BPR-RENDLE-reading

对论文的理解

以前的推荐都是想办法计算人们对一个物品的喜爱程度，从而进行推荐，目标函数大都是想让预测出的值与实际中的显性值匹配。而 BPR 则是从比较中来进行学习，通过两个物品的关系来进行对物品的排序，是一种针对 Ranking 的思想。

同时这也使得同等情况下能获得更多的信息。比如 0000011111 这 10 个物品，单个的话只有 10 条信息，而比较的话则有 25 条信息。扩大了信息的含量（假设 0 并不仅仅是不喜欢），从而能够更好的进行排序任务。

说一下 BPR 算法的好处：

BPR 其实是一种思想，思想在算法中的体现就是目标函数，即往哪走。也即目标函数。模型则是我们看待事物的方式，也就是对问题的抽象方式，比如 MF 和 kNN。算法则是我们在目标空间中寻找最优解的方法，就是如何达到目标。

这样 BPR 就可以很容易的应用到各种已有的算法中，同时本文也提出了一种好的学习算法 LearnBPR，使得 BPR 的可迁移性更强。

下面是对论文的详细解读

Abstract:

以前的不足

MF kNN Even though these methods are designed for the item prediction task of personalized ranking, none of them is directly optimized for ranking.

提出解决方法

A generic optimization criterion BPR-Opt: the maximum posterior estimator derived from a Bayesian analysis of the problem. And a learning method was provided for optimizing models with respect to BPR-Opt.

怎么用

We show how to apply our method to two state-of-the-art recommender models

用的好处

Our experiments indicate that for the task of personalized ranking our optimization method outperforms the standard learning techniques for MF and kNN.

Section2 Related work

MF models learned by SVD have shown to be very prone to overfitting. Thus regularized learning methods have been proposed.正则化是结构风险最小化策略的实现。一般模型越复杂，正则化值就越大。正则化的作用是选择经验风险与模型复杂度同时较小的模型。（李

航 机器学习方法 p20)

Section3 Personalized ranking

Interesting about implicit feedback systems is that only positive observations are available. The non-observed user-item pairs – e.g. a user has not bought an item yet – are a mixture of real negative feedback (the user is not interested in buying the item) and missing values (the user might want to buy the item in the future).
就是说没有"买的"的可能是不喜欢的也可能是没看到但喜欢的

Section4 BPR

Maximize the following posterior:
Part1

> The Bayesian formulation of finding the correct personalized ranking for all items $i \in I$ is to maximize the following posterior probability where Θ represents the parameter vector of an arbitrary model class (e.g. matrix factorization).
>
> $$p(\Theta| >_u) \propto p(>_u |\Theta)\, p(\Theta)$$

为什么呢?
>u 是我们所知道的，zta 是我们要寻找的。**我们要找到一个在已知信息下，与原始分布相似的 zta 值。**p(θ)使我们假设的分布。论文为了简单计算用了正态分布。

> We write the prior distribution of θ, our parameter of interest, as $P(\theta)$; this is a function that specifies which values of θ are more or less likely, based on our interpretation of our previous relevant information. The information gained from our new data is represented by the likelihood function, proportional to $P(D \mid \theta)$, which is then multiplied by the prior distribution (and rescaled) to yield the posterior distribution, $P(\theta \mid D)$, with which we can perform our desired inference. Thus, we can say that the likelihood function acts to update our prior distribution to a posterior distribution. A detailed technical introduction to Bayesian inference can be found in Etz and Vandekerckhove (in press), and interested readers can find an annotated list of useful Bayesian references in Etz, Gronau, Dablander, Edelsbrunner, and Baribault (in press).
>
> Mathematically, we use Bayes' rule to obtain the posterior distribution of our parameter of interest θ,
>
> $$P(\theta \mid D) = K \times P(\theta) \times P(D \mid \theta),$$
>
> where in this context $K = 1/P(D)$ is merely a rescaling constant. We often write this more simply as
>
> $$P(\theta \mid D) \propto P(\theta) \times P(D \mid \theta),$$

从这里我们可以看出想找 θ，要用假设的分布与先验知识一起来判断。

Suppose a coin is flipped n times and we observe x heads and $n - x$ tails. The probability of getting x heads in n flips is defined by the Binomial distribution,

$$P(X = x \mid p) = \binom{n}{x} p^x (1-p)^{n-x}, \tag{1}$$

where p is the probability of heads, and

$$\binom{n}{x} = \frac{n!}{x!(n-x!)}$$

is the number of ways to get x heads in n flips. For example, if $x = 2$ and $n = 3$, this value is $3!/(2! \times 1!) = 6/2 = 3$, since there are three ways to get two heads in three flips (i.e., Head-Head-Tail, Head-Tail-Head, Tail-Head-Head). Thus, the probability of getting two heads in three flips if p is .50 would be $3 \times .50^2 \times (1 - .50)^1 = .375$, or three out of eight.

If the coin were fair, so that $p = .50$, and we flip it ten times, the probability it comes up six heads and four tails is

$$P(X = 6 \mid p = .50) = \frac{10!}{6! \times 4!} (.50)^6 (1 - .50)^4 \approx .21.$$

If the coin were a trick coin, so that $p = .75$, the probability of six heads in ten tosses is

$$P(X = 6 \mid p = .75) = \frac{10!}{6! \times 4!} (.75)^6 (1 - .75)^4 \approx .15.$$

借助这个例子来理解。

https://psyarxiv.com/85ywt

Part2

we define the individual probability that a user really prefers item i to item j as:

$$p(i >_u j|\Theta) := \sigma(\hat{x}_{uij}(\Theta))$$

where σ is the logistic sigmoid:

$$\sigma(x) := \frac{1}{1 + e^{-x}}$$

其中，σ(x)是 sigmoid 函数。这里你也许会问，为什么可以用这个 sigmoid 函数来代替呢? **其实这里的代替可以选择其他的函数，不过式子需要满足 BPR 的完整性，反对称性和传递性。**原论文作者这么做除了是满足这三个性质外，另一个原因是为了方便优化计算。

https://www.cnblogs.com/pinard/p/9128682.html

Part3

So far, we have only discussed the likelihood function. In order to complete the Bayesian modeling approach of the personalized ranking task, we introduce a general prior density $p(\Theta)$ which is a normal distribution with zero mean and variance-covariance matrix Σ_Θ.

$$p(\Theta) \sim N(0, \Sigma_\Theta)$$

原作者为什么这么假设呢？个人觉得还是**为了优化方便，因为后面我们做优化时，需要计算 lnP(θ)，而对于上面假设的这个多维正态分布，其对数和||θ||2 成正比**

https://www.cnblogs.com/pinard/p/9128682.html

Part4

$$
\begin{aligned}
\text{BPR-OPT} &:= \ln p(\Theta| >_u) \\
&= \ln p(>_u |\Theta)\, p(\Theta) \\
&= \ln \prod_{(u,i,j)\in D_S} \sigma(\hat{x}_{uij})\, p(\Theta) \\
&= \sum_{(u,i,j)\in D_S} \ln \sigma(\hat{x}_{uij}) + \ln p(\Theta) \\
&= \sum_{(u,i,j)\in D_S} \ln \sigma(\hat{x}_{uij}) - \lambda_\Theta ||\Theta||^2
\end{aligned}
$$

where λ_Θ are model specific regularization parameters.

最终准则也就是目标函数。

Part5

原始论文：

$$
\begin{aligned}
\frac{\partial \text{BPR-OPT}}{\partial \Theta} &= \sum_{(u,i,j)\in D_S} \frac{\partial}{\partial \Theta} \ln \sigma(\hat{x}_{uij}) - \lambda_\Theta \frac{\partial}{\partial \Theta} ||\Theta||^2 \\
&\propto \sum_{(u,i,j)\in D_S} \frac{-e^{-\hat{x}_{uij}}}{1+e^{-\hat{x}_{uij}}} \cdot \frac{\partial}{\partial \Theta} \hat{x}_{uij} - \lambda_\Theta \Theta
\end{aligned}
$$

$$
\Theta \leftarrow \Theta + \alpha \left(\frac{e^{-\hat{x}_{uij}}}{1+e^{-\hat{x}_{uij}}} \cdot \frac{\partial}{\partial \Theta} \hat{x}_{uij} + \lambda_\Theta \Theta \right)
$$

其他论文：

http://www.shichuan.org/doc/35.pdf

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \theta} &= \sum_{(u,i,j)\in \mathcal{T}_r} \frac{\partial}{\partial \theta} ln\sigma(\hat{x}_{uij}) - \lambda_\theta \frac{\partial}{\partial \theta} \|\theta\|^2 \\
&\propto \sum_{(u,i,j)\in \mathcal{T}_r} \frac{1}{1+e^{\hat{x}_{uij}}} \frac{\partial}{\partial \theta} \hat{x}_{uij} - \lambda_\theta \theta.
\end{aligned}
$$

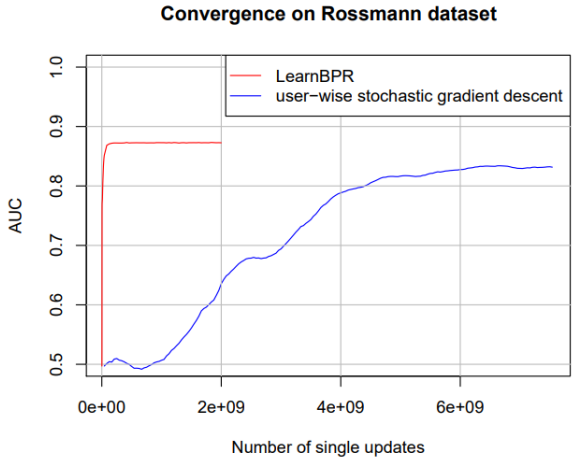
$$\theta \leftarrow \theta + \eta(\frac{1}{1+e^{\hat{x}_{uij}}}\frac{\partial}{\partial\theta}\hat{x}_{uij} - \lambda_\theta\theta),$$

猜测**原始论文求导数求错**了，多加了一个负号。

Part6:

> In general this is a good approach for our skew problem but the order in which the training pairs are traversed is crucial. A typical approach that traverses the data item-wise or user-wise will lead to poor convergence as there are so many consecutive updates on the same user-item pair – i.e. for one user-item pair (u, i) there are many j with $(u, i, j) \in D_S$.
>
> To solve this issue we suggest to use a stochastic gradient descent algorithm that chooses the triples randomly (uniformly distributed). With this approach

为什么顺序的效果不好？是因为会对（u，i）对做很多连续的更新，会导致过拟合的产生，使得效果不好，甚至会让 AUC 下降。



蓝色下降的过程猜测是对一个（u，i）对进行大量训练的结果：（u，i，j）有很多个。

下述论文在引用的时候说是 empirically 说明：

https://dl.acm.org/doi/10.1145/3109859.3109880

> adopted in [20]. It has been empirically shown [20] that the order in which the training pairs are traversed is crucial. In particular, an item-wise or user-wise traverse approach leads to poor convergence whereas the proposed bootstrap sampling method, that is drawing user-items triples uniformly at random, converges much faster. In order to learn the BPR-MF model, we thus use the LearnBPR algorithm from [20], summarized in Algorithm 1.

Part7:

Learning models with BPR

Because in our optimization we have triples $(u, i, j) \in D_S$, we first decompose the estimator $\hat{x}_{uij}$ and define it as:

$$\hat{x}_{uij} := \hat{x}_{ui} - \hat{x}_{uj}$$

MF

$$\hat{x}_{ui} = \langle w_u, h_i \rangle = \sum_{f=1}^{k} w_{uf} \cdot h_{if}$$

gorithm LEARNBPR. As stated before for optimizing with LEARNBPR, only the gradient of $\hat{x}_{uij}$ with respect to every model parameter θ has to be known. For the matrix factorization model the derivatives are:

$$\frac{\partial}{\partial \theta} \hat{x}_{uij} = \begin{cases} (h_{if} - h_{jf}) & \text{if } \theta = w_{uf}, \\ w_{uf} & \text{if } \theta = h_{if}, \\ -w_{uf} & \text{if } \theta = h_{jf}, \\ 0 & \text{else} \end{cases}$$

在开始学习之前，唯一需要知道的就是各个参数的梯度计算公式

kNN

$$\hat{x}_{ui} = \sum_{l \in I_u^+ \wedge l \neq i} c_{il}$$

where $C : I \times I$ is the symmetric item-correlation/ item-similarity matrix. Hence the model parameters of kNN are $\Theta = C$.

$$\frac{\partial}{\partial \theta} \hat{x}_{uij} = \begin{cases} +1 & \text{if } \theta \in \{c_{il}, c_{li}\} \wedge l \in I_u^+ \wedge l \neq i, \\ -1 & \text{if } \theta \in \{c_{jl}, c_{lj}\} \wedge l \in I_u^+ \wedge l \neq j, \\ 0 & \text{else} \end{cases}$$

Section5 Relations to other methods

WR-MF

$$\sum_{u \in U} \sum_{i \in I} c_{ui}(\langle w_u, h_i \rangle - 1)^2 + \lambda ||W||_f^2 + \lambda ||H||_f^2$$

where c_{ui} are not model parameters but apriori given weights for each tuple (u, i). Hu et al. have additional data to estimate c_{ui} for positive feedback and they set $c_{ui} = 1$ for the rest. Pan et al. suggest to set $c_{ui} = 1$ for positive feedback and choose lower constants for the rest.

区别:

> First of all, it is obvious that this optimization is on instance level (one item) instead of pair level (two items) as BPR. Apart from this, their optimization is a least-square which is known to correspond to the MLE for normally distributed random variables. However, the task of item prediction is actually not a regression (quantitative), but a classification (qualitative) one, so the logistic optimization is more appropriate.
>
> A strong point of WR-MF is that it can be learned in $O(\text{iter}\,(|S|\,k^2 + k^3\,(|I| + |U|)))$ provided that c_{ui} is constant for non-positive pairs. Our evaluation indicates that LearnBPR usually converges after a subsample of $m \cdot |S|$ single update steps even though there are much more triples to learn from.

WR-MF 还是在独立物品层面而 BPR 是在关系层面，并且 BPR 的学习算法更优

MMMF

$$\sum_{(u,i,j) \in D_s} \max(0, 1 - \langle w_u, h_i - h_j \rangle) + \lambda_w ||W||_f^2 + \lambda_h ||H||_f^2$$

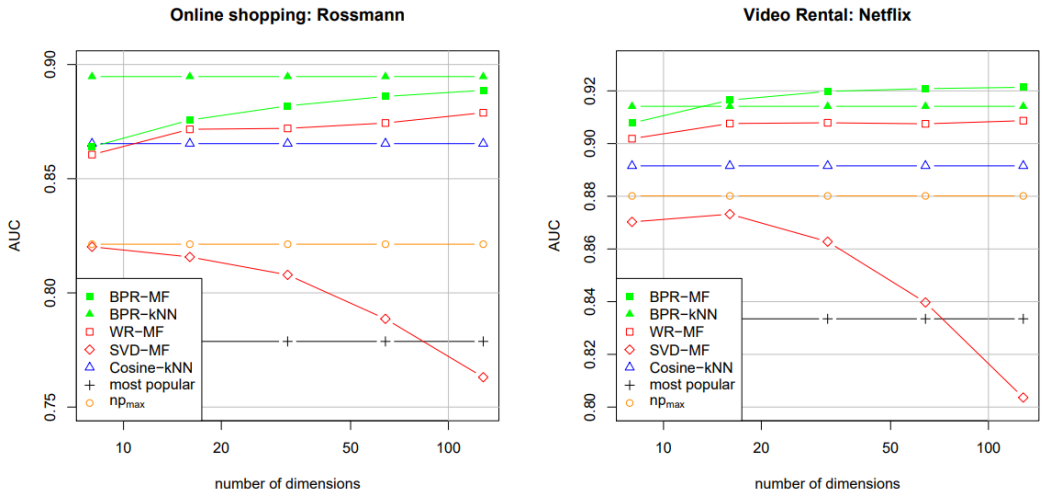
区别:

> One difference is that the error functions differ – our hinge loss is smooth and motivated by the MLE. Additionally, our BPR-Opt criterion is generic and can be applied to several models, whereas their method is specific for MF.

an implicit setting. But when their learning method is applied to implicit feedback datasets, the data has to be densified like described above and the number of training pairs D_S is in $O(|S||I|)$. Our method LEARNBPR can handle this situation by bootstrapping from D_S (see Section 4.2).

BPR 范围更广并且学习方法更优!

Section6 Evaluation



BPR 对模型效果的提升具有很好的帮助。

论文的详细解读到此结束

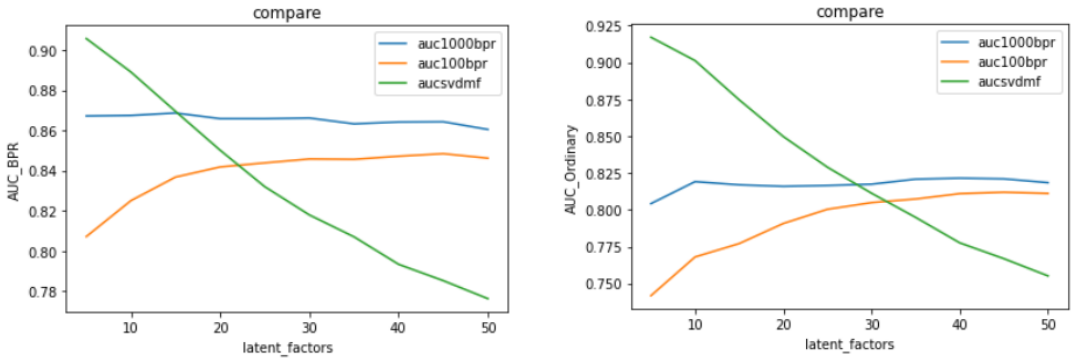
相关的算法开源实现的学习中的一些发现或者收获：

在数据科学中，最难搞的往往不是算法，而是数据，找数据和处理数据到获得自己想要的格式会花费大量的时间。而在本论文的复现中也是一样。刚开始找代码的时候，他们的代码写的都非常的结构化，对于我来说很不合适，而且有很多代码是用 tf 库写的，我比较熟悉 pytorch 的格式。

最后找到了别人的 github 的仓库 https://github.com/RunlongYu/BPR_MPR ，数据时处理好的格式。但是他仅仅提供了 BPR-MF 算法，AUC 的计算方式也不是论文中的计算方式。我在他原有代码的基础上增加了用 SVD-MF 算法处理的过程。并实现了**论文中 AUC 的计算方法**。对于不同的评估方法，最后的结果会有较大差异。

在仓库作者提供的较小数据集 **943 Users; 1682 Items** 中，SVD-MF 在 latent_factors 较小时效果很好。BPR-MF 算法随着 latent_factors 的增大效果越好，并且训练次数的提升也会带来较大好处。论文中数据集为 **10, 000 users on 4000 items**，数据集相对较大。并且论文中说“We use the leave one out evaluation scheme, where we remove for each user randomly one action (one useritem pair) from his history, i.e. we remove one entry from I + u per user u.“他每一个用户仅仅剔除一个，这样相对而言学到的知识就越多，对剩下一个预测自然就越好。而我实现的数据集是五五分的，训练集和测试集一样大小。如此，SVD-MF 效果好也是可以理解的，因为他更能反应矩阵整体的分布情况。

可见，并没有适用于所有情景的好算法，只能根据实际情况来分析问题。



在阅读文献的过程也有一些其他的收获：

NDCG 以及 推荐系统常用的评估方法：

https://towardsdatascience.com/evaluate-your-recommendation-engine-using-ndcg-759a851452d1

https://towardsdatascience.com/ranking-evaluation-metrics-for-recommender-systems-263d0a66ef54

SGD 与 Adam

https://medium.com/mdr-inc/from-sgd-to-adam-c9fce513c4bb