基于 KMeans 算法与 LDA 算法的文本主题聚类

李鹏，10175501102@stu.ecnu.edu.cn

November 13, 2019

实验目的

本次实验是在第二次分类实验的基础上进行文本聚类。文本聚类主要是依据著名的聚类假设：同类的文档相似度较大，而不同类的文档相似度较小。作为一种无监督的机器学习方法，聚类由于不需要训练过程，以及不需要预先对文档手工标注类别，因此具有一定的灵活性和较高的自动化处理能力，已经成为对文本信息进行有效地组织、摘要和导航的重要手段，为越来越多的研究人员所关注。

本次实验的聚类任务是依据话题聚类，使用机器学习算法尽可能将讨论相同话题的文档聚到相同的簇，讨论不同的文档聚到不同的簇。主题模型（话题模型）在文本聚类的框架下又有自身的特点，它能够采用概率图模型进行概率建模，把话题当作隐变量来处理，由此产生了 LSA、pLSA、LDA 等一系列主题模型。

本次实验还要求利用数据的真实标签对聚类结果进行评价，其中主要涉及 FMI 与 RI 指标。

实验方法

实验方法主要涉及文本特征提取方法，聚类算法，结果评价方法。

文本特征提取

由于之前的文本分类实验已经提取了文本的特征，本次实验将主要基于这些提取的特征进行文本聚类实验。本次实验利用的文本特征有：

- 基于词频加权的向量空间模型
- 基于 TF-IDF 的向量空间模型
- 基于词向量平均的文档表示

其中基于词频加权的向量空间模型根据文档中的词的词频对基础的向量空间模型进行加权，这里的文档已经利用 TFIDF 进行了关键词提取，每个文档最多提取 30 个关键词，以减少向量的维度；基于 TF-IDF 加权的向量空间模型根据文档中每个词的 TFIDF 值对基础的向量空间模型进行加权，此处的文档也已经进行了关键词提取；基于 Word2Vec 的文档表示是利用基于大语料库预训练的词向量进行平均，得到文档的向量表示，这是一种易于实现且常见的文档表示方法。

聚类算法

得到文档的向量表示后，有很多种可供选择的聚类算法。聚类方法主要分为划分法、层次法、基于密度的方法，基于网格的方法，基于模型的方法。同时，从主题模型的角度考虑，也有 LSA、pLSA、NMF、LDA 等一系列方法。

本次实验主要利用 KMeans 聚类算法和 LDA 算法进行文本主题聚类，同时有设计层次聚类、谱聚类等算法。处于训练时间与迭代速度的考虑本次实验没有考虑基于深度学习的模型。一些 SOTA 的模型如 Bayesian SMM 由于过于复杂暂时也没有考虑。

KMeans 算法是一种常见的聚类算法，这里不再详细描述，只不过需要注意的是 Kmeans 的假设是同一类别的文本服从的是高斯分布。另外，由于高维空间的维度灾难问题，Kmeans 算法的时间复杂度非常高。

LDA（Latent Dirichlet Allocation）是一种典型的词袋模型，即它认为一篇文档是由一组词构成的一个集合，词与词之间没有顺序以及先后的关系。一篇文档可以包含多个主题，文档中每一个词都由其中的一个主题生成。它可以将文档集中每篇文档的主题按照概率分布的形式给出；同时是一种无监督学习算法，在训练时不需要手工标注的训练集，需要的仅仅是文档集以及指定主题的数量 k 即可；此外 LDA 的另一个优点则是，对于每一个主题均可找出一些词语来描述它。对算法具体的描述可以参考李航老师的《统计学习方法（第二版）》。

需要注意的是，不同的聚类算法往往和不同的文本特征表示相适应。如考虑到时间复杂度，KMeans 算法更适合低维的向量表示，因此与向量空间模型相比词向量模型或许更适合 KMeans 算法；考虑到 LDA 模型需要利用文档中的词频信息，所以基于词频加权的向量空间模型更适合 LDA 算法。

结果评价方法

结果评价一直是无监督学习的一个难点。聚类算法常见的评价方法可以分为外部指标和内部指标。外部指标是利用已经有的标签对聚类效果进行评价，而内部指标主要是考虑聚类的簇内相似度与簇间相似度。常见的外部指标有 Jaccard 系数、FM 指数（FMI）、Rand 指数（RI）等；常见的内部指标有 Davies-Bouldin 指数、Dunn 指数等。

本次实验将主要利用 FMI 指数与改进过的 RI 指数“ARI”对聚类效果进行评价。

实验环境

- Python 3.6.4
- Numpy 1.14.2
- Sklearn 0.21.3

更多包请见代码。

实验过程与实验结果

特征提取

特征提取主要采用的是第二次实验的特征提取方法，这里不再赘述。

基于 Kmeans 的文本聚类

本实验先是利用准备好的基于词向量加权的文档向量表示对训练集数据与测试集数据分别进行聚类（方法 1）。其中训练集数据维度（9804, 300），测试集数据维度（9833, 300）。模型的参数设置为：

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=20, n_init=10, n_jobs=None, precompute_distances='auto',
    random_state=123, tol=0.0001, verbose=0)
```

为了查看聚类效果，本人对聚类结果进行了降维可视化（如下图），证明聚类还是按照预期的方式在进行。

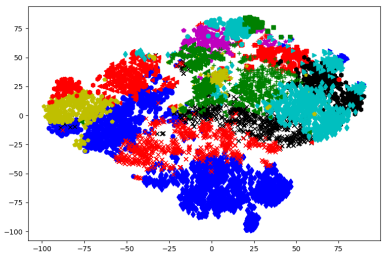

Figure 1: Cluster Result on Training DataSet

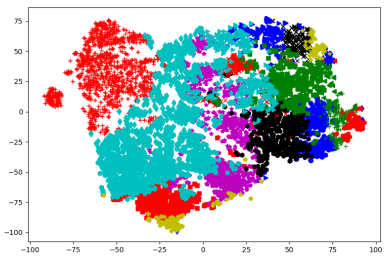

Figure 2: Cluster Result on Test DataSet

之后利用所有的数据进行聚类，得到模型后再分别对训练集与测试集进行聚类（方法 2），实验后发现效果有部分提升，但是仍然很低。

之后尝试利用基于词频加权的向量空间模型进行 KMeans 聚类（方法 3），发现 FMI 值有显著提高，但是 ARI 值却显著降低。

之后对模型进行了多次调参，如加大初始中心的次数、加大 KMeans 的迭代次数、对向量进行标准化、采用 TF-IDF 加权的词向量进行文本表征等操作，实验表明并没有显著效果。

基于 LDA 的文本聚类

然后尝试利用 LDA 模型对文本进行话题分布估计，并使用 argmax() 操作得到概率最大的主题类别作为预测的类别（方法 4），实验结果表明 FMI 值与 ARI 值都有显著提升。

实验结果

最终得到的实验结果如下表：

	FMI	ARI
训练集		
词向量 _KMeans（方法 1）	0.205289	0.114529
词向量 _KMeans（方法 2）	0.208372	0.116476
Count2Vec_KMeans（方法 3）	0.283095	0.044891
Count2Vec_LDA（方法 4）	0.311782*	0.237095*
测试集		
词向量 _KMeans（方法 1）	0.200622	0.108163
词向量 _KMeans（方法 2）	0.203421	0.111379
Count2Vec_KMeans（方法 3）	0.262587	0.040055
Count2Vec_LDA（方法 4）	0.308959*	0.234119*

总结与思考

在第二次实验提取了文本的特征表示后，本次实验相对简单，但是实验结果却一般。在分类任务中，基于 Word2Vec 的模型要显著好于基于词频加权的模型，但是在聚类任务中却比基于词频加权的 LDA 模型低 10%。主要原因个人认为是虽然词向量能够表示一定的语义，但是在高维空间中各个主题的文档并不是单独的高斯分布，而是交叉在一起，在没有其他变化层的作用下，单纯使用 KMeans 并不适合做文本聚类，而传统的 LDA 模型则相对较好。

最终的实验结果仍然一般，个人认为更多的特征需要加入到向量中，也可以采用集成学习或者深度学习等方法。不过需要注意的是，要考虑文档数据的不均衡性会不会影响模型效果。