NLP-3

罗志钊 PT2100565

问题描述

从给定的语料库中均匀抽取200个段落（每个段落大于500个词）， 每个段落的标签就是对应段落所属的小说。利用LDA模型对于文本建模，并把每个段落表示为主题分布后进行分类。验证与分析分类结果。

预备知识

Topic model

主题模型是以非监督学习的方式对文档的隐含语义结构进行聚类的统计模型

$$p(w_i|d_j) = \sum_{k=1}^{K} p(w_i|t_k) \times p(t_k|d_j)$$

其中

$$w_i\text{代表单词；}d_j\text{代表文档；}t_k\text{代表主题}$$

LDA算法

引入

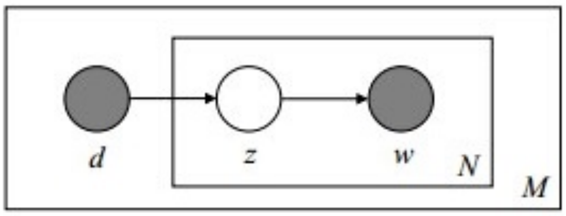
在一个已知的数据集中，每个词和文档对应的p(w_i|d_j) 都是已知的。而主题模型就是根据这个已知的信息，通过计算p (w_i|t_k) 和p(t_k|d_j) 的值，从而得到主题的词分布和文档的主题分布信息。而要得到这个分布信息，现在常用的方法就是LSA(LSI)和LDA。其中LSA主要是采用SVD的方法进行暴力破解，而LDA则是通过贝叶斯学派的方法对分布信息进行拟合。这里我们使用LDA算法。

算法简介

利用LDA模型生成一篇文档的方式：

- 按照先验概率 P(d_i) 选择一篇文档d_i
- 从狄利克雷分布（即Dirichlet分布）α 中取样生成文档 d_i 的主题分布 θ*i，换言之，主题分布 θ*i 由超参数为 α 的Dirichlet分布生成。
- 从主题的多项式分布 θ_i 中取样生成文档 d_i 第 j 个词的主题 z_i , j。
- 从主题的多项式分布 θ_i 中取样生成文档d_i第j个词的主题z_i,j。
- 从狄利克雷分布（即Dirichlet分布）β 中取样生成主题z_i , j对应的词语分布 ϕ*zi , j ，换言之，词语分布 ϕ*zi , j 由参数为 β 的Dirichlet分布生成。
- 从词语的多项式分布 ϕ_zi , j 中采样最终生成词语 w_i , j。

根据文档反推其主题分布：



图中被涂色的d、w表示可观测变量，未被涂色的z表示未知的隐变量;
从而可以根据大量已知的文档-词项信息，训练出文档-主题和主题-词项，如下公式所示：

$$P(w_j|d_j)=\sum_{k=1}^{K}P(w_j|z_k)P(z_k|d_i)$$

故得到文档中每个词的生成概率为：

$$P(d_i|w_j)=P(d_i)P(w_j|d_i)=P(d_i)\sum_{k=1}^{K}P(w_j|z_k)P(z_k|d_i)$$

由于可事先计算求出，而P(w_j|z_k)和P(z_k|d_i)未知，所以θ=(P(w_j|z_k),P(z_k|d_i))就是我们要估计的参数，通俗点说，就是要最大化这个θ。

问题分析

由于语料库存在大量标点符号和广告类重复文本，因而和作业一类似，使用jieba进行分词得到处理后片段。

将金庸16本小说均匀分成200个段落（每个段落大于500词），并使用gensim中的corpora和LDA模型训练，构建词频矩阵，提取其主题分布特征。这是本次问题的核心。

选出其中10个段落并抽取500词进行测试，并与这10个段落的真实来源（标签）进行比对，检验分析结果。

实验过程

预处理

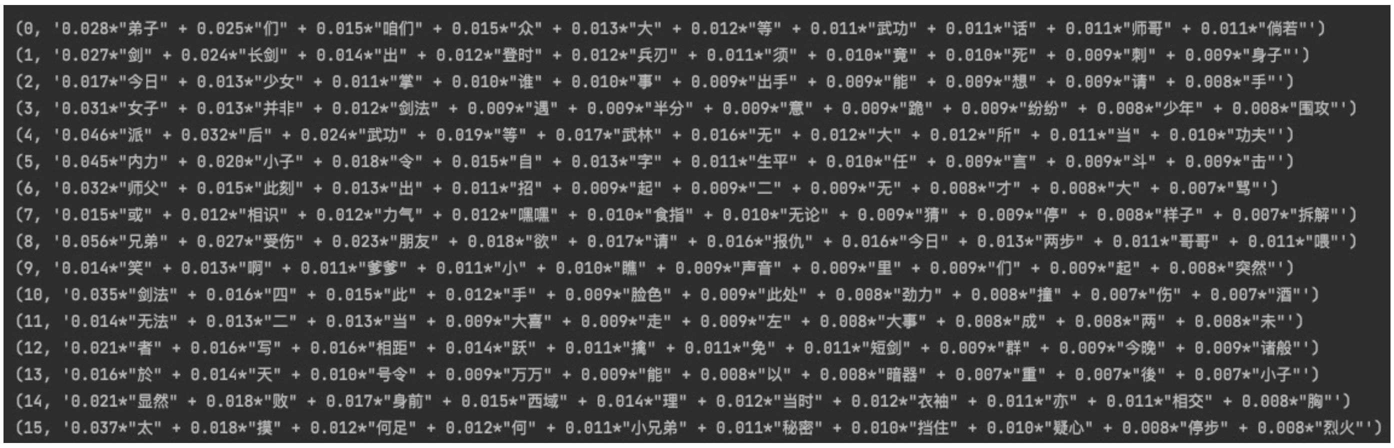
将金庸先生的16本小说中所有的非中文词汇和标点符号去除，具体代码和作业一类似；同时由于语言中存在大量stop words，会严重每本小说得出的主题，部分暂停词如下。

```
sw = ["的", "了", "在", "是", "我", "有", "和", "就",
      "不", "人", "都", "一", "一个", "上", "也", "很", "到", "说", "要", "去", "你",
      "会", "着", "没有", "看", "好", "自己", "这","罢"
,"这",'在','又','在','得','那','他','她','不','而','道','与','之','见','却','问','可','但'
                             ,'没','啦','给','来','既','叫','只','中','么','便'
                             ,'听','为','跟','个','甚','下','还','过','向','如此'
                             ,'已','位','对','如何','将','岂','哪','似','以免','均'
                             ,'虽然','即','由','再','使','从','麽','其实','阿','被']
```

训练lDA模型

把数据均匀分割成200个段落，利用gensim中自带的lDA模型来训练，corpora来构建词频矩阵。

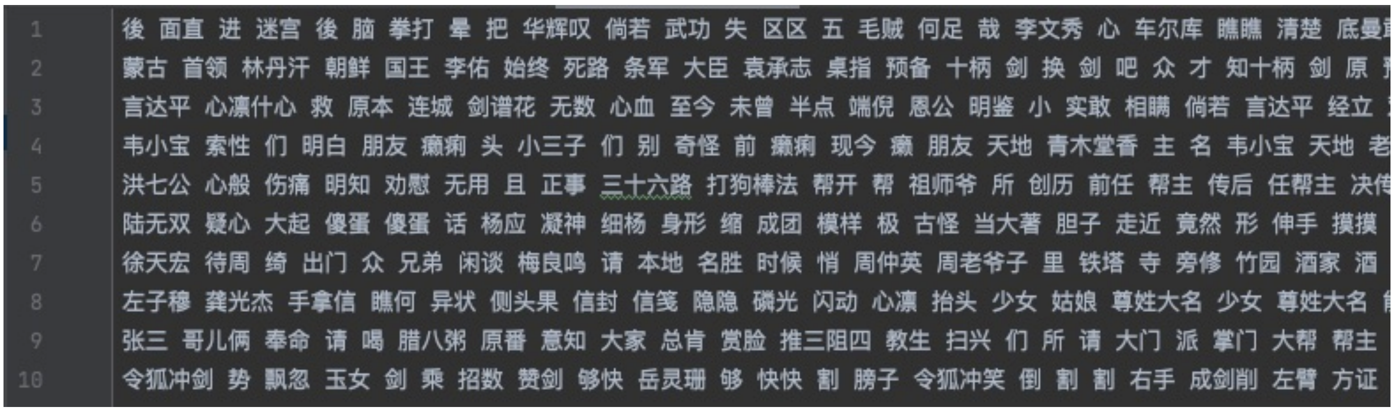
```
    dictionary = corpora.Dictionary(train)
    # corpus是把每本小说ID化后的结果，每个元素是新闻中的每个词语，在字典中的ID和频率
    corpus = [dictionary.doc2bow(text) for text in train]
    lda = models.LdaModel(corpus=corpus, id2word=dictionary, num_topics=16)
    topic_list_lda = lda.print_topics(16)
    print("16本小说主题分布为：\n")
    for topic in topic_list_lda:
        print(topic)
```



```
(0, '0.028*"弟子" + 0.025*"们" + 0.015*"咱们" + 0.015*"众" + 0.013*"大" + 0.012*"等" + 0.011*"武功" + 0.011*"话" + 0.011*"师哥" + 0.011*"倘若"')
(1, '0.027*"剑" + 0.024*"长剑" + 0.014*"出" + 0.012*"登时" + 0.012*"兵刃" + 0.011*"须" + 0.010*"竟" + 0.010*"死" + 0.009*"刺" + 0.009*"身子"')
(2, '0.017*"今日" + 0.013*"少女" + 0.011*"掌" + 0.010*"谁" + 0.010*"事" + 0.009*"出手" + 0.009*"能" + 0.009*"想" + 0.009*"请" + 0.008*"手"')
(3, '0.031*"女子" + 0.013*"并非" + 0.012*"剑法" + 0.009*"遇" + 0.009*"半分" + 0.009*"意" + 0.009*"跪" + 0.009*"纷纷" + 0.008*"少年" + 0.008*"围攻"')
(4, '0.046*"派" + 0.032*"后" + 0.024*"武功" + 0.019*"等" + 0.017*"武林" + 0.016*"无" + 0.012*"大" + 0.012*"所" + 0.011*"当" + 0.010*"功夫"')
(5, '0.045*"内力" + 0.020*"小子" + 0.018*"令" + 0.015*"自" + 0.013*"字" + 0.011*"生平" + 0.010*"任" + 0.009*"言" + 0.009*"斗" + 0.009*"击"')
(6, '0.032*"师父" + 0.015*"此刻" + 0.013*"出" + 0.011*"招" + 0.009*"起" + 0.009*"二" + 0.009*"无" + 0.008*"才" + 0.008*"大" + 0.007*"骂"')
(7, '0.015*"或" + 0.012*"相识" + 0.012*"力气" + 0.012*"嘿嘿" + 0.010*"食指" + 0.010*"无论" + 0.009*"猜" + 0.009*"停" + 0.008*"样子" + 0.007*"拆解"')
(8, '0.056*"兄弟" + 0.027*"受伤" + 0.023*"朋友" + 0.018*"欲" + 0.017*"请" + 0.016*"报仇" + 0.016*"今日" + 0.013*"两步" + 0.011*"哥哥" + 0.011*"喂"')
(9, '0.014*"笑" + 0.013*"啊" + 0.011*"爹爹" + 0.011*"小" + 0.010*"瞧" + 0.009*"声音" + 0.009*"里" + 0.009*"们" + 0.009*"起" + 0.008*"突然"')
(10, '0.035*"剑法" + 0.016*"四" + 0.015*"此" + 0.012*"手" + 0.009*"脸色" + 0.009*"此处" + 0.008*"劲力" + 0.008*"撞" + 0.007*"伤" + 0.007*"酒"')
(11, '0.014*"无法" + 0.013*"二" + 0.013*"当" + 0.009*"大喜" + 0.009*"走" + 0.009*"左" + 0.008*"大事" + 0.008*"成" + 0.008*"两" + 0.008*"未"')
(12, '0.021*"者" + 0.016*"写" + 0.016*"相距" + 0.014*"跃" + 0.011*"擒" + 0.011*"免" + 0.011*"短剑" + 0.009*"群" + 0.009*"今晚" + 0.009*"诸般"')
(13, '0.016*"於" + 0.014*"天" + 0.010*"号令" + 0.009*"万万" + 0.009*"能" + 0.008*"以" + 0.008*"暗器" + 0.007*"重" + 0.007*"後" + 0.007*"小子"')
(14, '0.021*"显然" + 0.018*"败" + 0.017*"身前" + 0.015*"西域" + 0.014*"理" + 0.012*"当时" + 0.012*"衣袖" + 0.011*"亦" + 0.011*"相交" + 0.008*"胸"')
(15, '0.037*"太" + 0.018*"摸" + 0.012*"何足" + 0.012*"何" + 0.011*"小兄弟" + 0.011*"秘密" + 0.010*"挡住" + 0.010*"疑心" + 0.008*"停步" + 0.008*"烈火"')
```

测试

选取其中10个段落的前500个词存入`dic_test_data.txt`中，部分内容如下，用于训练集中测试。



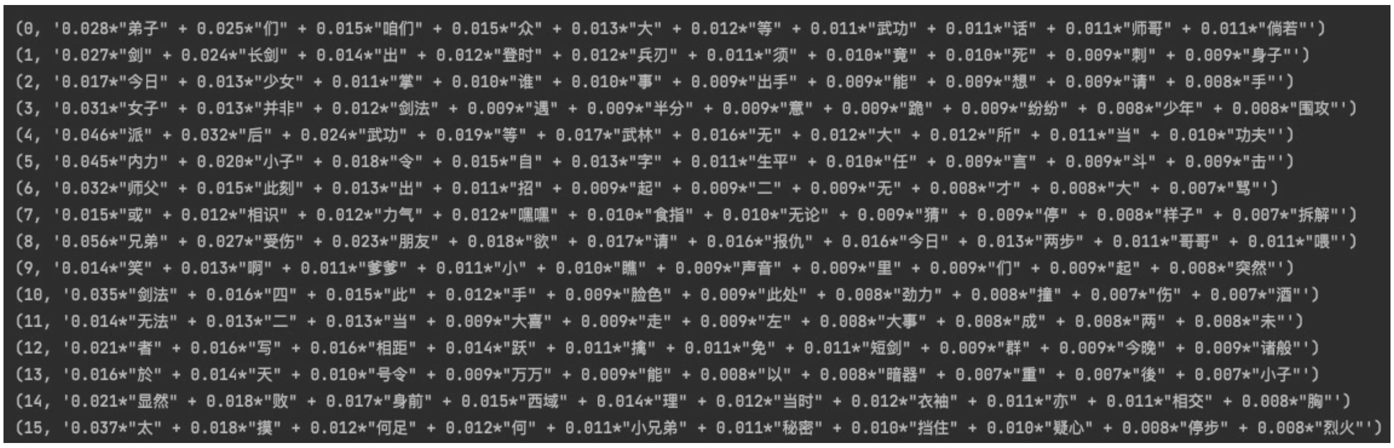
```
後 面直 进 迷宫 後 脑 拳打 晕 把 华辉叹 倘若 武功 失 区区 五 毛贼 何足 哉 李文秀 心 车尔库 瞧瞧 清楚 底曼
蒙古 首领 林丹汗 朝鲜 国王 李佑 始终 死路 条军 大臣 袁承志 桌指 预备 十柄 剑 换 剑 吧 众 才 知十柄 剑 原
言达平 心凛什心 救 原本 连城 剑谱花 无数 心血 至今 未曾 半点 端倪 恩公 明鉴 小 实敢 相瞒 倘若 言达平 经立
韦小宝 索性 们 明白 朋友 癞痢 头 小三子 们 别 奇怪 前 癞痢 现今 癞 朋友 天地 青木堂香 主 名 韦小宝 天地 老
洪七公 心般 伤痛 明知 劝慰 无用 且 正事 三十六路 打狗棒法 帮开 帮 祖师爷 所 创历 前任 帮主 传后 任帮主 决传
陆无双 疑心 大起 傻蛋 傻蛋 话 杨应 凝神 细杨 身形 缩 成团 模样 极 古怪 当大著 胆子 走近 竟然 形 伸手 摸摸
徐天宏 待周 绮 出门 众 兄弟 闲谈 梅良鸣 请 本地 名胜 时候 悄 周仲英 周老爷子 里 铁塔 寺 旁修 竹园 酒家 酒
左子穆 龚光杰 手拿信 瞧何 异状 侧头果 信封 信笺 隐隐 磷光 闪动 心凛 抬头 少女 姑娘 尊姓大名 少女 尊姓大名
张三 哥儿俩 奉命 请 喝 腊八粥 原番 意知 大家 总肯 赏脸 推三阻四 教生 扫兴 们 所 请 大门 派 掌门 大帮 帮主
令狐冲剑 势 飘忽 玉女 剑 乘 招数 赞剑 够快 岳灵珊 够 快快 割 膀子 令狐冲笑 倒 割 割 右手 成剑削 左臂 方证
```

把测试数据带入模型当中，验证分析结果。

实验结果

1. 主题段落并分类(部分)，得出16本金庸小说主题分布的分类。

   ```
   (0, '0.028*"弟子" + 0.025*"们" + 0.015*"咱们" + 0.015*"众" + 0.013*"大" + 0.012*"等" + 0.011*"武功" + 0.011*"话" + 0.011*"师哥" + 0.011*"倘若"')
   (1, '0.027*"剑" + 0.024*"长剑" + 0.014*"出" + 0.012*"登时" + 0.012*"兵刃" + 0.011*"须" + 0.010*"竟" + 0.010*"死" + 0.009*"刺" + 0.009*"身子"')
   ```

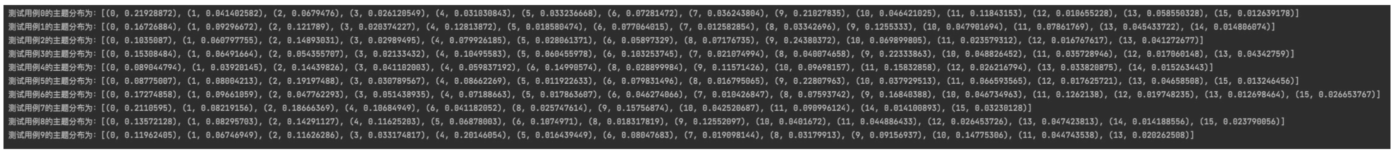


```
(0, '0.028*"弟子" + 0.025*"们" + 0.015*"咱们" + 0.015*"众" + 0.013*"大" + 0.012*"等" + 0.011*"武功" + 0.011*"话" + 0.011*"师哥" + 0.011*"倘若"')
(1, '0.027*"剑" + 0.024*"长剑" + 0.014*"出" + 0.012*"登时" + 0.012*"兵刃" + 0.011*"须" + 0.010*"竟" + 0.010*"死" + 0.009*"刺" + 0.009*"身子"')
(2, '0.017*"今日" + 0.013*"少女" + 0.011*"掌" + 0.010*"谁" + 0.010*"事" + 0.009*"出手" + 0.009*"能" + 0.009*"想" + 0.009*"请" + 0.008*"手"')
(3, '0.031*"女子" + 0.013*"并非" + 0.012*"剑法" + 0.009*"遇" + 0.009*"半分" + 0.009*"意" + 0.009*"跪" + 0.009*"纷纷" + 0.008*"少年" + 0.008*"围攻"')
(4, '0.046*"派" + 0.032*"后" + 0.024*"武功" + 0.019*"等" + 0.017*"武林" + 0.016*"无" + 0.012*"大" + 0.012*"所" + 0.011*"当" + 0.010*"功夫"')
(5, '0.045*"内力" + 0.020*"小子" + 0.018*"令" + 0.015*"自" + 0.013*"字" + 0.011*"生平" + 0.010*"任" + 0.009*"言" + 0.009*"斗" + 0.009*"击"')
(6, '0.032*"师父" + 0.015*"此刻" + 0.013*"出" + 0.011*"招" + 0.009*"起" + 0.009*"二" + 0.009*"无" + 0.008*"才" + 0.008*"大" + 0.007*"骂"')
(7, '0.015*"或" + 0.012*"相识" + 0.012*"力气" + 0.012*"嘿嘿" + 0.010*"食指" + 0.010*"无论" + 0.009*"猜" + 0.009*"停" + 0.008*"样子" + 0.007*"拆解"')
(8, '0.056*"兄弟" + 0.027*"受伤" + 0.023*"朋友" + 0.018*"欲" + 0.017*"请" + 0.016*"报仇" + 0.016*"今日" + 0.013*"两步" + 0.011*"哥哥" + 0.011*"喂"')
(9, '0.014*"笑" + 0.013*"啊" + 0.011*"爹爹" + 0.011*"小" + 0.010*"瞧" + 0.009*"声音" + 0.009*"里" + 0.009*"们" + 0.009*"起" + 0.008*"突然"')
(10, '0.035*"剑法" + 0.016*"四" + 0.015*"此" + 0.012*"手" + 0.009*"脸色" + 0.009*"此处" + 0.008*"劲力" + 0.008*"撞" + 0.007*"伤" + 0.007*"酒"')
(11, '0.014*"无法" + 0.013*"二" + 0.013*"当" + 0.009*"大喜" + 0.009*"走" + 0.009*"左" + 0.008*"大事" + 0.008*"成" + 0.008*"两" + 0.008*"未"')
(12, '0.021*"者" + 0.016*"写" + 0.016*"相距" + 0.014*"跃" + 0.011*"擒" + 0.011*"免" + 0.011*"短剑" + 0.009*"群" + 0.009*"今晚" + 0.009*"诸般"')
(13, '0.016*"於" + 0.014*"天" + 0.010*"号令" + 0.009*"万万" + 0.009*"能" + 0.008*"以" + 0.008*"暗器" + 0.007*"重" + 0.007*"後" + 0.007*"小子"')
(14, '0.021*"显然" + 0.018*"败" + 0.017*"身前" + 0.015*"西域" + 0.014*"理" + 0.012*"当时" + 0.012*"衣袖" + 0.011*"亦" + 0.011*"相交" + 0.008*"胸"')
(15, '0.037*"太" + 0.018*"摸" + 0.012*"何足" + 0.012*"何" + 0.011*"小兄弟" + 0.011*"秘密" + 0.010*"挡住" + 0.010*"疑心" + 0.008*"停步" + 0.008*"烈火"')
```

具体分布放入附录中

2. 验证分析结果

测试序号	标签	可能性最高的主题1	可能性最高的主题2	可能性最高的主题3
1	白马啸西风	**白马啸西风**	天龙八部	笑傲江湖
2	碧血剑	白马啸西风	飞狐外传	**碧血剑**
3	连城诀	天龙八部	飞狐外传	白马啸西风
4	鹿鼎记	笑傲江湖	射雕英雄传	飞狐外传
5	射雕英雄传	笑傲江湖	**射雕英雄传**	飞狐外传
6	神雕侠侣	天龙八部	飞狐外传	**神雕侠侣**
7	书剑恩仇录	白马啸西风	天龙八部	笑傲江湖
8	天龙八部	白马啸西风	飞狐外传	天龙八部
9	侠客行	天龙八部	**侠客行**	飞狐外传
10	笑傲江湖	鹿鼎记	侠客行	白马啸西风



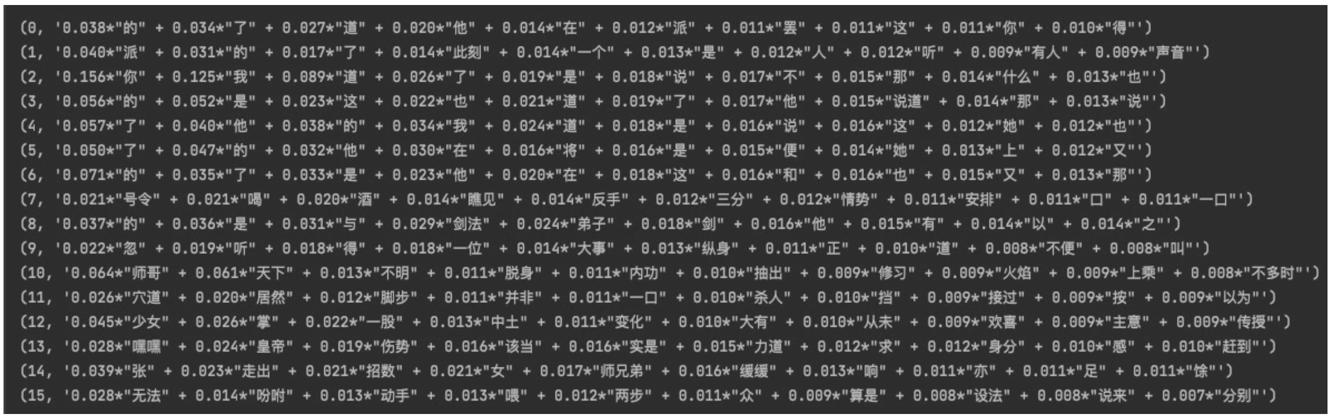
```
测试用例0的主题分布为: [(0, 0.21928872), (1, 0.041402582), (2, 0.0679476), (3, 0.026120549), (4, 0.031030843), (5, 0.033236668), (6, 0.07281472), (7, 0.036243804), (9, 0.21027835), (10, 0.046421025), (11, 0.11843153), (12, 0.010655228), (13, 0.058550328), (15, 0.012639178)]
测试用例1的主题分布为: [(0, 0.16726884), (1, 0.09296672), (2, 0.121789), (3, 0.020374227), (4, 0.12813872), (5, 0.018580474), (6, 0.077064015), (7, 0.012582854), (8, 0.03342696), (9, 0.1255333), (10, 0.047901694), (11, 0.07861769), (13, 0.045433722), (14, 0.014806074)]
测试用例2的主题分布为: [(0, 0.1035087), (1, 0.060797755), (2, 0.14893031), (3, 0.02989495), (4, 0.079926185), (5, 0.028061371), (6, 0.05897329), (8, 0.07176735), (9, 0.24380372), (10, 0.069899805), (11, 0.023579312), (12, 0.016767617), (13, 0.041272677)]
测试用例3的主题分布为: [(0, 0.15308484), (1, 0.06491664), (2, 0.054355707), (3, 0.02133432), (4, 0.10495583), (5, 0.060455978), (6, 0.103253745), (7, 0.021074994), (8, 0.040074658), (9, 0.22333863), (10, 0.048826452), (11, 0.035728946), (12, 0.017060148), (13, 0.04342759)]
测试用例4的主题分布为: [(0, 0.089044794), (1, 0.03920145), (2, 0.14439826), (3, 0.041102003), (4, 0.059837192), (6, 0.14990574), (8, 0.028899984), (9, 0.11571426), (10, 0.09698157), (11, 0.15832858), (12, 0.026216794), (13, 0.033820875), (14, 0.015263443)]
测试用例5的主题分布为: [(0, 0.08775007), (1, 0.08004213), (2, 0.19197488), (3, 0.030789567), (4, 0.08662269), (5, 0.011922633), (6, 0.079831496), (8, 0.016795065), (9, 0.22807963), (10, 0.037929513), (11, 0.066593565), (12, 0.017625721), (13, 0.04658508), (15, 0.013246456)]
测试用例6的主题分布为: [(0, 0.17274858), (1, 0.09661059), (2, 0.047762293), (3, 0.051438935), (4, 0.07188663), (5, 0.017863607), (6, 0.046274066), (7, 0.010426847), (8, 0.07593742), (9, 0.16840388), (10, 0.046734963), (11, 0.1262138), (12, 0.019748235), (13, 0.012698464), (15, 0.026653767)]
测试用例7的主题分布为: [(0, 0.2110595), (1, 0.08219156), (2, 0.18666369), (4, 0.10684949), (6, 0.041182052), (8, 0.025747614), (9, 0.15756874), (10, 0.042520687), (11, 0.090996124), (14, 0.014100893), (15, 0.03230128)]
测试用例8的主题分布为: [(0, 0.13572128), (1, 0.08295703), (2, 0.14291127), (4, 0.11625203), (5, 0.06878003), (6, 0.1074971), (8, 0.018317819), (9, 0.12552097), (10, 0.0401672), (11, 0.044886433), (12, 0.026453726), (13, 0.047423813), (14, 0.014188556), (15, 0.023790056)]
测试用例9的主题分布为: [(0, 0.11962405), (1, 0.06746949), (2, 0.11626286), (3, 0.033174817), (4, 0.20146054), (5, 0.016439449), (6, 0.08047683), (7, 0.019098144), (8, 0.03179913), (9, 0.09156937), (10, 0.14775306), (11, 0.044743538), (13, 0.020262508)]
```

具体测试结果放入附录。

心得及优化展望

1. 去除暂停词的必要性。

   首先是未去除暂停词的训练结果



```
(0, '0.038*"的" + 0.034*"了" + 0.027*"道" + 0.020*"他" + 0.014*"在" + 0.012*"派" + 0.011*"罢" + 0.011*"这" + 0.011*"你" + 0.010*"得"')
(1, '0.040*"派" + 0.031*"的" + 0.017*"了" + 0.014*"此刻" + 0.014*"一个" + 0.013*"是" + 0.012*"人" + 0.012*"听" + 0.009*"有人" + 0.009*"声音"')
(2, '0.156*"你" + 0.125*"我" + 0.089*"道" + 0.026*"了" + 0.019*"是" + 0.018*"说" + 0.017*"不" + 0.015*"那" + 0.014*"什么" + 0.013*"也"')
(3, '0.056*"的" + 0.052*"是" + 0.023*"这" + 0.022*"也" + 0.021*"道" + 0.019*"了" + 0.017*"他" + 0.015*"说道" + 0.014*"那" + 0.013*"说"')
(4, '0.057*"了" + 0.040*"他" + 0.038*"的" + 0.034*"我" + 0.024*"道" + 0.018*"是" + 0.016*"说" + 0.016*"这" + 0.012*"她" + 0.012*"也"')
(5, '0.050*"了" + 0.047*"的" + 0.032*"他" + 0.030*"在" + 0.016*"将" + 0.016*"是" + 0.015*"便" + 0.014*"她" + 0.013*"上" + 0.012*"又"')
(6, '0.071*"的" + 0.035*"了" + 0.033*"是" + 0.023*"他" + 0.020*"在" + 0.018*"这" + 0.016*"和" + 0.016*"也" + 0.015*"又" + 0.013*"那"')
(7, '0.021*"号令" + 0.021*"喝" + 0.020*"酒" + 0.014*"瞧见" + 0.014*"反手" + 0.012*"三分" + 0.012*"情势" + 0.011*"安排" + 0.011*"口" + 0.011*"一口"')
(8, '0.037*"的" + 0.036*"是" + 0.031*"与" + 0.029*"剑法" + 0.024*"弟子" + 0.018*"剑" + 0.016*"他" + 0.015*"有" + 0.014*"以" + 0.014*"之"')
(9, '0.022*"忽" + 0.019*"听" + 0.018*"得" + 0.018*"一位" + 0.014*"大事" + 0.013*"纵身" + 0.011*"正" + 0.010*"道" + 0.008*"不便" + 0.008*"叫"')
(10, '0.064*"师哥" + 0.061*"天下" + 0.013*"不明" + 0.011*"脱身" + 0.011*"内功" + 0.010*"抽出" + 0.009*"修习" + 0.009*"火焰" + 0.009*"上乘" + 0.008*"不多时"')
(11, '0.026*"穴道" + 0.020*"居然" + 0.012*"脚步" + 0.011*"并非" + 0.011*"一口" + 0.010*"杀人" + 0.010*"挡" + 0.009*"接过" + 0.009*"按" + 0.009*"以为"')
(12, '0.045*"少女" + 0.026*"掌" + 0.022*"一股" + 0.013*"中土" + 0.011*"变化" + 0.010*"大有" + 0.010*"从未" + 0.009*"欢喜" + 0.009*"主意" + 0.009*"传授"')
(13, '0.028*"嘿嘿" + 0.024*"皇帝" + 0.019*"伤势" + 0.016*"该当" + 0.016*"实是" + 0.015*"力道" + 0.012*"求" + 0.012*"身分" + 0.010*"感" + 0.010*"赶到"')
(14, '0.039*"张" + 0.023*"走出" + 0.021*"招数" + 0.021*"女" + 0.017*"师兄弟" + 0.016*"缓缓" + 0.013*"响" + 0.011*"亦" + 0.011*"足" + 0.011*"馀"')
(15, '0.028*"无法" + 0.014*"吩咐" + 0.013*"动手" + 0.013*"喂" + 0.012*"两步" + 0.011*"众" + 0.009*"算是" + 0.008*"设法" + 0.008*"说来" + 0.007*"分别"')
```

   充斥大量无意义的词，但是哈工大、川大等的暂停词库数据又过于庞大，为了提升运行效率。我这里去除了金庸小说中常见的暂停词。

2. 测试模型和真实结果有出入的原因

   在验证分析结果中和真实的标记出入存在。究其原因，其一是因为武侠小说的相关性比较高。其二，是因为有些小说比较短，得到的主题词分布集中在一些日常话语中，在其他段落中也会有较高频率的出现。

   而一些特殊人名，如韦小宝、郭靖、令狐冲等名字，精确度就很高。

3. 根据心得2，可以进一步得到优化方向

   根据16本金庸小说统一得到的主题词，并将这些词在小说中去除，重新训练LDA，并带入新的测试集进行测试，并比较欧式距离，循环往复，直至测试结果和真实标签的欧式距离小于某个置信度较高的值，循环结束。

参考内容

1. https://zhuanlan.zhihu.com/p/31470216
2. https://www.cnblogs.com/Luv-GEM/p/10881838.html

附录

主题分布

```
(0, '0.028*"弟子" + 0.025*"们" + 0.015*"咱们" + 0.015*"众" + 0.013*"大" + 0.012*"等" + 
0.011*"武功" + 0.011*"话" + 0.011*"师哥" + 0.011*"倘若"')
(1, '0.027*"剑" + 0.024*"长剑" + 0.014*"出" + 0.012*"登时" + 0.012*"兵刃" + 0.011*"须" + 
0.010*"竟" + 0.010*"死" + 0.009*"刺" + 0.009*"身子"')
(2, '0.017*"今日" + 0.013*"少女" + 0.011*"掌" + 0.010*"谁" + 0.010*"事" + 0.009*"出手" + 
0.009*"能" + 0.009*"想" + 0.009*"请" + 0.008*"手"')
(3, '0.031*"女子" + 0.013*"并非" + 0.012*"剑法" + 0.009*"遇" + 0.009*"半分" + 0.009*"意" + 
0.009*"跪" + 0.009*"纷纷" + 0.008*"少年" + 0.008*"围攻"')
(4, '0.046*"派" + 0.032*"后" + 0.024*"武功" + 0.019*"等" + 0.017*"武林" + 0.016*"无" + 
0.012*"大" + 0.012*"所" + 0.011*"当" + 0.010*"功夫"')
(5, '0.045*"内力" + 0.020*"小子" + 0.018*"令" + 0.015*"自" + 0.013*"字" + 0.011*"生平" + 
0.010*"任" + 0.009*"言" + 0.009*"斗" + 0.009*"击"')
(6, '0.032*"师父" + 0.015*"此刻" + 0.013*"出" + 0.011*"招" + 0.009*"起" + 0.009*"二" + 
0.009*"无" + 0.008*"才" + 0.008*"大" + 0.007*"骂"')
(7, '0.015*"或" + 0.012*"相识" + 0.012*"力气" + 0.012*"嘿嘿" + 0.010*"食指" + 0.010*"无论" 
+ 0.009*"猜" + 0.009*"停" + 0.008*"样子" + 0.007*"拆解"')
(8, '0.056*"兄弟" + 0.027*"受伤" + 0.023*"朋友" + 0.018*"欲" + 0.017*"请" + 0.016*"报仇" + 
0.016*"今日" + 0.013*"两步" + 0.011*"哥哥" + 0.011*"喂"')
(9, '0.014*"笑" + 0.013*"啊" + 0.011*"爹爹" + 0.011*"小" + 0.010*"瞧" + 0.009*"声音" + 
0.009*"里" + 0.009*"们" + 0.009*"起" + 0.008*"突然"')
(10, '0.035*"剑法" + 0.016*"四" + 0.015*"此" + 0.012*"手" + 0.009*"脸色" + 0.009*"此处" + 
0.008*"劲力" + 0.008*"撞" + 0.007*"伤" + 0.007*"酒"')
(11, '0.014*"无法" + 0.013*"二" + 0.013*"当" + 0.009*"大喜" + 0.009*"走" + 0.009*"左" + 
0.008*"大事" + 0.008*"成" + 0.008*"两" + 0.008*"未"')
(12, '0.021*"者" + 0.016*"写" + 0.016*"相距" + 0.014*"跃" + 0.011*"擒" + 0.011*"免" + 
0.011*"短剑" + 0.009*"群" + 0.009*"今晚" + 0.009*"诸般"')
(13, '0.016*"於" + 0.014*"天" + 0.010*"号令" + 0.009*"万万" + 0.009*"能" + 0.008*"以" + 
0.008*"暗器" + 0.007*"重" + 0.007*"後" + 0.007*"小子"')
(14, '0.021*"显然" + 0.018*"败" + 0.017*"身前" + 0.015*"西域" + 0.014*"理" + 0.012*"当时" + 
0.012*"衣袖" + 0.011*"亦" + 0.011*"相交" + 0.008*"胸"')
(15, '0.037*"太" + 0.018*"摸" + 0.012*"何足" + 0.012*"何" + 0.011*"小兄弟" + 0.011*"秘密" + 
0.010*"挡住" + 0.010*"疑心" + 0.008*"停步" + 0.008*"烈火"')
```

验证分析

```
测试用例0的主题分布为：[(0, 0.21928872), (1, 0.041402582), (2, 0.0679476), (3,
0.026120549), (4, 0.031030843), (5, 0.033236668), (6, 0.07281472), (7, 0.036243804),
(9, 0.21027835), (10, 0.046421025), (11, 0.11843153), (12, 0.010655228), (13,
0.058550328), (15, 0.012639178)]
测试用例1的主题分布为：[(0, 0.16726884), (1, 0.09296672), (2, 0.121789), (3, 0.020374227),
(4, 0.12813872), (5, 0.018580474), (6, 0.077064015), (7, 0.012582854), (8, 0.03342696),
(9, 0.1255333), (10, 0.047901694), (11, 0.07861769), (13, 0.045433722), (14,
0.014806074)]
测试用例2的主题分布为：[(0, 0.1035087), (1, 0.060797755), (2, 0.14893031), (3, 0.02989495),
(4, 0.079926185), (5, 0.028061371), (6, 0.05897329), (8, 0.07176735), (9, 0.24380372),
(10, 0.069899805), (11, 0.023579312), (12, 0.016767617), (13, 0.041272677)]
测试用例3的主题分布为：[(0, 0.15308484), (1, 0.06491664), (2, 0.054355707), (3,
0.02133432), (4, 0.10495583), (5, 0.060455978), (6, 0.103253745), (7, 0.021074994), (8,
0.040074658), (9, 0.22333863), (10, 0.048826452), (11, 0.035728946), (12, 0.017060148),
(13, 0.04342759)]
测试用例4的主题分布为：[(0, 0.089044794), (1, 0.03920145), (2, 0.14439826), (3,
0.041102003), (4, 0.059837192), (6, 0.14990574), (8, 0.028899984), (9, 0.11571426),
(10, 0.09698157), (11, 0.15832858), (12, 0.026216794), (13, 0.033820875), (14,
0.015263443)]
测试用例5的主题分布为：[(0, 0.08775007), (1, 0.08004213), (2, 0.19197488), (3,
0.030789567), (4, 0.08662269), (5, 0.011922633), (6, 0.079831496), (8, 0.016795065),
(9, 0.22807963), (10, 0.037929513), (11, 0.066593565), (12, 0.017625721), (13,
0.04658508), (15, 0.013246456)]
测试用例6的主题分布为：[(0, 0.17274858), (1, 0.09661059), (2, 0.047762293), (3,
0.051438935), (4, 0.07188663), (5, 0.017863607), (6, 0.046274066), (7, 0.010426847),
(8, 0.07593742), (9, 0.16840388), (10, 0.046734963), (11, 0.1262138), (12,
0.019748235), (13, 0.012698464), (15, 0.026653767)]
测试用例7的主题分布为：[(0, 0.2110595), (1, 0.08219156), (2, 0.18666369), (4, 0.10684949),
(6, 0.041182052), (8, 0.025747614), (9, 0.15756874), (10, 0.042520687), (11,
0.090996124), (14, 0.014100893), (15, 0.03230128)]
测试用例8的主题分布为：[(0, 0.13572128), (1, 0.08295703), (2, 0.14291127), (4, 0.11625203),
(5, 0.06878003), (6, 0.1074971), (8, 0.018317819), (9, 0.12552097), (10, 0.0401672),
(11, 0.044886433), (12, 0.026453726), (13, 0.047423813), (14, 0.014188556), (15,
0.023790056)]
测试用例9的主题分布为：[(0, 0.11962405), (1, 0.06746949), (2, 0.11626286), (3,
0.033174817), (4, 0.20146054), (5, 0.016439449), (6, 0.08047683), (7, 0.019098144), (8,
0.03179913), (9, 0.09156937), (10, 0.14775306), (11, 0.044743538), (13, 0.020262508)]
```

源代码

```python
import jieba, os, re
import numpy as np
from gensim import corpora, models

def D_dispose():
    fileName = "./dic_train_data.txt"
    if not os.path.exists('./dic_train_data.txt'):
        outputs = open(fileName, 'w', encoding='UTF-8')
        DSRoot = "./data"
        catalog = "inf.txt"
        with open(os.path.join(DSRoot, catalog), "r", encoding='utf-8') as f:
            all_files = f.readline().split(",")
            print(all_files)
        for name in all_files:
            with open(os.path.join(DSRoot, name + ".txt"), "r", encoding='utf-8') as f:
                file_read = f.readlines()
                train_num = len(file_read)
                choice_index = np.random.choice(len(file_read), train_num,
replace=False)
                sw = ["的", "了", "在", "是", "我", "有", "和", "就",
      "不", "人", "都", "一", "一个", "上", "也", "很", "到", "说", "要", "去", "你",
      "会", "着", "没有", "看", "好", "自己", "这","罢"
,"这",'在','又','在','得','那','他','她','不','而','道','与','之','见','却','问','可','但'
                      ,'没','啦','给','来','既','叫','只','中','么','便'
                      ,'听','为','跟','个','甚','下','还','过','向','如此'
                      ,'已','位','对','如何','将','岂','哪','似','以免','均'
                      ,'虽然','即','由','再','使','从','麼','其实','阿','被']
                for train in choice_index[0:train_num]:
                    line = file_read[train]
                    line = re.sub('\s', '', line)
                    line = re.sub('[\u0000-\u4DFF]', '', line)
                    line = re.sub('[\u9FA6-\uFFFF]', '', line)
                    line= re.sub('[\u9FA6-\uFFFF]', '', line)
                    for a in sw:
                        line= re.sub(a, '', line)
                    if len(line) == 0:
                        continue
                    seg_list = list(jieba.cut(line, cut_all=False))
                    line_seg = ""
                    for term in seg_list:
                        line_seg += term + " "
                    outputs.write(line_seg.strip() + '\n')
        outputs.close()
        print("处理完原始数据")


if __name__ == "__main__":
    D_dispose()
    #整理成gensim需要的输入格式
    fr = open('./dic_train_data.txt', 'r', encoding='utf-8')
    train = []
    for line in fr.readlines():
        line = [word.strip() for word in line.split(' ')]
        train.append(line)
```

```
#训练LDA模型
dictionary = corpora.Dictionary(train)
# corpus是把每本小说ID化后的结果，每个元素是新闻中的每个词语，在字典中的ID和频率
corpus = [dictionary.doc2bow(text) for text in train]
lda = models.LdaModel(corpus=corpus, id2word=dictionary, num_topics=16)
topic_list_lda = lda.print_topics(16)
print("16本小说主题分布为：\n")
for topic in topic_list_lda:
    print(topic)

fileTest = "./dic_test_data.txt"
news_test = open(fileTest, 'r', encoding='UTF-8')
test = []
for line in news_test:
    line = [word.strip() for word in line.split(' ')]
    test.append(line)
for text in test:
    corpus_test = dictionary.doc2bow((text))
corpus_test = [dictionary.doc2bow(text) for text in test]
topics_test = lda.get_document_topics(corpus_test)

for i in range(10):
    print("测试用例"+str(i)+'的主题分布为：'+str(topics_test[i]))

fr.close()
news_test.close()
```