2011-10-19

Probabilistic latent semantic analysis (pLSA)

statistics algorithm machine learning topic model

Probabilistic latent semantic analysis (概率潜在语义分析，pLSA) 是一种 Topic model (http://en.wikipedia.org/wiki/Topic_model)，在99年被 Thomas Hofmann 提出。它和随后提出的 LDA (http://en.wikipedia.org/wiki/Latent_Dirichlet_allocation) 使得 Topic Model 成为了研究热点，其后的模型大都是建立在二者的基础上的。

我们有时会希望在数量庞大的文档库中自动地发现某些结构。比如我们希望在文档库发现若干个“主题”，并将每个主题用关键词的形式表现出来。 我们还希望知道每篇文章中各个主题占得比重如何，并据此判断两篇文章的相关程度。而 pLSA 就能完成这样的任务。

我之前取了 Wikinews (http://en.wikinews.org/) 中的 1000 篇新闻，试着用 pLSA 在其中发现 K=15 个主题。比如一篇关于 Wikileaks 的阿萨奇被保释消息 (http://en.wikinews.org/wiki/Wikileaks_founder_Julian_Assange_granted_bail,_set_free)的新闻，算法以 100% 的概率把它分给了主题 9，其关键词为：

> media phone hacking wikileaks assange australian stated information investigation murdoch

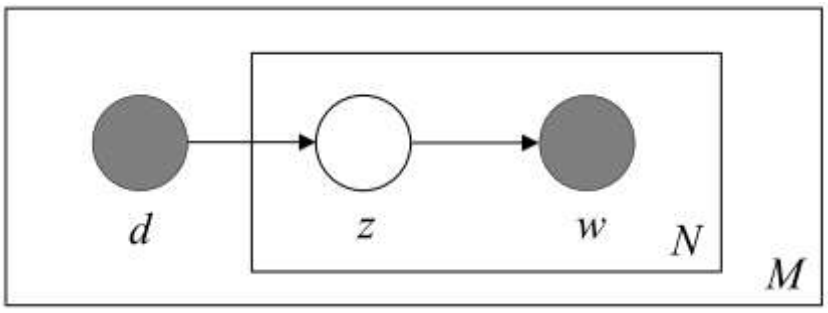
可以看到这个主题的发现还是非常靠谱的。又比如这条中国人民的老朋友威胁要大打打核战争 (http://en.wikinews.org/wiki/North_Korea_warns_of_'self-defensive_blows,'_nuclear_war,_if_military_exercises_take_place) 的新闻。 算法把它以 97.7% 的概率分给了主题 3，2.3% 的概率分给了主题 7。主题 3 的关键词是：

> south north court china military death tornado service million storm

主题 7 的关键词是：

> nuclear plant power japan million carbon radiation china water minister

对于每对出现的 (d, w) 都对应着一个表示“主题”的隐藏变量 $z \in Z$。 pLSA 是一个生成模型 (http://en.wikipedia.org/wiki/Generative_model)，它假设 d、w 和 z 之间的关系用贝叶斯网络 (http://en.wikipedia.org/wiki/Bayesian_network)表示是这样的（从 [Blei03] (#blei03) 偷的图）：



实心的节点 d 和 w 表示我们能观察到的文档和单词，空心的 z 表示我们观察不到的隐藏变量，用来表示隐含的主题。图中用了所谓的“盘子记法 (http://en.wikipedia.org/wiki/Plate_notation)”， 即用方框表示随机变量的重复。这里方框右下角的字母 M 和 N 分别表示有 M 篇文档，第 j 篇文档有 N_j 个单词。每条有向边表示随机变量间的依赖关系。也就是说，pLSA 假设每对 (d, w) 都是由下面的过程产生的：

1. 以 $P(d)$ 的先验概率选择一篇文档 d
2. 选定 d 后，以 $P(z|d)$ 的概率选中主题 z
3. 选中主题 z 后，以 $P(w|z)$ 的概率选中单词 w

而我们感兴趣的正是其中的 $P(z|d)$ 和 $P(w|z)$：利用前者我们可以知道每篇文章中各主题所占的比重，利用后者我们则能知道各单词在各主题中出现的概率，从而进一步找出各主题的“关键词”。记 $\theta = (P(z|d), P(w|z))$， 表示我们希望估计的模型参数。当然 θ 不仅仅代表两个数，而是对于每对 $(w^{(j)}, z^{(k)})$ 和 $(d^{(i)}, z^{(k)})$， 我们都要希望知道 $P(z^{(k)}|d^{(i)})$ 和 $P(w^{(j)}|z^{(k)})$ 的值。也就是说，模型中共有 $|Z| \cdot |D| + |W| \cdot |Z|$ 个参数。我们还知道：

$$
\begin{aligned}
P(d, w) &= P(d)P(w|d) \\
P(w|d) &= \sum_z P(w|z)P(z|d)
\end{aligned}
$$

根据最大log似然估计法，我们要求的就是

$$
\begin{aligned}
\arg\max_\theta L(\theta) &= \arg\max_\theta \sum_{d,w} n(d, w) \log P(d, w; \theta) \\
&= \arg\max_\theta \sum_{d,w} n(d, w) \log P(w|d; \theta) P(d) \\
&\left\{ \sum \quad (d \quad) 1 \quad P(\quad |d \ \theta) \quad \sum \quad (d \quad) 1 \quad P(d) \right\}
\end{aligned}
$$

$$\begin{aligned}\arg\max_\theta L(\theta) &= \arg\max_\theta \sum_{d,w} n(d,w)\log P(w|d;\theta) \\ &= \arg\max_\theta \sum_{d,w} n(d,w)\log \sum_z P(w|z)P(z|d)\end{aligned}$$

这里出现了 $\log$ 套 $\sum$ 的形式，导致很难直接拿它做最大似然。但假如能观察到 z，问题就很简单了。于是我们想到根据 EM 算法（参见我的上篇笔记 (/2011/10/em-algorithm/)），可以用下式迭代逼近 $\arg\max_\theta L(\theta)$：

$$\arg\max_\theta Q_t(\theta) = \arg\max_\theta \sum_{d,w} n(d,w) E_{z|d,w;\theta_t}[\log P(w,z|d;\theta)]$$

其中

$$\begin{aligned}E_{z|d,w;\theta_t}[\log P(w,z|d;\theta)] &= \sum_z P(z|d,w;\theta_t)\log P(w,z|d;\theta) \\ &= \sum_z P(z|d,w;\theta_t)[\log P(w|z) + \log P(z|d)]\end{aligned}$$

在 E-step 中，我们需要求出 $Q_t(\theta)$ 中除 θ 外的其它未知量，也就是说对于每组 $(d^{(i)}, w^{(j)}, z^{(k)})$ 我们都需要求出 $P(z^{(k)}|d^{(i)}, w^{(j)}; \theta_t)$。根据贝叶斯定理 (http://en.wikipedia.org/wiki/Bayes%27_theorem)贝叶斯定理，我们知道：

$$P(z^{(k)}|d^{(i)}, w^{(j)}; \theta_t) = \frac{P_t(z^{(k)}|d^{(i)})P_t(w^{(j)}|z^{(k)})}{\sum_z P_t(z|d^{(i)})P_t(w^{(j)}|z)}$$

而 $P_t(z|d)$ 和 $P_t(w|z)$ 就是上轮迭代求出的 θ_t。这样就完成了 E-step。

接下来 M-step 就是要求 $\arg\max_\theta Q_t(\theta)$ 了。利用基本的微积分工具 [2] (#id10)，可以分别对每对 $(w^{(j)}, z^{(k)})$ 和 $(d^{(i)}, z^{(k)})$ 求出：

$$\begin{aligned}P_{t+1}(w^{(j)}|z^{(k)}) &= \frac{\sum_d n(d,w^{(j)})P(z^{(k)}|d,w^{(j)};\theta_t)}{\sum_{d,w} n(d,w)P(z^{(k)}|d,w;\theta_t)} \\ P_{t+1}(z^{(k)}|d^{(i)}) &= \frac{\sum_w n(d^{(i)},w)P(z^{(k)}|d^{(i)},w;\theta_t)}{\sum_{w,z} n(d,w)P(z|d^{(i)},w;\theta_t)}\end{aligned}$$

以上就是 pLSA 算法了。最后贴个我用 MATLAB 写的实现 [3] (#id11)：

```
function [p_w_z, p_z_d, Lt] = pLSA(n_dw, n_z, iter_num)
% PLSA  Fit a pLSA model on given data
%       in which n_dw(d,w) is the number of occurrence of word w
%       in document d, d, n_z is the number of topics to be discovered
%

% pre-allocate space
[n_d, n_w] = size(n_dw); % max indices of d and w
p_z_d = rand(n_z, n_d); % p(z|d)
p_w_z = rand(n_w, n_z); % p(w|z)
n_p_z_dw = cell(n_z, 1); % n(d,w) * p(z|d,w)
for z = 1:n_z
    n_p_z_dw{z} = sprand(n_dw);
end

p_dw = sprand(n_dw); % p(d,w)
Lt = []; % log-likelihood
for i = 1:iter_num
    %disp('E-step');
    for d = 1:n_d
        for w = find(n_dw(d,:))
            for z = 1:n_z
                n_p_z_dw{z}(d,w) = p_z_d(z,d) * p_w_z(w,z) * ...
                    n_dw(d,w) / p_dw(d, w);
            end
        end
    end

    %disp('M-step');
    %disp('update p(z|d)')
    concat = cat(2, n_p_z_dw{:}); % make n_p_z_dw{:}(d,:)) possible
    for d = 1:n_d
        for z = 1:n_z
            p_z_d(z,d) = sum(n_p_z_dw{z}(d,:));
        end
        p_z_d(:,d) = p_z_d(:,d) / sum(concat(d,:));
    end

    %disp('update p(w|z)')
    for z = 1:n_z
        for w = 1:n_w
            p_w_z(w,z) = sum(n_p_z_dw{z}(:,w));
        end
        p w z(: z)   p w z(: z) / sum(n p z dw{z}(:));
```

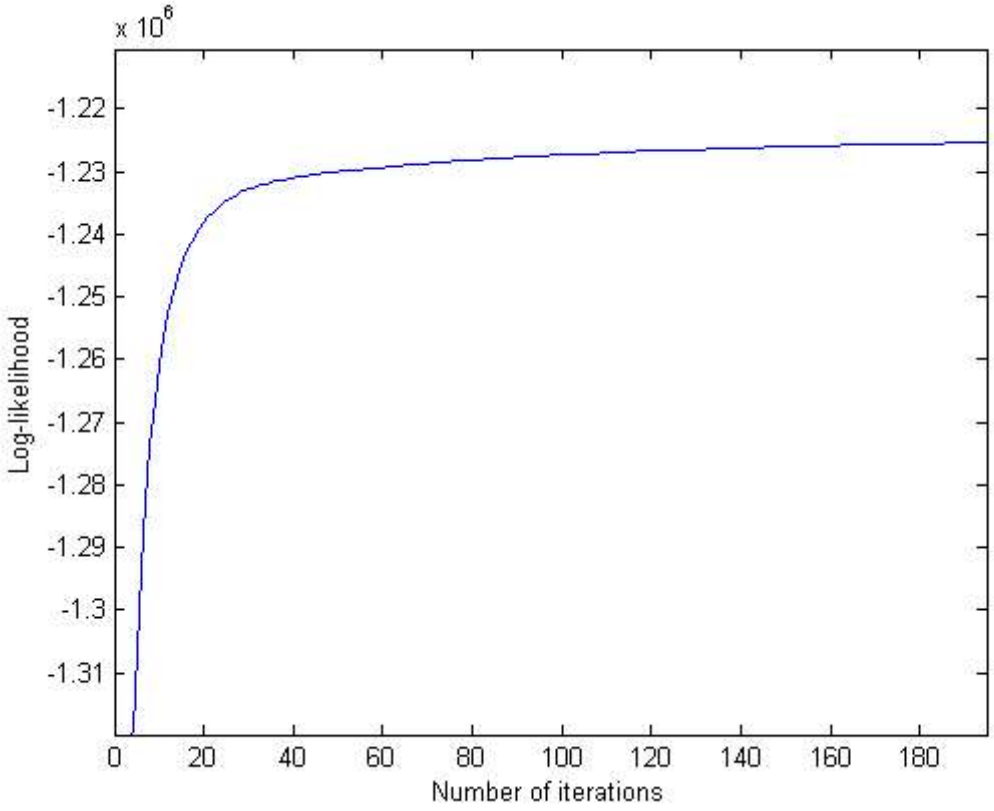
```
            end
            L = L + n_dw(d,w) * log(p_dw(d, w));
        end
    end

    Lt = [Lt; L];
    %plot(Lt); ylim([2*median(Lt)-L-0.1 L+(L-median(Lt))/2+0.1]);
    %drawnow; pause(0.1)
end

end
```

第一次拿 Mablab 写程序，比较丑..... [4] (#id12)

下图是 Log 似然度随迭代收敛的情况。可以看到收敛速度还是相对较快的。 而且由于是 EM 算法的缘故，Log 似然度确实是单调上升的.



最后，pLSA 的问题是在文档的层面上没有一个概率模型，每篇文档的 P(d|z) 都是需要拟合的模型参数。这就导致参数的数目会随文档数目线性增长、不能处理训练集外的文档这样的问题。所以02 David Blei、Andrew Ng（就是正在 ml-class.org (http://www.ml-class.org/) 里上公开课的那位） 和 Michael Jordan 又提出了一个更为简洁的模型：LDA。有时间的话下次再写了。

[2] 具体而言，这里要求的是 $Q_t(\theta)$ 在 $\sum_w P(w|z) = 1$ 和 $\sum_z P(z|d) = 1$ 约束条件下的极值。根据拉格朗日乘数法，解：

$$\nabla_\theta \left(Q(\theta) + \sum_z \alpha_z (\sum_w P(w|z) - 1) + \sum_d \beta_d (\sum_z P(z|d) - 1) \right) = \mathbf{0}$$

[3] 完整的程序和数据在这里。

[4] 吐槽：用 Matlab 做简单字符串处理怎么都那么恶心！长度不同的字符串竟然算是不同类型的！Cell array 怎么那么难用！

[Blei03] Blei, D.M. et al. 2003. Latent Dirichlet Allocation. Journal of Machine Learning Research. 3, 4-5 (2003), 993-1022.

[Hofmann99] Hofmann, T. 1999. Probabilistic latent semantic indexing. Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval SIGIR 99. pages, (1999), 50-57.

[Gildea99] Gildea, D. and Hofmann, T. 1999. Topic-based language models using EM. Proceedings of the 6th European Conference on Speech (1999), 2167-2170.

[Brants05] Brants, T. 2005. Test Data Likelihood for PLSA Models. Information Retrieval. (2005), 181-196.

Comments

27 Comments Tom Dong's Blog 1 Login

Recommend 4 Share 按评分高低排序

Join the discussion…

TH • 3年前

博主好　读了您的文章感到十分受益　我有两个问题请教您：1　文中代码16行的注释p(d w)是否应该是

Tom Mod ➚ Zhuyadong • 4年前

n_z是传给函数pLSA的参数

^ | v • Reply • Share ›

ningyuwhut • 3年前

原来作者的网站被墙了，晚上才想起来翻墙……

^ | v • Reply • Share ›

在路上 • 4年前

说好的LDA呢？ pLSA写的很好！

^ | v • Reply • Share ›

Fei Huang • 4年前

博主你好，对于测试机算主题都是用的fold in的方法。

但感觉从训练集的P(w)、P(Z)、P(w|z)得到P(z|w)，也能当做对测试集主题的估计方法，为什么没看到有人用呢 0 0

^ | v • Reply • Share ›

Lvp • 4年前

请问博主，

M-step的两个Pt+1公式是怎么推导出来的，论文中也没有详细写

^ | v • Reply • Share ›

Jerry Zhong ➚ Lvp • 4年前

博主已经说了，拉格朗日乘子

^ | v • Reply • Share ›

Furyng • 4年前

博主 你好，在更新pwd矩阵的时候，会不会出现pwd中元素大于1的情况，因为pwd是由pzd和pwz生成的，而EM过程中的pzd和pwz不是最终的结果，所以通过它们生成的pwd可能出现元素值大于1的情况，所以我觉得需要归一化一下，

^ | v • Reply • Share ›

Tom Mod ➚ Furyng • 4年前

不会，计算过程中p_z_d和p_w_z的值都满足概率分布的条件，所以算出的p_dw也会满足

^ | v • Reply • Share ›

2011xiaowei • 5年前

感谢博主详细的解释和认真负责的态度，我学到了很多。仔细拜读了你的程序，有点小问题请教在

Tom Mod 2011xiaowei • 4年前

这是对整个p_z_d矩阵的每列进行normalize。Matlab这里写得比较恶心，还是看上面列出的公式比较清楚。。

Reply • Share ›

Emianlinegzn • 5年前

还有Blei et al 2003的名字。。。万分感谢

Reply • Share ›

Tom Mod ➚ Emianlinegzn • 5年前

就是上面“参考文献”标题下倒数第二条啊：
(Brants, 2005) Brants, T. 2005. Test Data Likelihood for PLSA Models. Information Retrieval. (2005), 181-196.

Reply • Share ›

Emianlinegzn ➚ Tom • 5年前

囧了，光看见脚注了。。。

Reply • Share ›

Emianlinegzn • 5年前

能告诉我Brants 2005年的文章具体是哪篇吗？谢谢

Reply • Share ›

Englefly • 5年前

请教一个问题：算法pLSA中，topic集合 Z 的初始值是怎么得到的呢？是W，还是预先给定的集合？

Reply • Share ›

Tom Mod ➚ Englefly • 5年前

P(z|d)和P(w|z)初始都随机即可。详见上面的Matlab代码。

Reply • Share ›

Sh Clearsky • 5年前

那个pLSA代码的页面找不到呀，能不能再提供一份代码？

Reply • Share ›

Tom Mod ➚ Sh Clearsky • 5年前

请见这里：
https://github.com/tomtung/Lea...

Reply • Share ›

可以“fold in”，即固定P(w|z)，只算P(z|d)

Reply • Share ›

robot13 • 5年前

这个 P(w|d)=∑zP(w|z)P(z|d) 想了很久才发现
P(w|z,d)=P(w|z)是因为w和d在z确定的情况下是独立的
是这么推导的么?

PS. 验证码太强大

Reply • Share ›

Tom Mod ➦ robot13 • 5年前

对，d→z→w 这条 trail 被 z 给 d-separate了，d、w 因此条件独立。z 的数量远远少于 d 和 w，起到了一个类似“瓶颈”的作用。

Reply • Share ›

bluetracks • 5年前

有空讲讲LDA吧，这东西我已经听过不同的人讲过3遍了，还是不得要领。Topic Model里面这些概率真是必须有些Graph model的基础啊。。。不然太费劲了

Reply • Share ›

Tom Mod ➦ bluetracks • 5年前

表示 LDA 的推导还没弄明白。。。对于我这种数学菜还是太复杂了些 > <

Reply • Share ›

Subscribe 在您的网站上使用Disqus Add Disqus Add 隐私

Copyright © 逆铭 (tomtung) 2012 (cc) BY-NC-SA

Proudly powered by Pelican and the Gumby Framework.

Github Twitter Weibo Douban