北京航空航天大學
BEIHANG UNIVERSITY

深度学习与自然语言处理（NLP）第二次课后作业

院（系）名称	自动化科学与电气工程学院
专业名称	自动化
学号	ZY2303808
学生姓名	牛晨然

2024 年 05 月

1 内容介绍

从给定的语料库中均匀抽取 1000 个段落作为数据集（每个段落可以有 K 个 token,K 可以取 20，100，500,1000,3000），每个段落的标签就是对应段落所属的小说。利用 LDA 模型在给定的语料库上进行文本建模，主题数量为 T，并把每个段落表示为主题分布后进行分类（分类器自由选择），分类结果使用 10 次交叉验证（i.e.900 做训练，剩余 100 做测试循环十次）。实现和讨论如下的方面：（1）在设定不同的主题个数 T 的情况下，分类性能是否有变化？；（2）以"词"和以"字"为基本单元下分类结果有什么差异？（3）不同的取值的 K 的短文本和长文本，主题模型性能上是否有差异？

2 实验原理

LDA（Latent Dirichlet Allocation）是一种文档主题生成模型，也称为一个三层贝叶斯概率模型，包含词、主题和文档三层结构。所谓生成模型，就是说，我们认为一篇文章的每个词都是通过“以一定概率选择了某个主题，并从这个主题中以一定概率选择某个词语”这样一个过程得到。文档到主题服从多项式分布，主题到词服从多项式分布。

LDA 是一种非监督机器学习技术，可以用来识别大规模文档集或语料库中潜藏的主题信息。它采用了词袋的方法，这种方法将每一篇文档视为一个词频向量，从而将文本信息转化为了易于建模的数字信息。但是词袋方法没有考虑词与词之间的顺序，这简化了问题的复杂性，同时也为模型的改进提供了契机。每一篇文档代表了一些主题所构成的一个概率分布，而每一个主题又代表了很多单词所构成的一个概率分布。

对于语料库中的每篇文档，LDA 定义了如下生成过程：

1.对每一篇文档，从主题分布中抽取一个主题；

2.从上述被抽到的主题所对应的单词分布中抽取一个单词；

3.重复上述过程直至遍历文档中的每一个单词。

语料库中的每一篇文档与 T（通过反复试验等方法事先给定）个主题的一个多项分布相对应，将该多项分布记为 θ 。每个主题又与词汇表中的 V 个单词的

一个多项分布相对应，将这个多项分布记为 φ。

先定义一些字母的含义：文档集合 D，主题（topic)集合 T

D 中每个文档 d 看作一个单词序列<$w1,w2,\cdots,wn$>，wi表示第i个单词，设 d 有 n 个单词。D 中涉及的所有不同单词组成一个大集合，LDA 以文档集合 D 作为输入，希望训练出的两个结果向量：对每个 D 中的文档 d，对应到不同 Topic 的概率θd<$pt1,\cdots,ptk$>，其中，pti表示 d 对应 T 中第 i 个 topic 的概率。计算方法是直观的，$pti=nti/n$，其中nti表示 d 中对应第 i 个 topic 的词的数目，n 是 d 中所有词的总数。对每个 T 中的 topict,生成不同单词的概率φt<$pw1,\cdots,pwm$>，其中，pwi表示 t 生成 VOC 中第 i 个单词的概率。计算方法同样很直观，$pwi=Nwi/N$，其中Nwi表示对应到 topict 的 VOC 中第 i 个单词的数

目，N 表示所有对应到 topict 的单词总数。LDA 的核心公式如下：$p(w|d)=p(w|t)*p(t|d)$

直观的看这个公式，就是以 Topic 作为中间层，可以通过当前的θd和φt给出了文档 d 中出现单词 w 的概率。其中$p(t|d)$利用θd计算得到，$p(w|t)$利用φt计算得到。

实际上，利用当前的θd和φt，我们可以为一个文档中的一个单词计算它对应任意一个 Topic 时的$p(w|d)$，然后根据这些结果来更新这个词应该对应的 topic。然后，如果这个更新改变了这个单词所对应的 Topic，就会反过来影响θd和φt。

LDA 算法开始时，先随机地给θd和φt赋值（对所有的 d 和 t）。然后上述过程不断重复，最终收敛到的结果就是 LDA 的输出。再详细说一下这个迭代的学习过程：

1.针对一个特定的文档 ds 中的第 i 单词 wi，如果令该单词对应的 topic 为 tj，可以把上述公式改写为：

$pj(wi|ds)=p(wi|tj)*p(tj|ds)$

2.现在我们可以枚举 T 中的 topic,得到所有的$pj(wi|ds)$,其中 j 取值 1~k。然后可以根据这些概率值结果为 ds 中的第 i 个单词 wi 选择一个 topic。最简单的想法是取令$pj(wi|ds)$最大的 tj（注意，这个式子里只有 j 是变量）。

3.然后,如果 ds 中的第 i 个单词 wi 在这里选择了一个与原先不同的 topic，就会对θd和φt有影响了（根据前面提到过的这两个向量的计算公式可以很容易

知道)。它们的影响又会反过来影响对上面提到的$p(w|d)$的计算。对 D 中所有的 d 中的所有 w 进行一次$p(w|d)$的计算并重新选择 topic 看作一次迭代。这样进行 n 次循环迭代之后，就会收敛到 LDA 所需要的结果了。

随机森林是一种集成学习方法，用于解决分类和回归问题。它通过构建多个决策树，并将它们的输出结合起来进行预测，从而提高了模型的性能和鲁棒性。随机森林的基本组成部分是决策树。决策树是一种基于树结构的分类模型，通过对数据进行递归分割，将数据划分为不同的类别。通过构建多个决策树，并将他们的预测结果进行组合，提高整体模型的性能。

3 实验结果与分析

使用随机森林分类器，设置每段的 token=20，100，500，1000；topic=5，20，100，500 分别按照字分词和 jieba 分词，共进行了 32 次实验，每次实验使用 10 次交叉验证，准确率如下表所示，表格横向为 topic，纵向为 token：

表 1 按词分类结果

token \ topic	5	20	100	500
20	0.11	0.09	0.1	0.11
100	0.1	0.1	0.12	0.12
500	0.21	0.3	0.38	0.4
1000	0.35	0.44	0.69	0.72

表 2 按字分类结果

token \ topic	5	20	100	500
20	0.11	0.14	0.1	0.14
100	0.17	0.22	0.36	0.24
500	0.32	0.63	0.8	0.76
1000	0.47	0.84	0.9	0.86

通过分析实验结果，可以得到以下结论：

①相同的 token 和 topic 情况下，按字分类比按词分类的准确率要高。

②当 token 数量增加时，分类的准确率也会增长，增加 token 的数量有利于提高分类准确率。

③当 token 数量较少时，改变 topic 的数量基本不会对准确率产生影响；token 数量较大时，当 topic 数量增加，准确率会先增加再减小，选择适中的 topic 数量有利于提高分类准确率。

4 参考文献

[1] https://blog.csdn.net/weixin_50891266/article/details/116273153

[2] https://blog.csdn.net/sai_simon/article/details/123082619