基于 LDA 的学术会议主题发现

游沛杰 13307130325

2016-1-8

Contents

1 选题介绍

本次项目我们实现了一个论文主题发现系统，通过对近年学术会议论文使用 LDA 提取主题并分析，从而发现相关领域的研究进展。

之前考虑过的其他想法如下:

1. 研究小学语文的连词成句 *，让计算机参加小升初？
2. 可视化关键词在文章中出现的位置，进一步分析各种词语的重要性来研究人的阅读习惯。
3. “自动 (好吃的) 菜名生成器”以及相关分析，进一步通过菜名和食谱文章用词来描述不同研究八大菜系特点并在中国地图上可视化等等。
4. 梵语-英语/粤语-普通话翻译机

之前听说过的自然语言处理模型包括 LDA, word embedding(word2vec thing), RNN-LSTM 等等，但是对具体算法和效果并没有详细的认识，在这里我们通过这次课程项目主要想学习 latent Dirichlet allocation 模型。

考虑到其他任务难以实现或者没有数据来源。而主题发现这一题目，既可以让我学习新的模型 (latent Dirichlet allocation)；又可以运用我们上课讲到的 NLTK 处理语料；同时可以完成项目并对计算机领域加深了解，所以我最终选择了实现学术会议主题发现系统。

Table 1: 项目实现情况简介

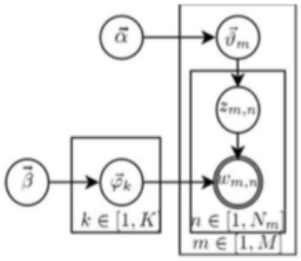
语言	Python
辅助工具	NLTK, PDFMiner
环境	Mac OS X

2 算法背景

2.1 文本建模

LDA 模型中文本生成分为几步:

1. 对每个文章生成一个 doc-topic 模型（有 Dirichlet 先验分布）
2. 每一次按 doc-topic 分布随机一个 topic，然后按对应的 topic-word 概率生成一个词。



2.2 Dirichlet 分布与多项分布

多项分布的共轭先验是 Dirichlet 分布，即若先验分布是 Dirichlet 分布，条件概率是多项分布，则按照贝叶斯公式后验概率与先验概率形式相同为了 ichlet 分布，具有良好性质，其中多项分布形式为：

$$p(x;n) = \frac{N!}{n_1! * n_2 * ... * n_m!} x_1^{n_1} * x_2^{n_2} * ... * x_m^{n_m}$$

Dirichlet 分布:

$$p(x;n) = \frac{\Gamma(N)}{\Gamma(n_1) * \Gamma(n_2) * ... * \Gamma(n_m)} x_1^{n_1} * x_2^{n_2} * ... * x_m^{n_m}$$

doc-topic, topic-word 都是这样的 Dirichlet-Multinomial 共轭分布。

2.3 Gibbs 采样

知道 Gibbs 采样。

3 实现过程

以下是实现具体细节，按实现时间排序。

3.1 语料收集

以下我们主要选取 NIPS(Neural Information Processing Systems)2015 作为本次的语料库。因为 NIPS 网站提供链接 [1]，方便我们直接下载全部论文。

使用 Python 爬虫，从网页上找到所有论文的链接然后下载，因为 NIPS 网站排版具有一定格式 (所有 paper 链接在一个 <ul> 标签中)，我们先把这段 HTML 代码找出来，然后找出里面的 url 并下载对应文件。

3.2 PDF 数据处理

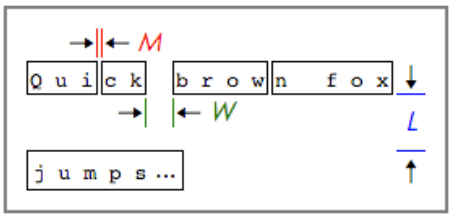
由于我们得到的原始语料库是 PDF 文件，所以需要先提取里面的文本信息。一开始使用 pyPdf 直接提取文字，但是发现不能保留行内空格，如图：

```
AnomalousTiePlateDetectionForRailroadInspectionYingLiandSharathPankantiIBMT.J.
USAyingli@us.ibm.comAbstractThispaperdescribesourlatestworkonidentifyinganomal
pectionusingmachinevisiontechnology.Specifically,wehavedevelopedacompletelyauto
eplateswithanomalousspikingpatternsusingvariousvideoanalytics.Inparticular,
```

这时候我们有两种解决方案：

1. 没有空格的英文分词
2. 尝试另外个工具包

后来我们使用一个叫作 PDFMiner 的 Python 工具包，通过判断 PDF 中两个字符之间的间隔大小来进行分词。




```
Introduction
The goal of visual texture synthesis is to infer a generating proces
then allows to produce arbitrarily many new samples of that texture.
quality of the synthesised texture is usually human inspection and t
sised if a human observer cannot tell the original texture from a sy
In general, there are two main approaches to find a texture generatin
to generate a new texture by resampling either pixels [5, 28] or who
texture. These non-parametric resampling techniques and their numero
ments (see [27] for review) are capable of producing high quality na
```

通过 PDF 转换成文本数据，我们也把论文中的图片和一些数学公式去掉了，剩下的基本是文字内容。

把数字和标点符号去掉，可以通过简单的正则表达式匹配完成。因为数字和标点对我们的 topic model 而言提供的信息量很少，所以这样处理十分合理。同时我们也统一转换成小写字母。因为 LDA 是一种词袋模型 (bag of words)，最终只需要词频分布，所以我们不需要断句等工作。

```
texture synthesis using convolutional neural networks centre for integrati
ubingen germany bernstein center for computational neuroscience t ubingen
information processing university of t ubingen germany leon a gatys leon g
centre for integrative neuroscience university of t ubingen germany bernst
neuroscience t ubingen germany max planck institute for biological cyberne
college of medicine houston tx usa matthias bethge centre for integrative
germany bernstein center for computational neuroscience t ubingen germany
cybernetics t ubingen germany abstract here we introduce a new model of na
spaces of convolutional neural networks optimised for object recognition s
perceptual quality demonstrating the generative power of neural networks t
```

相对中文而言，学术论文大多是用英文书写的，所以分词不是必须的。
文本提取后 部分 词频统计如图：

```
em 600
lstm 498
cnn 445
bandit 421
lda 257
hinton 232
nlp 30
language 311
```

3.3 单词筛选

一个直觉的想法就是如果我们把在学术领域中没有特殊含义的词去掉以后，最终效果会有显著提升。

1. 回想在 PDF 转换到文本数据的过程中，我们可能引入了一些**噪声**，如可能把公式中的字母也包含进来了，或因为我们的转换程序中间隔设置不合理有错误的单词提取结果。能否把这些” 单词” 去掉？如下例子中的的 cid

```
deterministic learning algorithm can achieve this
the regret rt then becomes a ran dom variable that
sting literature on non stochastic bandits is
ned as cid t cid cid t it t cid t t cid rt max i k
omness injected by the learner proving bounds on
```

针对以上问题，我根据单词长度进行筛选

- 单词小于等于 2 个字符，认为对于 LDA 没有特殊含义，直接去掉 (去掉的单词包括 a, I, am, be, he 还有德语 de 等)
- 单词长度大于 15 字符，认为是 PDF 转换结果出错，所以直接去掉。(剩下的 PDF 转换出错结果占比例较少)

2. 另外英文中普遍存在有禁用词 (stop word) 的情况，在这里我使用 NLTK 的禁用词语料库进行处理：

```
sw7 = stopwords.words('english')
word_dict = [word for word in word_list if word not in sw7]
```

在文献领域也有领域的禁用词，如 cid, reference, abstract, 针对这些词我建立了自己的禁用词表。

3. 英文与中文不同在于英文还有形式上的变化，为了让结果更准确，决定对词提取语干/词干。根据课本 NLTK 介绍，我们使用 PorterStemmer 作为我们词干提取器 (参照了课本” 规范化文本” 一节)。

```
porter = nltk.PorterStemmer()
word_dict = [porter.stem(word) for word in word_dict]
```

这里有一个问题，如关键词 “R-CNN”(Region Convolutional Neural Network) 会首先分成 “R CNN” 两个单词 (去掉标点字符), 然后因为字母 R 长度为 1，会被删去，剩下 “CNN”，那么这样的结果是否合理？可以在后续研究中探讨。

3.4 初步尝试 LDA 效果

以上处理完以后，我们对每篇论文进行词频统计，并把结果存储为稀疏矩阵格式，使用 lda 类进行训练，结果如下 (k=7，显示每个类别最可能生成的单词)

```
Topic 0 machine learning: learn data set label use classi featur predict cation dataset exampl function model
space train method kernel test differ base number approach perform problem class

Topic 1 optimization    : algorithm optim function problem set gradient method bound convex iter converg comput
use approxim solut step object time descent updat case follow number submodular result

Topic 2 matrix          : matrix algorithm estim sampl rank error spars norm result log problem theorem random
model set statist vector comput dimension condit data bound follow low analysi

Topic 3                 : algorithm bound set distribut regret log optim problem learn sampl theorem function
risk result follow loss probabl bandit gener case arm estim time game onlin

Topic 4 graph model     : model graph state time node network edg learn use observ algorithm control comput
polici set dynam trajectori variabl system path gener process structur transit neuron

Topic 5 probability?    : model distribut sampl estim log approxim use infer posterior function variat method
gaussian paramet likelihood data process comput latent mean variabl gradient point stochast topic

Topic 6 deep learning   : network model imag train layer use neural learn deep input gener convolut object
output sequenc propos perform comput result lstm represent recurr predict cnn transform
```

我们发现，虽然 “algorithm” 这个单词在每个类别中排名都比较靠前，但是其实在文献语料库中普遍存在，所以对我们判断一篇文章的主题并没有太大帮助，考虑如何改进把这些词去掉？另外 topic 3 不知道是什么主题。

3.5 改进 1

我首先的想法是给词频**设置一个区间**，去掉频率出现过高的词和出现次数过少的词。

那么出现次数过少的词又是什么呢？有部分作者人名和提取文本失败的结果。考虑在一百万词中只出现 1 次，可能反而降低了 LDA 效果。所以对低频词也有阈值来筛选，去掉长尾。[1]

另外我们发现有两个字母的关键词，应该出现在我们的词典中，但是被筛走了，比如 EM（代表 expectation maximization 算法）。所以我们又建立了一个小词典来加入这些之前处理中被排除的词。

3.6 改进后效果

详见后面 “效果分析”

[1] 最后实验中我们使用的频率区间为 (7, 4000) 次。

3.7 More?

以上结果看起来比较科学，但是毕竟是人手筛选过的单词列表，不够自动化，于是进一步考虑能否让计算机自动计算哪些词对论文主题影响最大的呢？

根据我的分析，可以有两类不同的词筛选方法：

1. 在 NIPS2015 中出现，但是一般英文文章中出现概率比较低 的词 (如报纸，小说)。
2. 在一篇文章中经常出现但是在整个语料库中分布不均衡 的词。

以上是受到同学写的 TF-IDF 影评分析启发。

由于 NLTK 已自带语料库，所以我打算继续充分利用这个工具包。针对以上第一个思路，按两个语料库出现单词的词频之比来排序:

$$\frac{p_{NIPS}}{\alpha + p_{others}}$$

增加 α 来平滑概率，因为专业名词如 CNN，NLP 在普通文章中不可能出现。然后去掉比值较小的单词，以下是 NIPS 语料库和古腾堡语料库对比的结果，比值高代表在 NIPS 中更容易出现 (方便起见使用频数之比代替频率之比)：

```
2459.0 training          0.00856291883842 man
2394 optimization        0.00852514919011 forth
2380 theorem             0.00841167738743 land
2360 rst                 0.00839532412327 said
2347 neural              0.00823170731707 day
2145 gradient            0.00768049155146 things
1989 parameters          0.00698324022346 shall
1988 classi              0.00641539695269 voice
1954 optimal             0.00610376398779 thing
1792 stochastic          0.00607902735562 went
1695 max
1693 gaussian
1661 cation
```

去掉比值小于 1 的单词后使用 LDA 结果发现结果基本不变，并且看起来已经比较合理，于是以下针对第 2 点的 TF-IDF 没有实现：

$$p(x_i) * \log \frac{|D|}{\alpha + |D_i|}$$

4 效果及分析

这里只给出最终程序的部分结果，更多中间结果和分析过程在上一节中提到，文中给出的图如果不够清晰可以打开附件中的原图片查看。

4.1 How to Run It?

To run my program, you may need to:

- Install Python2, NLTK, PDFMiner, lda

- Download my source code
- Download data from here[2] to directory data/
- Run preprocessing program in src/pre directory
- Run src/gen to generate dict and input data of lda
- Run src/run.py to demostrate the lda model

4.2 数据集效果

在这次项目开发中，我们使用 NIPS2015 的论文作为语料库

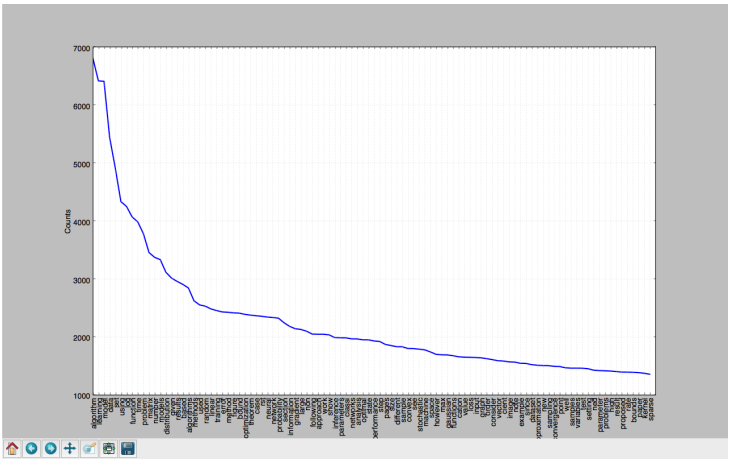
Table 2: 文章字数 (word) 统计

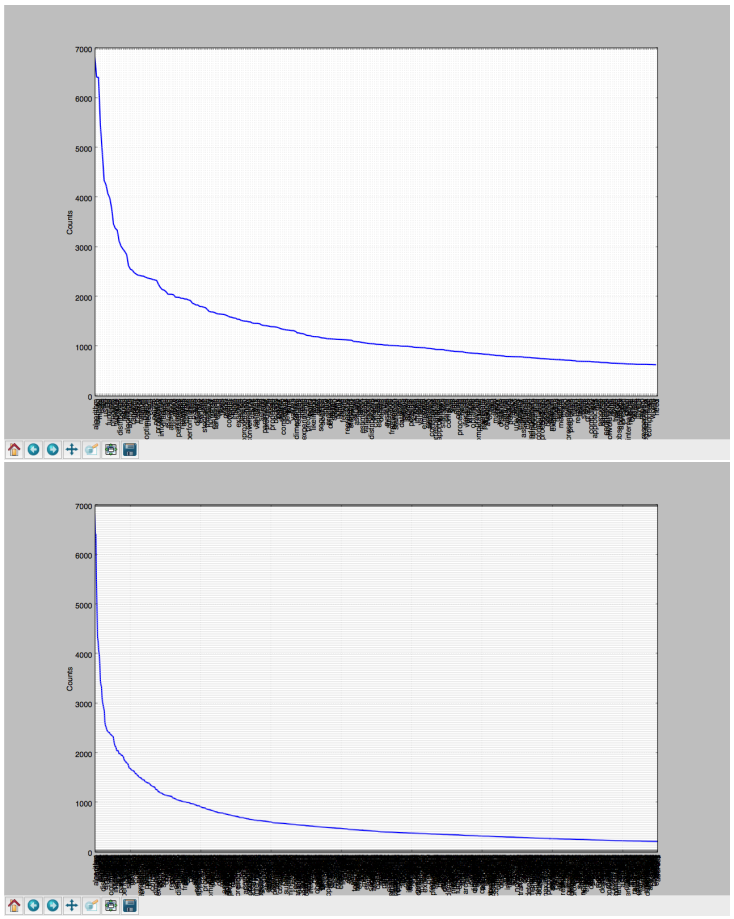
文章数	总单词数	筛选后的字数	平均每篇文章单词数
403	3229239	2148449	5331

Table 3: 字典 (dictionary) 统计

文章数	总字典	筛选后条目	每篇文章出现的不同词
403	30000 以上	9956 - 10537	600-750

首先是使用 NLTK 对单词/词根频率统计结果 (依次为频率排名前 100，前 300，前 1000 的词频曲线)：

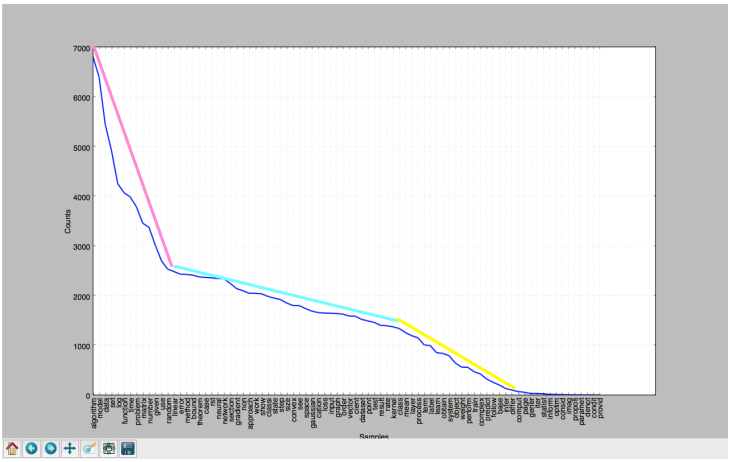




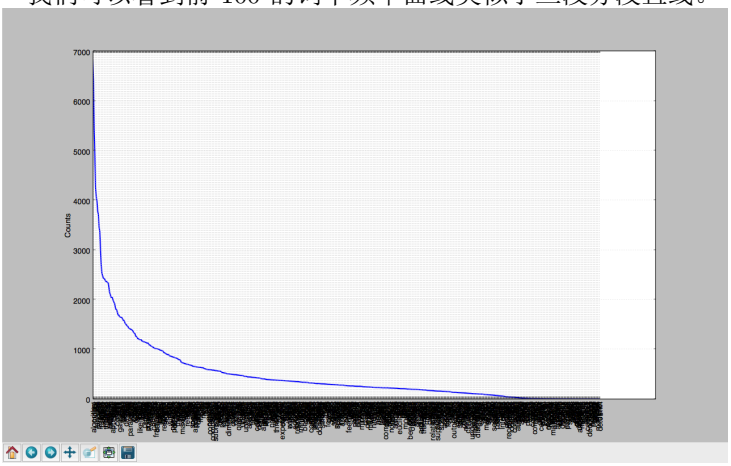
可以发现，排名与词频基本上是呈反比关系，一定层度上符合 Zipf 法则 [3]：

$$f \propto \frac{1}{r}$$

词干的频率曲线图与上面类似，但是在前 100 排名时，曲线形状不正常，如下图所示：



我们可以看到前 100 的词干频率曲线类似于三段分段直线。



而前 1000 的排名总体效果还是与双曲线接近。

4.3 LDA 主题效果

Latent Dirichlet Allocation 训练时的 log likelihood 曲线如图，可以发现在迭代 800-1000 次以后已经收敛了 (实验中一开始迭代了 1500 次，但是从效果上来看有 over-fitting 的嫌疑，后来只迭代了 800 次)

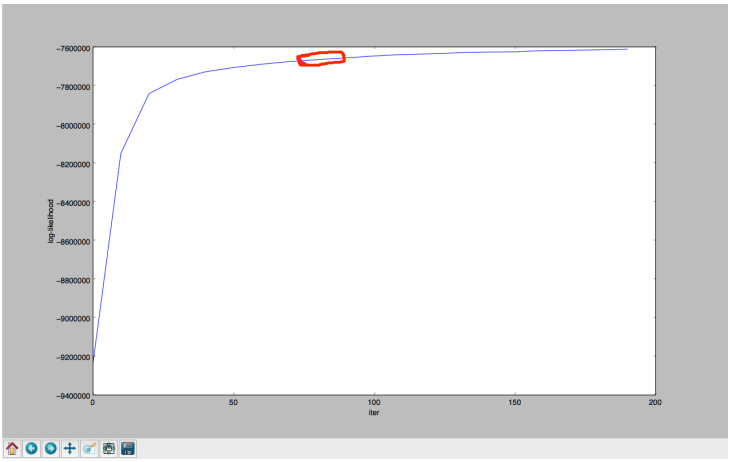


Table 4: LDA 运行情况

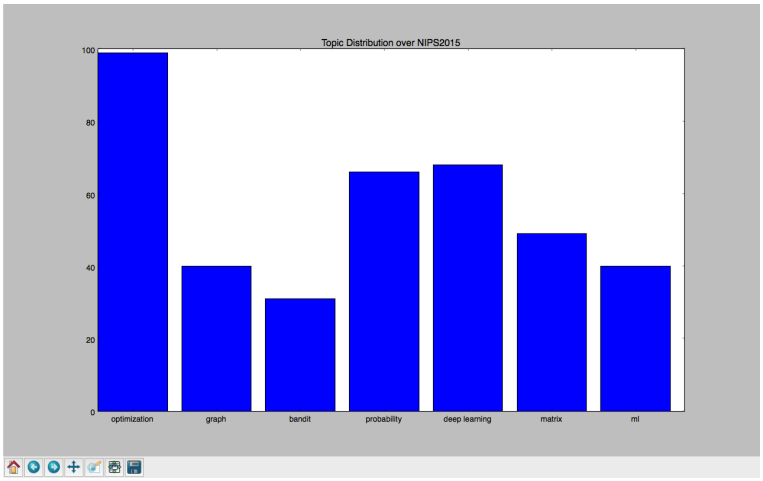
文章数	总字典	每篇文章出现的不同词	topic	运行时间
403	9956 - 10537	600-750	7	138.58s

主题相关的词如下：

```
------optimaztion          : optim bound problem distribut method algorithm result converg comput case convex
follow sampl gener point gradient linear rate step iter theorem approxim estim set non

------graph model          : graph node time edg network tree number infer cluster submodular partit variabl
weight structur model given problem greedi size random constraint algorithm order comput set

------online learning       : regret action bandit polici arm state bound game problem risk reward optim observ
time decis agent control expect valu inform probabl player cost strategi base

------probability & stat    : distribut estim sampl model infer approxim state process posterior paramet
gaussian variat likelihood time method comput latent observ variabl mean gener figur prior bayesian use

------deep learning     : network train imag layer neural input deep gener object model output convolut use
sequenc propos perform comput predict represent result featur weight map word dataset

------matrix               : matrix rank estim spars error norm vector dimension problem sampl low matric
statist result gaussian comput random condit column observ entri high paramet spectral theorem

------machine learning      : label classi featur kernel predict cation dataset train space exampl embed metric
test loss class queri method perform task margin differ base score number set
```

归纳出来的主题 (7 个) 分别是 (0: optimization, 1: graph model, 2: online learning(bandit thing), 3: probability & statistics, 4: deep learning, 5: matrix factorization, 6: machine learning) ，我们可以明显看到主题与对应词相关性。

Topic 在语料库中的分布 (每篇文章取概率最大的主题)：



我们发现第一类主题 (优化) 在语料库中出现的可能性最高，优化问题在机器学习的文献中普遍存在，所以这个主题中的词都是在语料库中常见的词，那么像这样有一个主题包含有全部常用词的情况是否合理呢？回想起隐含 Dirichlet 分布的文本模型，先从 doc-topic 中抽取一个 topic，然后按照 topic-word 分布生成词，如果一个主题把所有常用词都包含了，让别的主题更专注于更专业的词，可能最后产生的概率更大。

在训练 LDA 时我们使用 9:1 随机把文章分到训练集与测试集中，以下是在测试集上的 LDA 对单篇文章主题分类效果：

```
test
0.00    0.00    0.22    0.30    0.00    0.00    0.00
0.48    0.00    0.00    0.28    0.00    0.00    0.00
0.43    0.00    0.00    0.00    0.00    0.21    0.00
0.39    0.42    0.00    0.00    0.00    0.00    0.00
0.30    0.00    0.00    0.26    0.00    0.00    0.00
0.30    0.00    0.00    0.00    0.00    0.39    0.00
0.72    0.00    0.00    0.00    0.00    0.00    0.00
0.35    0.30    0.00    0.00    0.00    0.25    0.00
0.68    0.00    0.00    0.00    0.00    0.00    0.00
```

另外我们发现一个 topic 的相关词有 “Bandit”, “arm” 等，一开始不知道以为是算法出错了，结果上网搜索后发现是一种 online learning 的方法，主要应用在计算广告中，这也说明我们的系统真的起到了主题发现的作用，通过 topic model 而不是人眼浏览论文就能发现学术会议的几大主题，十分方便。

5 The End

到这里我们基本完成了本次 “提取 NIPS 主题” 的任务，以下是总结。

5.1 总结

在这次项目中我们实现了一个文献主题提取的系统，并更深入了解了 LDA 模型。LDA 作为流行的 topic model 一种，在新闻自动分类，用户评论关键词提取，甚至个性化推荐中有重要应用。

另外，做到后面我清楚认识到主题离真正语义上的 “研究方向” 还是有一定距离的，topic 主要把语义相近的词归为一个类别，如上面主题中的 cnn, lstm 虽然都属于深度学习，但是一个主要是图像处理，一个主要是语音和语言的处理，真正想要发现 “研究方向” 应该要区分这样的词到不同 topic 中去。

自然语言处理对我来说一直是一个比较繁琐的过程，在本次课程项目，我的时间主要花在获取数据和文本处理上面，可见有专人帮忙构造语料库可以节约大量时间。

这是我第一次没完全看懂算法就直接拿来用的项目，之前其他课程中我都是手写的 (比如语音处理的 MFCC，比如 HMM)，但是写一个 project 十分的慢。这个学期我们只有 18 周，到今天为止同学们已经连续忙碌考试和课程项目了超过一个月了，做完这个还有 2 个项目要完成。所以我想试一下用网上给出的机器学习工具包来减轻工作量 (虽然文本处理也花了不少时间)。会用一个算法，远比了解算法原理细节要难，等完成所有项目后，打算在假期继续研究 LDA 原理 [6]，目前知道 Gibbs 采样但是不知道在 LDA 中有什么用。

5.2 Counterclaim?

现在回顾项目实现过程中，发现有一些可以改进的地方：

- PDF 转文本数据结果不应该这么差 (在有些文章中还是不能把单词分开)。这是不应该出现的，因为如果手工对这些 PDF 文件进行选择-赋值-粘贴到.txt 文件，总是有办法能分开的。
- 手工筛选词典不太科学，因为 LDA 就是要自动发现单词和主题之间的关系，应该一开始就只使用 TF-IDF 来过滤。
- 本来想用微软的 lightlda(据说有很多优化，非常快，所以才想做 LDA 方面的 project)，但是始终配置安装不成功，最后使用的一个叫 lda 的 Python 模块，比较慢 (400 篇文章, 10000 种单词，总共 1,000,000 词，7 个主题，训练了 138.58 秒)。
- 因为字典大小 (数据维数) 始终在变，所以难以比较不同字典上使用的 LDA 模型，下次做项目应该使用相同大小的字典从而比较效果？
- 因为效果比较客观，训练比较慢，所以没有对不同主题数条件下的结果进行对比。

在即将完成本项目的时候，突然发现类似的方法已经有人做过了 [4]。对比一下提取出来的主题，发现大部分主题相同，但是有一类以 “Bandit”, “arm” 等我认为是 online learning 的主题，但是网站上标成了 theory。

总体来说我们实现的 LDA 是有效的。

6 Thank You

感谢您的仔细阅读！
Report generate in LaTeX
View this project on GitHub[5]
Full data used in this project is available here[2]
高清大图可以查看其他附件。

References

[1] http://papers.nips.cc/book/advances-in-neural-information-processing-systems-27-2015

[2] notavailabienow

[3] Manning C D, Schutze H, et al. 统计自然语言处理基础 [M]. 电子工业出版社, 2005.

[4] http://cs.stanford.edu/people/karpathy/nips2015/

[5] https://github.com/kjkszpj/NlPS

[6] Steven Bird, Python 自然语言处理.

[7] LDA 数学八卦