训练，得到最终的分类模型。我们使用的验证标准是分类准确度和F1值。

在使用doc2vec方法的时候，我们首先考虑为了达到更高的准确度准备同时用它的两种训练方式得到2个100维度的文档向量，再将他们融合起来得到最终的特征向量。对于这两种模型我们均使用默认参数。但是当我们把训练后的文档向量用神经网络算法去学习之后，测试集上的准确度居然是**25%**。这基本上就是随机猜测的概率，我们完全无法相信似乎doc2vec什么有用的信息也没有学到。随后我们又从gensim的github上找到了一个新的案例，我们仿照它的代码重构了自己的可能错误的doc2vec的训练代码，并且使用了三种不同的训练方法：PV-DMC ，PV-DMM，PV-DBOW（解释见上文）

```
simple_models = [
    # PV-DM w/concatenation - window=5 (both sides) approximates paper's 10-word total window
size
    Doc2Vec(dm=1, dm_concat=1, size=100, window=5, negative=5, hs=0, min_count=2,
workers=cores),
    # PV-DBOW
    Doc2Vec(dm=0, size=100, negative=5, hs=0, min_count=2, workers=cores),
    # PV-DM w/average
    Doc2Vec(dm=1, dm_mean=1, size=100, window=10, negative=5, hs=0, min_count=2, workers=cores),

]
```

这里的dm参数1表示使用dm方法，0表示使用DBOW方法；size表示最终输出的文档向量的维度；min_count表示文档中出现最少多少词的词才会被真实加入算法运算，这里设置为2也就是增加到多频词的权重；window表示训练模型的词窗口的大小；negative表示负类词的选择个数。

并对这三种训练方法得到的文档向量再次放入神经网络训练分类，没想到三种方法的正确率居然还都是**25%**。我们这时候对doc2vec产生了很大的怀疑，至少是在这个数据集上的怀疑，是否有什么原因让它不适合。我们又考虑是否是因为停止词没有去除导致垃圾信息过多造成文档向量没有表明文档特征呢。然后我们对输入文档进行过滤，使用gensim自带的STOPWORDS。然而最终结果还是没有变化。此时我们几乎要放弃对doc2vec的使用了，我们觉得它根本无法获取有用的文档信息所以才导致后面的神经网络算法只有随机猜测的正确率。

就在我们调试最后一次的时候偶然发现了一个奇怪的现象就是似乎不管模型参数如何变，一篇文档最终计算出来的向量始终是不变的。一开始觉得只是偶然，后来刻意多测试了才发现真的是完全没有变，这也解释了为什么准确度只有随机猜测的25%。当我们意识到某个地方一定错了之后又去反复查阅了doc2vec的文档，最终发现模型训练过程的并没有问题，问题出现在对文档使用模型进行向量转化的代码上。

```
#this is wrong
model.infer_vector(corpus[z].words)

#this is right
model.infer_vector(gensim.utils.simple_preprocess(corpus[z].words))
```

错误的原因就是官方文档中明确说了

> `infer_vector`(*doc_words*, *alpha=0.1*, *min_alpha=0.0001*, *steps=5*)
>
> Infer a vector for given post-bulk training document.
>
> Document should be **a list of (word) tokens**.

而我们之前所做的都是输入原文本，正确的代码应该是输入一个文本数组。不知是什么原因输入的格式都不一样，gensim也没有报错并且正常的输出结果，这里应该是它的一个bug。或者是因为版本更新造成的差异，因为官方推荐的例子中也是输入原文本。

在修正了代码之后后续的分类模型可以正常被训练了，并且在输入分类算法之前我们对于文档向量进行了归一化的处理将每个值缩放到[0,1]的范围里，以提高最后的准确度。因为我们设定的文档向量是100维，而最终的类别有4类，所以我们为神经网络模型选择了一个50节点的隐藏层，并且使用Relu的激活函数和Adam的优化方法。最终使用PV-DMC 和PV-DMM得到的正确率是**75%**，而PV-DBOW的到的正确率竟然达到了**97.41%**！在使用朴素贝叶斯作为分类算法的时候PV-DMC 和PV-DMM得到的正确率是**72%**，而PV-DBOW的正确率为97.10%**，但是训练时间明显比神经网络算法要快。

完善

首先我们验证了一些想法。之前训练的时候我们对文档去除了停止词，现在再加上看是否会有准确度的下降或者上升。测试结果仍然是97%左右,似乎并没有什么影响。

另外我们调整了朴素贝叶斯分类器的alpha值从150到0.01，发现当alpha值为110时准确度最高为**97.35%**，再高或者再低都会降低一般在96.8%~91.1%之间。

对,于神经网络分类算法我们调整了hidden_layer_size,alpha和learning_rate_init的值。我们发现只要降低learning_rate_init到足够小总是可以对训练结果有一定的提升但是训练时间增长过快，最后再权衡下我们找到参数hidden_layer_size=90，alpha=0.001，learning_rate_init=0.0001的时候算法结果很好达到**97.67%**且训练时间也可以接受。

IV. 结果

模型的评价与验证

我们最终的模型是使用PV-DBOW训练的doc2vec模型加神经网络多元分类模型(hidden_layer_size=90，alpha=0.001，learning_rate_init=0.0001)组合起来的文档分类模型。我们在与两种其他方法PV-DMC 和PV-DMM以及一种微调方法即不去除停止词构建文档模型比较下得出该模型是最佳模型。我们还验证了该模型测试集上的confusion_matrix以及F1值。

```
                                  [[377   4   5   8]
                                                    [  0 396   1   2]
                                                    [  4   2 384   4]
                                                    [  3   1   3 391]]


              precision    recall  f1-score   support

          1       0.98      0.96      0.97       394
          2       0.98      0.99      0.99       399
          3       0.98      0.97      0.98       394
          4       0.97      0.98      0.97       398

avg / total       0.98      0.98      0.98      1585
```

上表可以看出最佳模型对于需要验证的四个类别，无论是召回率还是查全率以及正确的个数上都是非常优秀的。其他模型的结果见下表：

	准确率	**F1**	训练时间
PV-DBOW	0.98	0.98	6min
PV-DMC	0.75	0.75	12min
PV-DMM	0.75	0.75	7min

根据Distributed Representations of Sentences and Documents的描述，作者在IMDB的50000条评论中使用doc2vec方法最高达到了92%的准确率，我们这里只有四类，而且还是类别差距比较大的四类，新闻的大类都分别不一样（soc，comp，rec，sic）所以达到98%的准确度也是完全有可能的。从敏感性上再次考虑，我们换四组类别更近一些的验证数据，是否模型的准确率会下降呢？

我们选取了'soc.religion.christian','talk.politics.guns','talk.politics.mideast','talk.politics.misc'这四类，其中三个的二级子类都是相同的。用之前的最佳模型测试这四组结果，得到的精确度和F1值仍然是98%，这说明我们的最佳模型有很强的鲁棒性。

所以我认为我们得到的最佳模型是非常合理的，并且远远超越了预想。**doc2vec**是非常成功文档向量化的方法。

合理性分析

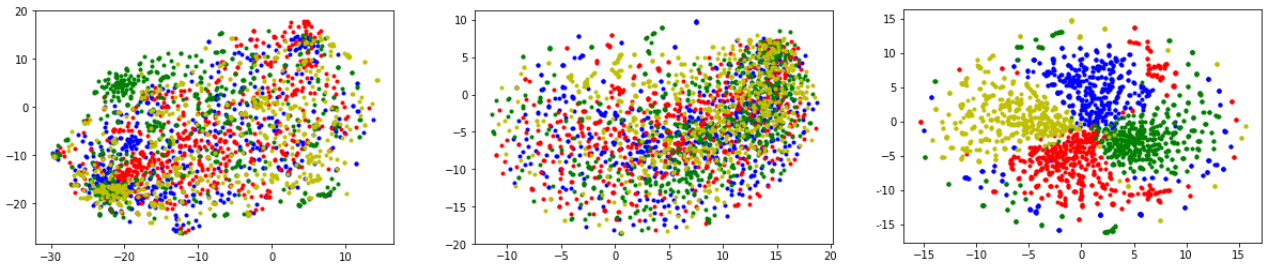
我们选择的基准模型是斯坦福大学小组的，他们在20类上的准确度是85%，我们的最佳模型在4类上的准确度为98%。因为上一段的验证我们的鲁棒性非常不错，所以我们有理由相信最佳模型的表现应该比基准模型好或者差不多水平。我们的更好结果应该得益于doc2vec向量化对整体文档信息的记忆和保全。这里的最佳模型确确实实解决了本文开头提出的文档分类问题。

V. 项目结论

结果可视化

我们认为最佳模型的出色表现绝大多数原因在于doc2vec向量化的成功，正因为它找到了不同类别文档的特征差异才使得后续的分类算法可以高度精确的对从未看过文本的测试集进行分类。那么如果doc2vec得到的向量是可分别的，那么就很像一个聚类算法。我们想看看不经过分类算法仅通过查看文档向量的可视化能否明确的发现不同类别呢。

我们使用sk-learn的TSNE[10]包对四类文本的训练集的100维向量进行二维可视化展示结果如图：



上图可以很容易的看出表现最好的PV-DBOW模型在可视化之后也是非常美，四个类别几乎完美分开，而且界限明显。相比之下另外两种模型的可视化似乎并不能很好的区分清楚，所以准确率相对较低。虽然另两个模型在二维图上没有分开却并不妨碍分类算法处理后的准确率可以达到75%+，这说明这些文档向量化之后的的确确有获得文档类别本身的特征，只是这种特征在二维图上还不能很好的展示出来，但是当我们用其他分类算法再训练一次的化就可以很轻松的达到不错的结果了。所以我们建议以后处理文档分类也可以使用类似的方式，以达到最好的效果。

对项目的思考

本项目是解决一个大量文档的多元分类问题。我们使用了*20newsgroup*数据集，用所有数据训练出了文档向量模型，并且运用在了四个分类上，通过神经网络分类算法实现最终的分类。我们使用了doc2vec技术来获得文档的向量表示，结果非常好。通过验证我们得到PV-DBOW模型在这个数据集上可以获得最好的效果。

最后我们验证了该模型的合理性和鲁棒性，在更换输入类别后模型依然有优异的表现。从而证明了doc2vec的有效性，并且确立了doc2vec+neural_network方法的优势。这个方法可以推广到大多数的文本分类问题中去，我相信它都会给出非常不错的结果。

本项目在实施过程中因为模型库版本的问题耽误了许多时间，也几乎要放弃doc2vec。但是最终还是坚持了下来才看到如此美的可视化结果。所以建议大家如果做机器学习类问题遇到了非常离谱的结果，那么最应该去做的事情就是查阅库的官方文档。或者保证版本和原作者一致，或者对照最新的文档验证每个参数是否设置正确。

需要作出的改进

因为doc2vec的两种训练方式在本项目中结果差异比较大，我们对落后的那一种方式也尝试进行了一些改变，比如选择不去除停止词以及增加训练次数。因为是提取文档信息而非词袋子模型所以选择不删除停止词确实提高了准确率达到82%，最后再加上增加训练次数准确率提高到88%，或许还可以再进一步提高，但是考虑到时间成本再加上已经有非常好的模型了所以意义不大。

未来的改进方法我们可以把两个模型的向量结合起来，以整合后的向量作为文档的向量我想这样应该可以获得更多的文档信息，也一定会提高分类的准确度。而且我们还需要将分类拓展到所有20类文档中去，验证最佳模型是否还能有很好的表现。深度学习中的RNN对于文档处理也很适合，我们也可以考虑尝试用RNN网络来训练对文档的预测，用隐藏层的最终结果来表示该文档的向量，最后再结合其他分类算法实现对文档的分类。

参考文献

[1] Doc2vec tutorial | RaRe Technologies rare-technologies.com/doc2vec-tutorial/

[2] Word2vec - Wikipedia en.wikipedia.org/wiki/Word2vec

[3] 基于 Word2Vec 的一种文档向量表示 《计算机科学》 2016.6

[4] 20newsgroup www.qwone.com/~jason/20Newsgroups

[5] Tf-Idf en.wikipedia.org/wiki/Tf%E2%80%93idf

[6] Distributed Representations of Sentences and Documents arxiv.org/pdf/1405.4053v2.pdf

[7] NLP Stanford Classifier: nlp.stanford.edu/wiki/Software/Classifier/20_Newsgroups

[8] gensim document radimrehurek.com/gensim/models/doc2vec.html

[9] 情感分析新方法 datartisan.com/article/detail/48.html

[10] doc2vec & IMDB github.com/RaRe-Technologies/gensim/blob/develop/docs/notebooks/doc2vec-IMDB.ipynb

[11] Stanford Classifier nlp.stanford.edu/software/classifier.html

[12] Naive Bayes Classifier: en.wikipedia.org/wiki/Naive_Bayes_classifier

---

额外的话：非常非常感谢udactiy的这个毕业项目，让我真正有了一次从头到尾写一篇学术论文的体验。选择doc2vec本身对我也是个挑战，因为原本没有接触过这个工具，也不知道会遇到什么困难。从最初的不知从何下手，到模型无法分类的绝望直到最后如此美丽的可视化分类图出现的那一刻，回想这一路上的成长真是有许多许多说不出的感受。不完整的体验一次这个过程你永远也不知道什么叫做成功，这算是我的第一篇学术论文，在udacity的帮助下我给自己打个满分：D