程序设计文档

1 系统结构图

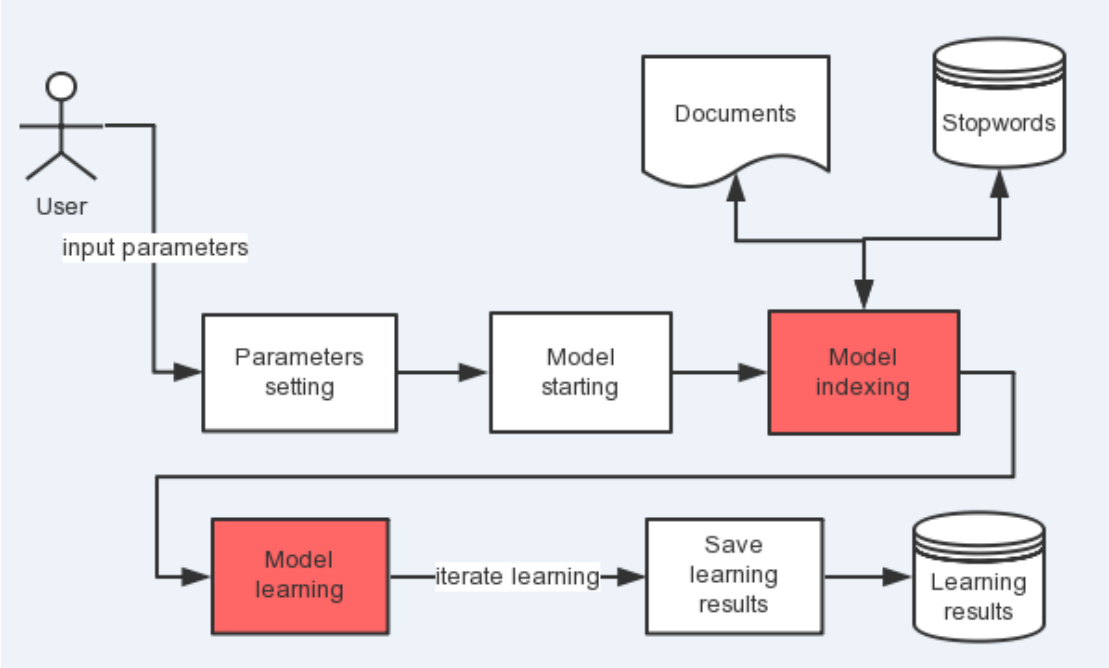


图 1 系统结构图

系统整体设计如上方所示，

1 首先用户输入模型相关参数，用户点击开始，模型开始工作。

2 进入模型索引化模块（重要），即将文档集中的所有文档的词（除去停用词）都索引化，索引化指的是将文档中的词汇和字典中的词关联起来（可以说成为映射）。

3 模型学习（重要）

4 保存学习结果，即 ①文档-主题分布，② 主题-词汇分布，③ 主题分配给词的情况，和 ④ **主题下的 top 词汇及概率**。第 4 个结果是焦点。

2 功能模块结构图

Model indexing 模块和 Model learning 模块比较重要，接下来会详细说明。

2.1 Model indexing 模块

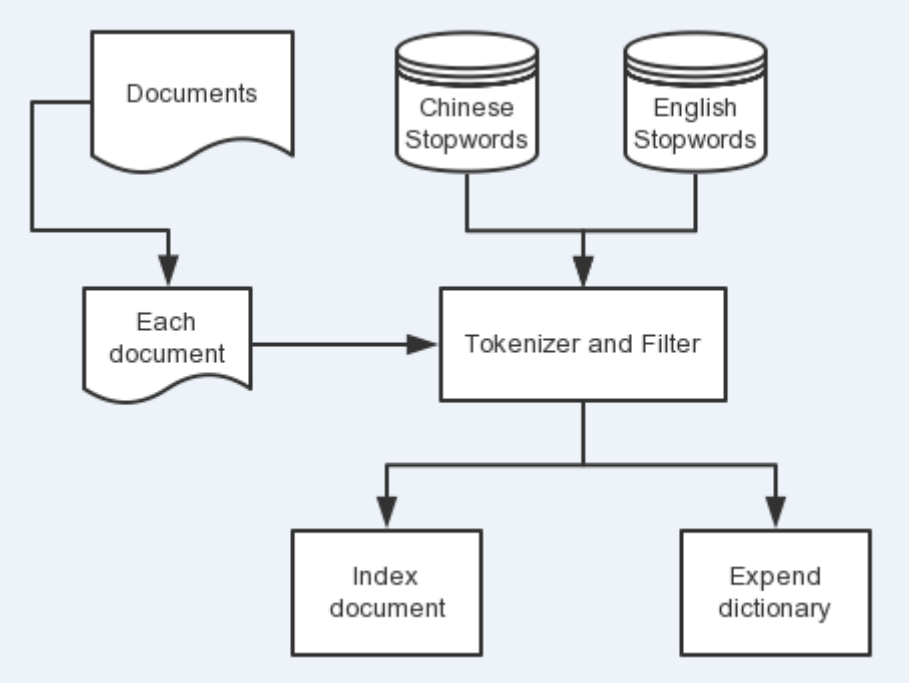
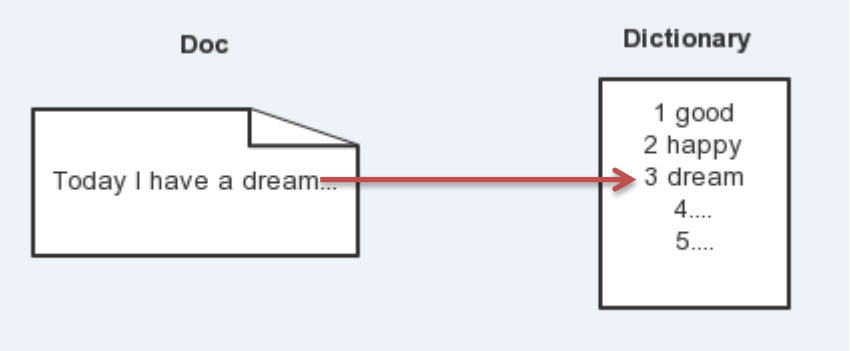


图 2　Model indexing 模块结构图

作用：将每篇文档的索引化，同时去噪和扩充字典。值得注意的是字典不是一开始就存在的，它是随着读取文档的数量而不断扩充的。



例如上图所示，docword[4]=3。这样的形式将文档中的词和字典索引联系起来。

2.2 Model learning 模块

Model learning 模块是最重要的模块，之前的去除参数和索引化，都是为了构建适合模型学习的数据类型。下面是该模块的结构图：

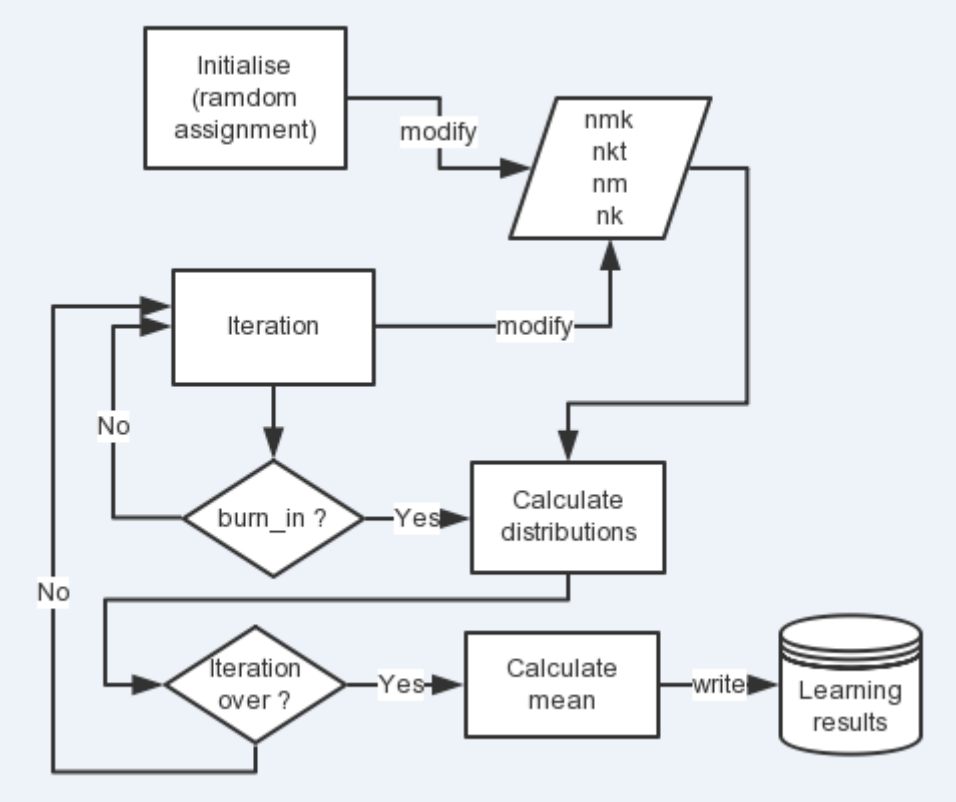


图 3　Model learning 模块结构图

Model learning 采用 Gibbs sampling 算法。这里使用两个略微不同的 Gibbs 采样方程式。其中方程式 1 来源于论文：Parameter estimation for text analysis。方程 2 来源于论文：Finding scientific topics。两个方程式都是给定词汇下，主题的 full conditional distribution，本质是一样的。考虑到方程式对结果的影响，所以使用了两种方程。

下面简单对算法进行说明，

① 首先初始化，即对文档中的每一个词汇随机分配一个主题，相应的计数变量（**nmk**：第 m 篇文档中第 k 个主题分配的次数；**nkt**：词项 t 被分配为主题 k 的次数；**nm**：文档 m 的主题分配次数和；**nk**：主题 k 被分配的总次数）加 1。

例如第 2 篇文档的第 3 个词汇被分配到主题 4，假设该词汇在字典中的索引是 5.

则有 nmk[2-1][4-1]++; nkt[4-1][5]++; nm[2-1]++; nk[4-1]++;

② 给文档中的每一个词汇重新分配主题，分配主题的计算公式来源于上述论文。然后对计数变量修改。一直重复此步骤，至超过 burn-in 时期。这里说明一下 Gibbs 采样，这是一种用于近似后验分布的算法，该算法就是不停的采样，采样的方式是固定其他维度不动，采样当前维度的值。这样，样本每次只是变动一个维度的值。当迭代至 burn-in 时期后，采样出来的样本就服从后验分布了。

$$p(z_i \mid \vec{z}_{-i}, \vec{w}) \xrightarrow{\text{Iterate}} p(\vec{z} \mid \vec{w})$$

③ 保存学习结果，因为 Gibbs 采样出来的样本具有条件依赖性，即样本之间关联性很大。为了减少样本之间的影响，使样本彼此独立。所以决定在 burn-in 期后，每 savestep 轮保存一次分布结果，当达到最终的迭代时，将前几次保存的结果求平均，作为最终的分布结果保存至文件中。