

主题模型LDA及其在微博推荐&广告算法中的应用-第1期

数据挖掘与机器学习 24 十一, 2014 10



@吴宇
WB

【前言】本篇文章中所涉及的大部分理论知识，都是由微博的推荐算法和广告算法团队共同收集，共同学习的，而现在这两个团队也合并成为一个更大的--推荐&广告算法团队。相信在未来我们会一起将推荐&广告算法做的更好，也欢迎更多机器学习狂热分子加入我们。

这个系列文章主要分为5章：

第1章介绍本文中涉及的基础算法和基础工具 包括EM 变分推

🔍 To search type and hit enter

近期文章

🕒 微博推荐数据服务代理: hyper_proxy的设计和实现

🕒 微博推荐计算层解决方案: lab_common_so框架

🕒 行走在网格之间：微博用户关系模型

🕒 千人千面：微博用户画像

🕒 认识每一个“你”：微博中的用户模型

近期评论

🗨 Domain Name 发表在《在线最优化求解(Online Optimization)之二：截断梯度法(TG)》

🗨 porn 发表在《在线最优化求解(Online Optimization)之三：FOBOS》

第2章介绍LDA相关的理论知识，包括LDA的基本思想，Inference过程和参数估计，以及速度更快的Online VBEM算法，最后详细介绍了如何用Gibbs Sampling来训练LDA。

第3章将解读ICML 2014的best paper，给我们的启示以及我们利用微博短文本做的相关实验，证明了Gibbs Sampling的效果要好于Online VBEM算法。

第4章重点介绍LDA目前在微博的内容推荐，用户推荐以及广告算法中的应用。

第5章介绍我了解的一些关于LDA在工业界和学术界的进展。

附：由于文章较长，我会分两期分享，第1期只有前3章的内容，第2期分享第4章和第5章。水平有限，如有错误，请及时指正。

目录

1. 一些基础算法和工具

1.1 EM算法

1.2 Variance Inference

分类目录

📁 产品

📁 引擎架构

📁 数据挖掘与机器学习

📁 未分类

📁 算法与策略

文章归档

📅 2016 年一月

📅 2015 年十一月

📅 2015 年十月

📅 2015 年四月

📅 2015 年二月

📅 2014 年十二月

📅 2014 年十一月

📅 2014 年八月

📅 2014 年七月

2.LDA理论知识

2.1 Inference

2.2 Parameter estimation 参
数估计

2.3 平滑

2.4 Online VBEM算法

2.5 Gibbs Sampling训练过程

3.利用微博短文本做的相关实验

3.1 Dirichlet分布的物理解释

3.2 ICML 2014 Best Paper解
读

3.3 VBEM算法和Gibbs
Sampling的实验结果对比

1. 一些基础算法和工具

LDA全称为Latent Dirichlet
Allocation，在2003年由David M.

inference, Gibbs Sampling, 并介绍我们用来训练LDA的两个工具。

1.1 EM算法

EM全称为Expectation Maximization，即期望最大化。用于含隐变量的极大似然估计。现定义符号如下：

观察到的数据集记为 X

隐变量记为 Z

待估计的参数记为 Θ

并令： $Y=(X,Z)$

在给定 X 的情况下，估计 Θ 的方法通常为极大似然估计，简称MLE：

$$\Theta = \arg \max_{\Theta} L(X | \Theta)$$

但并不是每次都有解析解。当求解困难时就可以引入隐变量，利用EM算法来迭代求解，近似MLE过程。具体过程如下：

E步： ψ

$$Q(\Theta^{t+1} | \Theta^t) = E[\ln P(Y | \Theta^{t+1}) | \Theta^t, X]$$

M步： ψ

$$\Theta^{t+1} = \arg \max_{\Theta^{t+1}} Q(\Theta^{t+1} | \Theta^t) \quad \psi$$

并利用Jensen's inequality:

$\log(E[X]) \geq E[\log(X)]$ 。我们可以得到

【3】：

$$\begin{aligned}
 & \log P(X / \Theta^{t+1}) \\
 &= \log \int_Z P(X, Z / \Theta^{t+1}) dZ \\
 &= \log \int_Z P(X, Z / \Theta^{t+1}) \frac{P(Z / X, \Theta^t)}{P(Z / X, \Theta^t)} dZ \\
 &= \log E_{Z/X, \Theta^t} \left[\frac{P(X, Z / \Theta^{t+1})}{P(Z / X, \Theta^t)} \right] \\
 &\geq E_{Z/X, \Theta^t} \left[\log \left(\frac{P(X, Z / \Theta^{t+1})}{P(Z / X, \Theta^t)} \right) \right] = E_{Z/X, \Theta^t} [\log(P(X, Z / \Theta^{t+1})) - \log(P(Z / X, \Theta^t))] \\
 &= E_{Z/X, \Theta^t} [\log(P(X, Z / \Theta^{t+1}))] - E_{Z/X, \Theta^t} [\log(P(Z / X, \Theta^t))] \\
 &= Q(\Theta^{t+1} | \Theta^t) - E_{Z/X, \Theta^t} [\log(P(Z, X / \Theta^t) - \log(P(X / \Theta^t))] \\
 &= Q(\Theta^{t+1} | \Theta^t) - Q(\Theta^t | \Theta^t) + \log(P(X / \Theta^t))
 \end{aligned}$$

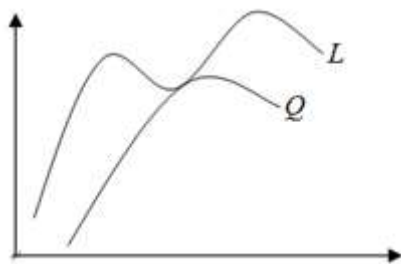
即：

$$\log P(X / \Theta^{t+1}) - \log(P(X / \Theta^t)) \geq Q(\Theta^{t+1} | \Theta^t) - Q(\Theta^t | \Theta^t)$$

带入 $L(X|\Theta) = \log P(X|\Theta)$ ，可知：

$$L(X / \Theta^{t+1}) - L(X / \Theta^t) \geq Q(\Theta^{t+1} | \Theta^t) - Q(\Theta^t | \Theta^t).$$

也就是当我们step-by-step的增大 Q 更新 Θ ，实际上是在增大 L 的下界，从而实现了近似MLE的过程。如下图所示【2】：



可以看到EM算法只能得到一个局部的最优解。关于EM算法的进一步理解可以参考【2】的第9章 我也写过

分布。这里主要参考【6】，来解释一下变分推断的原理。除了在1.1节中定义的数学符号外，再定义分布 p 和分布 q ，两个分布之间的距离用 $KL(q||p)$ 表示，根据KL距离的定义可知：

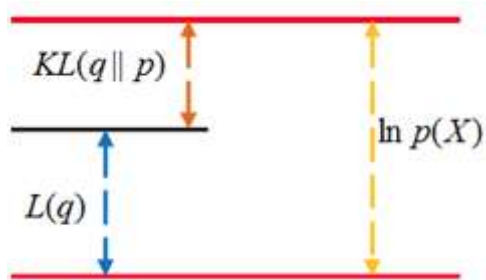
$$\ln p(X) = L(q) + KL(q||p)$$

其中：

$$L(q) = \int q(Z) \ln \frac{p(X, Z)}{q(Z)} dZ$$

$$KL(q||p) = - \int q(Z) \ln \frac{p(Z|X)}{q(Z)} dZ$$

为了让观察到的数据集更好的拟合分布 p ，如果最大化 $L(q)$ ，就会最小化 $KL(q||p)$ （恒非负），其实就是令 $q(Z)$ 最接近于 $p(Z|X)$ ，从而实现了用分布 q 来近似隐变量后验概率分布的过程。如下图所示：



为了简化计算，假设变量之间互相独立，即：

$$q(Z) = \prod_{i=1}^M q_i(Z_i)$$

$$\begin{aligned}
\mathcal{L}(q) &= \int q_j \left\{ \int \ln p(X, Z) \prod_{i \neq j} q_i dZ_i \right\} dZ_j - \int q_j \ln q_j dZ_j + \text{const} \\
&= \int q_j \ln \tilde{p}(X, Z_j) dZ_j - \int q_j \ln q_j dZ_j + \text{const} \\
&= \int q_j \frac{\tilde{p}(X, Z_j)}{q_j} dZ_j + \text{const} \\
&= -KL(q_j || \tilde{p}(X, Z_j)) + \text{const}
\end{aligned}$$

其中：

$$\begin{aligned}
\ln \tilde{p}(X, Z_j) &= \int \ln p(X, Z) \prod_{i \neq j} q_i dZ_i \\
&= E_{i \neq j}[\ln p(X, Z)]
\end{aligned}$$

很显然令 $L(q)$ 最大的方式，即
令：

$$q_j = \tilde{p}(X, Z)$$

做一个归一化之后可知：

$$q_j^*(Z_j) = \frac{\exp(E_{i \neq j}[\ln p(X, Z)])}{\int \exp(E_{i \neq j}[\ln p(X, Z)]) dZ_j}$$

然后不断的迭代优化，直到收敛。

如果觉得公式理解有困难，可以把积分看成加和，即将连续变量看成离散变量，应该就比较容易能看懂了。

1.3 Gibbs Sampling

这个部分我主要参考的是腾讯的大

对于给定的概率分布 p ，希望能生成符合该分布的样本，但有时该分布会因为高维而变得很复杂，并不好生成对应的样本。在1953年，Metropolis利用马氏链能收敛到平稳分布的性质，首次提出了基于Markov Chain Monte Carlo（马尔科夫蒙特卡洛，MCMC）的采样方法，简单的讲就是人为构造了细致平稳条件，假设 p 为分布， q 为转移矩阵， x, y 为样本，具体算法如下：

Algorithm 5 MCMC 采样算法

1: 初始化马氏链初始状态 $X_0 = x_0$

2: 对 $t = 0, 1, 2, \dots$, 循环以下过程进行采样

- 第 t 个时刻马氏链状态为 $X_t = x_t$, 采样 $y \sim q(x|x_t)$
 - 从均匀分布采样 $u \sim Uniform[0, 1]$
 - 如果 $u < \alpha(x_t, y) = p(y)q(x_t|y)$ 则接受转移 $x_t \rightarrow y$, 即 $X_{t+1} = y$
 - 否则不接受转移, 即 $X_{t+1} = x_t$
-

但有时，接收率 α 可能偏小，导致转移的概率变小，采样次数变多。为了增大接收率，就出现了Metropolis-Hastings算法，如下：

Algorithm 6 Metropolis-Hastings 采样算法

1: 初始化马氏链初始状态 $X_0 = x_0$

2: 对 $t = 0, 1, 2, \dots$, 循环以下过程进行采样

- 第 t 个时刻马氏链状态为 $X_t = x_t$, 采样 $y \sim q(x|x_t)$
 - 从均匀分布采样 $u \sim Uniform[0, 1]$
 - 如果 $u < \alpha(x_t, y) = \min \left\{ \frac{p(y)q(x_t|y)}{p(x_t)q(y|x_t)}, 1 \right\}$ 则接受转移 $x_t \rightarrow y$, 即 $X_{t+1} = y$
 - 否则不接受转移, 即 $X_{t+1} = x_t$
-

接收率还能再增大吗？Gibbs sampling 就是接收率为1的MCMC，这

Algorithm 8 n维Gibbs Sampling 算法

- 1: 随机初始化 $\{x_i : i = 1, \dots, n\}$
- 2: 对 $t = 0, 1, 2, \dots$ 循环采样
 1. $x_1^{(t+1)} \sim p(x_1 | x_2^{(t)}, x_3^{(t)}, \dots, x_n^{(t)})$
 2. $x_2^{(t+1)} \sim p(x_2 | x_1^{(t+1)}, x_3^{(t)}, \dots, x_n^{(t)})$
 3. \dots
 4. $x_j^{(t+1)} \sim p(x_j | x_1^{(t+1)}, \dots, x_{j-1}^{(t+1)}, x_{j+1}^{(t)}, \dots, x_n^{(t)})$
 5. \dots
 6. $x_n^{(t+1)} \sim p(x_n | x_1^{(t+1)}, x_2^{(t+1)}, \dots, x_{n-1}^{(t+1)})$

所以gibbs sampling的关键在于求得分布 p 。往往越强大的算法，可能是形式越简单。论文【8】给了Gibbs Sampling最直观的描述，就两步：

1. choose dimension i (random or by permutation)
2. sample x_i from $p(x_i | x_1, x_2, \dots, x_{i-1}, x_{i+1}, \dots, x_n)$

1.4 Vowpal Wabbit

简称为VW，作者为John Langford，是雅虎研发的开源机器学习系统。主要有三个模型：回归模型，矩阵分解模型，LDA模型。而且能兼容hadoop，做并行运算。

前些天在微博上也有一条讨论VW性能的微博，有兴趣的可以在我之前的大牛同事@haibo_mind的微博里面找到相关信息，里面也有他写的使用VW的一些文章：

//@haibo_mind: vw也是一个非常好用的分布式机器学习框架，与hadoop可以无缝结合，特别适合创业公司。//@特级飞行员舒克 //@52nlp: //@尘缘-SYSU: 顺推一个我之前用VW做的广告文本分类的比赛：<http://l.cn/R74LaBn> Python & Shell & VW

@cswthjiang

fastML比较了Vowpal Wabbit, Liblinear/SBM 和 StreamSVM，验证了VW确实是神器。

<http://l.cn/R742J6H>

10月18日 00:12 来自 微博 weibo.com

转发 180 | 评论 3 | 4

源码下

载: https://github.com/JohnLangford/vowpal_wabbit/wiki

VW中的LDA模型采用的是online VBEM算法实现。训练文本格式如下:

```
No labels, no namespace
| word_id:word_ct word_id:word_ct word_id:word_ct
word_id:word_ct ...
| word_id:word_ct word_id:word_ct word_id:word_ct
word_id:word_ct ...
| word_id:word_ct word_id:word_ct word_id:word_ct
word_id:word_ct ...
...
```

需要word的序号, 以及word的词频, 但显然这个词频可以用任意表示word重要性的实数值来代替。

1.5 Gibbs Sampling源码

这份代码是LDA Gibbs Sampling的c++版本实现, 是Yi Wang写的, 下载地址: <http://code.google.com/p/ompi-lda/>

支持单机多线程, 也支持MPI。这份源码实现分布式的思想, 可以参考Newman的论文【9】

训练数据的格式如下:

```
38 1 126 1 127 1 128 1 129 1 130 1 131 1
141 1 142 1 143 1 144 1 145 1 146 1 147 1
159 1 140 1 160 1 161 1 162 1
163 1 164 1 165 1 166 1 167 1 168 1 169 1
182 1 183 1 184 1 185 1 186 1 140 1 162 1
```

倍，所以这个值不宜过大。在我们的实验中，取1的效果就不错了。

搭建MPI也非常简单。找到几台可以互相ssh的机器，根据【10】中的步骤就能搭建好了。

1.6 Point-Wise Mutual Information

简称为PMI，是用来判断LDA模型训练效果的指标。当我们训练完LDA模型时，会得到主题下词的分布，如下图(歌手的主题)：



The image shows a terminal window displaying the word distribution for 'topic 25'. The words are listed on the left, and their corresponding probabilities are on the right, separated by three dashes. The words '歌手' (Singer) and '我是歌手' (I am a Singer) are highlighted in yellow in the original image.

topic 25:		
歌手	---	0.0398
张杰	---	0.0298
周笔畅	---	0.0227
邓紫棋	---	0.0137
韩磊	---	0.0096
杨宗纬	---	0.0063
曹格	---	0.0060
爱的供养	---	0.0058
歌王	---	0.0057
总决赛	---	0.0056
我是歌手	---	0.0050
雯婕	---	0.0044
供养	---	0.0042
羽泉	---	0.0041
张宇	---	0.0038
歌单	---	0.0035
林志炫	---	0.0033
罗琦	---	0.0024
韦唯	---	0.0023
尚雯婕	---	0.0022
谭维维	---	0.0022
彭佳慧	---	0.0021
品冠	---	0.0020
黄绮珊	---	0.0020

$$\text{PMI-Score}(\mathbf{w}) = \frac{1}{45} \sum_{i < j} \text{PMI}(w_i, w_j), i, j \in \{1 \dots 10\}$$

$$\text{where } \text{PMI}(w_i, w_j) = \log \frac{P(w_i, w_j)}{P(w_i)P(w_j)},$$

其中： $P(w_i, w_j)$ 为词对 (w_i, w_j) 在测试文本中出现的概率， $P(w_i)$ 为 w_i 在文本中出现的概率。当然在计算PMI-score时也可不取均值，而采用中值。

2.LDA理论知识

先定义几个记号：

一篇文档由N个词组成： $w=(w_1, w_2, \dots, w_N)$

一个语料由M篇文档构成： $D=\{w_1, w_2, \dots, w_N\}$

词表的大小： V

LDA是给文档建模的一种方法，属于生成模型，它的基本思想是认为一篇文档是由很多隐藏的主题构成的。它假设一篇文档是按如下方式产生的：

step 1: 选择词的个数N，其中N由一个泊松分布产生

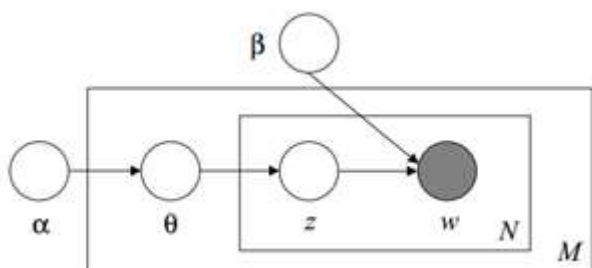
从参数为 狄里克雷分

(a) 选择一个topic z_n 服从参数为 θ 的多项式分布，满足

$$z_n \sim \text{Multinomial}(\theta)$$

(b) 从分布 $p(\omega_n | z_n, \beta)$ 选择词 ω_n

其中 β 是一个 $k \times V$ 的矩阵， k 为指定的topic个数，行是主题，列是词。从一个图模型的角度来看更直观：



于是，可以用如下的公式来表示生成一篇文档的概率：

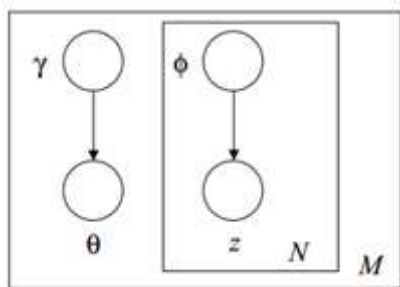
$$p(D|\alpha, \beta) = \prod_{d=1}^M \int p(\theta_d | \alpha) \left(\prod_{n=1}^{N_d} \sum_{z_{dn}} p(z_{dn} | \theta_d) p(w_{dn} | z_{dn}, \beta) \right) d\theta_d$$

2.1 Inference

当一个新的文本输入到主题模型LDA中，我们希望得到这个文本的隐主题分布，这个过程就称为Inference。用如下公式来表示，其中 w 代表新输入的文本：

$$p(\theta, z | w, \alpha, \beta) = \frac{p(\theta, z, w | \alpha, \beta)}{p(w | \alpha, \beta)}$$

中 γ 是Dirichlet的参数。而为了简化计算，将 θ 和 z 独立开来，如下图所示：



那为什么 θ 会服从Dirichlet分布呢？有两种解释方法：

第一种解释：根据1.2节变分推断中最后的结论可知，只要满足 $\ln q(\theta) = E_z[p(\theta, z | w, \alpha, \beta)] + \text{const}$ 即可用分布 q 去近似 $p(\theta, z | w, \alpha, \beta)$ ，这个式子展开可以得出 θ 会服从Dirichlet分布。

第二种解释：Dirichlet分布和多项式分布是共轭先验的，所以如果 θ 的先验分为Dirichlet分布，那么后验也是Dirichlet分布。这也是为什么假设先验为Dirichlet分布的原因，可以简化模型。

接下来根据变分推断的原理，通过最小化两个分布的距离，求得 γ, ϕ ：

$$(\gamma^*, \phi^*) = \underset{(\gamma, \phi)}{\operatorname{argmin}} D(q(\theta, z | \gamma, \phi) \| p(\theta, z | w, \alpha, \beta))$$

其中 $D()$ 表示KL距离。

$$\begin{aligned}\phi_{ni} &\propto \beta_{i w_n} \exp\{E_q[\log(\theta_i) | \gamma]\} \\ \gamma_i &= \alpha_i + \sum_{n=1}^N \phi_{ni}.\end{aligned}$$

不断的迭代就能求出 γ 了。完整的公式推导可参照【1】的附录部分。

2.2 Parameter estimation 参数估计

到目前为止就涉及到2个参数 α , β , 其它都是隐变量。将每个文档当作一个样本，似然函数显而易见：

$$\ell(\alpha, \beta) = \sum_{d=1}^M \log p(\mathbf{w}_d | \alpha, \beta)$$

在这里隐变量为 θ, z , 令 $\Theta = (\alpha, \beta)$, 很容易写出E步和M步：

E 步： ψ

$$Q(\Theta^{t+1} | \Theta^t) = E[\ln P(\theta, z, D | \Theta^{t+1}) | \Theta^t, D].$$

M 步： ψ

$$\Theta^{t+1} = \arg \max_{\Theta^{t+1}} Q(\Theta^{t+1} | \Theta^t).$$

但是在E步的计算过程中，需要使用2.1节中提到的variation inference来计算 γ, ϕ 。所以整个算法又称为VBEM算法。

对 β 求导等于0可知：

$$\beta_{ij} \propto \sum_{d=1}^M \sum_{n=1}^{N_d} \phi_{dni} w_{dn}^j$$

同理，对于 α 则有：

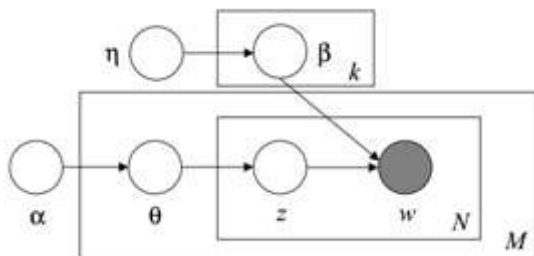
$$L_{[\alpha]} = \sum_{d=1}^M \left(\log \Gamma \left(\sum_{j=1}^k \alpha_j \right) - \sum_{j=1}^k \log \Gamma(\alpha_j) + \sum_{i=1}^k ((\alpha_i - 1) (\Psi(\gamma_{di}) - \Psi(\sum_{j=1}^k \gamma_{dj}))) \right)$$

求得 α 的迭代式为：

$$\alpha_{\text{new}} = \alpha_{\text{old}} - H(\alpha_{\text{old}})^{-1} g(\alpha_{\text{old}})$$

2.3平滑

我们之前提到过 β 是一个矩阵，它默认所有语料的词都在这个矩阵里面。但很显然这是不可能做到的，一旦出现不存在于矩阵中的词，在计算时就会出现概率为0的情况，所以我们要做平滑。在论文【1】中的做法是将 β 也作为隐变量，用另一个Dirichlet分布来生成这个矩阵，也就是词表中词是以一定的概率出现的，对于未出现的词也是存在一个先验概率的。如下图所示：



那在参数估计的时候，就变成了估计 λ, α 。仍旧是根据VBEM算法，可知：

$$\lambda_{ij} = \eta + \sum_{d=1}^M \sum_{n=1}^{N_d} \phi_{dni}^* w_{dn}^j$$

2.4 Online VBEM算法

对于VBEM来估计参数，显然是一个batch的过程。对于文档个数很多时，时间复杂度会很大，在论文【11】中提出了一种online的思路：

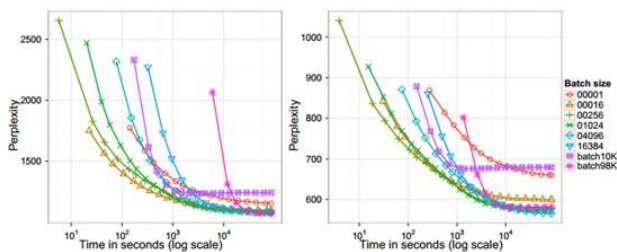
Algorithm 2 Online variational Bayes for LDA

```

Define  $\rho_t \triangleq (\tau_0 + t)^{-\kappa}$ 
Initialize  $\lambda$  randomly.
for  $t = 0$  to  $\infty$  do
  E step:
  Initialize  $\gamma_{tk} = 1$ . (The constant 1 is arbitrary.)
  repeat
    Set  $\phi_{twk} \propto \exp\{\mathbb{E}_q[\log \theta_{tk}] + \mathbb{E}_q[\log \beta_{kw}]\}$ 
    Set  $\gamma_{tk} = \alpha + \sum_w \phi_{twk} n_{tw}$ 
  until  $\frac{1}{K} \sum_k |\text{change in } \gamma_{tk}| < 0.00001$ 
  M step:
  Compute  $\tilde{\lambda}_{kw} = \eta + D n_{tw} \phi_{twk}$ 
  Set  $\lambda = (1 - \rho_t) \lambda + \rho_t \tilde{\lambda}$ 
end for

```

论文【11】在Nature（100,000条）和Wikipedia（352,549条）上都做了关于online 和batch的实验：



在VW工具包里面online和batch的实现都有，在VW训练命令行--minibatch选项就是用来调节batch的个数。

2.5 Gibbs Sampling训练过程

前面几节主要介绍了LDA模型用VBEM训练的过程，但当前的主流更偏向Gibbs Sampling，本节主要介绍如何用Gibbs Sampling来训练出主题模型的【8】。

其实主题模型就是一个矩阵，主题和词的矩阵，即本章开头提到的 β 。文档和主题的矩阵是附带出来的。

step1:

□ initialisation

zero all count variables, $n_m^{(k)}, n_m, n_k^{(t)}, n_k$

for all documents $m \in [1, M]$ **do**

for all words $n \in [1, N_m]$ in document m **do**

 sample topic index $z_{m,n}=k \sim \text{Mult}(1/K)$

 increment document–topic count: $n_m^{(k)} + 1$

 increment document–topic sum: $n_m + 1$

 increment topic–term count: $n_k^{(t)} + 1$

 increment topic–term sum: $n_k + 1$

end for

end for

直接读伪代码，很容易理解，这一步就是给每个词随机一个主题。文档--主题矩阵和主题 词矩阵对应处加1

```

□ Gibbs sampling over burn-in period and sampling period
while not finished do
  for all documents  $m \in [1, M]$  do
    for all words  $n \in [1, N_m]$  in document  $m$  do
      □ for the current assignment of  $k$  to a term  $t$  for word  $w_{m,n}$ :
        decrement counts and sums:  $n_m^{(k)} - 1; n_m - 1; n_k^{(t)} - 1; n_k - 1$ 
      □ multinomial sampling acc. to Eq. 79 (decrements from previous step):
        sample topic index  $\bar{k} \sim p(z_i | \vec{z}_{-i}, \vec{w})$ 
      □ use the new assignment of  $z_{m,n}$  to the term  $t$  for word  $w_{m,n}$  to:
        increment counts and sums:  $n_m^{(\bar{k})} + 1; n_m + 1; n_k^{(t)} + 1; n_k + 1$ 
    end for
  end for
  □ check convergence and read out parameters
  if converged and  $L$  sampling iterations since last read out then
    □ the different parameters read outs are averaged.
    read out parameter set  $\underline{\Phi}$  according to Eq. 82
    read out parameter set  $\underline{\Theta}$  according to Eq. 83
  end if
end while

```

最核心的步骤：对每个词用Gibbs Sampling采样出一个主题，并且更新文档--主题矩阵和主题--词矩阵。直到参数收敛为止。

之前在1.3节中已经提到过，Gibbs Sampling的核心就在于分布的计算，在主题模型LDA中就是要计算下式：

$$\begin{aligned}
 p(z_i=k | \vec{z}_{-i}, \vec{w}) &= \frac{p(\vec{w}, \vec{z})}{p(\vec{w}, \vec{z}_{-i})} = \frac{p(\vec{w} | \vec{z})}{p(\vec{w}_{-i} | \vec{z}_{-i}) p(w_i)} \cdot \frac{p(\vec{z})}{p(z_{-i})} \\
 &\propto \frac{\Delta(\vec{n}_z + \vec{\beta})}{\Delta(\vec{n}_{z,-i} + \vec{\beta})} \cdot \frac{\Delta(\vec{n}_m + \vec{\alpha})}{\Delta(\vec{n}_{m,-i} + \vec{\alpha})} \\
 &\propto \frac{\Gamma(n_k^{(t)} + \beta_t) \Gamma(\sum_{i=1}^V n_{k,-i}^{(t)} + \beta_t)}{\Gamma(n_{k,-i}^{(t)} + \beta_t) \Gamma(\sum_{i=1}^V n_k^{(t)} + \beta_t)} \cdot \frac{\Gamma(n_m^{(k)} + \alpha_k) \Gamma(\sum_{i=1}^K n_{m,-i}^{(k)} + \alpha_k)}{\Gamma(n_{m,-i}^{(k)} + \alpha_k) \Gamma(\sum_{i=1}^K n_m^{(k)} + \alpha_k)} \\
 &\propto \frac{n_{k,-i}^{(t)} + \beta_t}{\sum_{i=1}^V n_{k,-i}^{(t)} + \beta_t} \cdot \frac{n_{m,-i}^{(k)} + \alpha_k}{[\sum_{i=1}^K n_m^{(k)} + \alpha_k] - 1}
 \end{aligned}$$

采得主题的样本。

例如：有3个主题，根据上式计算出词在三个主题下的概率分别为（0.2,0.3,0.5），那就分成3个区间（0，0.2）（0.2,0.5），（0.5,1），然后随机个服从（0,1）均匀分布的变量，假设为0.75，落在第3个区间，就给这个词映射到第3个主题。

3.利用微博短文本做的 相关实验

本章主要介绍用微博短文本来训练LDA模型的一些实验。我选用了1000W条左右的优质微博，使用VW训练工具来训练LDA模型。训练格式如1.4节所述，下面我来展示一下我们逐渐优化结果的过程。

第一个版本，我只使用了TF，主题下的词分布如下：

歌手	---	0.0676
家庭	---	0.0461
张杰	---	0.0298
年龄	---	0.0281
组合	---	0.0278
妻子	---	0.0274
理想	---	0.0266
开车	---	0.0256
周笔畅	---	0.0196
收入	---	0.0183
丈夫	---	0.0178
稳定	---	0.0158
夫妻	---	0.0146
状况	---	0.0139
素质	---	0.0109
叶子	---	0.0106
林俊杰	---	0.0105
身体健康	---	0.0104
下车	---	0.0096
途中	---	0.0084
韩磊	---	0.0077
单车	---	0.0076

上图的效果并不十分理想，然后我用TF-IDF再加归一化，效果就好很多了，《我是歌手》相关的人都聚出来了：

歌手	---	0.0398
张杰	---	0.0298
周笔畅	---	0.0227
邓紫棋	---	0.0137
韩磊	---	0.0096
杨宗纬	---	0.0063
曹格	---	0.0060
爱的供养	---	0.0058
歌王	---	0.0057
总决赛	---	0.0056
我是歌手	---	0.0050
姜健	---	0.0044
供养	---	0.0042
羽泉	---	0.0041
张宇	---	0.0038
歌单	---	0.0035
林志炫	---	0.0033
罗琦	---	0.0024
韦唯	---	0.0023
尚雯婕	---	0.0022
谭维维	---	0.0022
彭佳慧	---	0.0021
品冠	---	0.0020
黄绮珊	---	0.0020
李小琳	---	0.0017
王之战	---	0.0017
巅峰	---	0.0017
立白	---	0.0016
彩排	---	0.0015
他不懂	---	0.0013
动力火车	---	0.0013

3.1 Dirichlet分布的物理解释

一枚均匀的硬币抛出第一次正面需要抛的次数符合几何分布。独立重复实验中成功的次数符合二项式分布。那如何解释狄里克雷分布呢？

假设一个盒子里最初有K种颜色的球各 α 个，每一次，从盒中随机取出一个球，把球放回，并且再向盒子中放进一个同样颜色的球。进行同样的操作N次，当N趋于无穷时，盒子中不同颜色

所以 α 越大，分布越均匀。 α 越小，分布越不均匀。

3.2 ICML 2014 Best Paper解读

ICML 是 International Conference on Machine Learning的缩写，即国际机器学习大会。是机器学习的顶级会议。2014年在北京举办，Best Paper给了PKU的Phd, Jian Tang【13】。下面是相关的微博：



这篇论文非常有价值的部分在于它用实验揭示了语料大小 D ，文档词的长度 N ，topic大小 K ， α 和 β 是如来影响LDA训练效果的，它采用了PMI来衡量模型好坏。

我选用了他在twitter（与微博的语料会比较接近）上的实验结果，来分析各个参数的影响。下图：

不会一直增大。所以文档长度不能太短，但也不宜过长。

启示2：训练语料的大小不能太小。

启示3： α 和 β 值的大小，会决定文本在主题上分布的集中程度，词在主题中的分布集中程度。最终对LDA模型训练的效果造成非常大的影响。

3.3 VBEM算法和Gibbs Sampling

的实验结果对比

实验工具：VBEM算法由1.4节所述的VW来实现。Gibbs Sampling算法由1.5节Yi Wang大神开发的工具实现。

我在本章开头提到的微博语料基础上，变换 α 和 β 值，得到了如下的一组结果：

α, β	主题个数	PMI Score
0.01, 0.005	150	0.54
0.1, 0.1	150	0.43
0.5, 0.5	150	0.71

下：

α, β	主题个数	PMI Score
0.01, 0.005	150	1.17
0.1, 0.1	150	1.19
0.5, 0.5	150	1.17

对比于以上两个表，我们能得到两个结论：

1. 对于VBEM算法， α 和 β 值初始值的设置对于模型最后的效果影响较大。而对于Gibbs Sampling算法， α 和 β 值初始值对于模型的效果影响不大。

2. Gibbs Sampling的PMI-score要远高于VBEM算法。

但VBEM算法有个最大的优点就是训练速度超快，如果不是对模型效果精益求精的项目，用VW实现性价比还是很高的。

(未完待续，本期只分享前三章，第4章和第5章将在下期分享)

[1] David M. Blei , Andrew Y. Ng and Michael I. Jordan. Latent Dirichlet Allocation. 2003

[2] 李航 . 统计学习方法

[3] Maya R. Gupta and Yihua Chen. Theory and Use of the EM Algorithm. 2011

[4]
http://blog.sina.com.cn/s/blog_62508b050101fey7.html

[5]
http://blog.sina.com.cn/s/blog_62508b050101fuhy.html

[6] Pattern Recognition and Machine Learning

[7] @Rickjin LDA 数学八卦

[8] Parameter estimation for text analysis

[9] Scalable Parallel Topic Models

[10]
<http://wuyananzan60688.blog.163.com/blog/static/12777616320132452036223/>

[11] Online Learning for Latent Dirichlet Allocation

[12] David Newman. Improving Topic
C h i h R l i d T i


Tags: 主题模型 LDA 机器学习



10 RESPONSES

 **Comments** 10  **Pingbacks** 0



spyhunter  2015 年 11 月 2 日 上午 2:22

Glad to see your web. thanks admin.

回复



volets battants pvc  2015 年 11 月 6 日 上午 11:05

What's up, this weekend is good in support of me, because this moment i am reading this great educational post here at my house.

回复



匿名  2015 年 11 月 18 日 下午 6:25

TF-IDF加归一化，其中的归一化指的是什么？将TF-IDF投射到0-1？

回复



小鹿  2016 年 3 月 16 日 下午 9:53

这里的TF，和TF-IDF，然后主题下发生变化，是怎么回事，求解答。

回复



วิถีเลือกขนตาปลอม  2015 年 11 月 20 日 上午 8:42

WOW just what I was searching for. Came here by searching for ขนตาปลอม

回复



Home  2016 年 1 月 6 日 上午 8:26

Thankfulness to my father who informed me regarding this webpage, this webpage is in fact amazing.

回复



restore iphone from backup

 2016 年 2 月 1 日 下午 4:05

However, what about this? what if you typed a catchier title?

I mean, I don't want to tell you how to run your website, but suppose you added something that grabbed people's attention? I mean 主题模型LDA及其在微博推荐&广告算法中的应用-第1期 - 微博推荐 is kinda vanilla.

You might peek at Yahoo's home page and see how they create news titles to get people to open the links. You might add a related video or a pic or two to grab readers excited about everything've written. Just my opinion, it could make your website a little bit more interesting.

my webpage - house cleaning jobs in houston tx (<http://www.edizheh.com>)

回复



csr racing 2 hack Ⓞ 2016 年 2 月 27 日 上午 11:18

Here you customize a ship and fly across the sky fighting waves of enemies.

Online CSR Racing 2 Hack are created for the browser and Facebook. Besides being visually appealing, these elements add more interest to the standard theme park management **CSR Racing 2 Hack**.

回复



bing Ⓞ 2016 年 3 月 22 日 上午 11:47

Oh my goodness! Amazing article dude! Thank you so much, However

I am having troubles with your RSS. I don't know the reason why

I cannot join it. Is there anybody having identical RSS problems?

Anyone who knows the answer will you kindly respond? Thanx!!

回复

评论

发表评论

