KDD'24 | Airbnb:基于模型蒸馏的多目标排序方法

原创 州懂学习笔记 州懂学习笔记 2024年10月08日 08:58 广东

州懂学习笔记
分享大模型推荐系统相关知识和学习笔记
53篇原创内容

公众号

KDD'24 | Airbnb:基于模型蒸馏的多目标排序方法

> 标题: Multi-objective Learning to Rank by Model Distillation
> 地址: https://arxiv.org/pdf/2407.07181
> 公司: Airbnb
> 会议: KDD'24

1. 前言

Airbnb的文章还是一如既往地满满干货, 十分贴近实际业务实践。这篇文章所讨论的多目标排序问题是搜广推系统最常被讨论的一个问题。

现有的多目标优化方法可以粗略分成两类:

- **标量化方法:** 将多个目标的损失函数做线性加权, 是业界最常见的多目标优化方法。特别提及ϵ-约束方法, 它将其中一个目标作为优化的主目标, 而将其它次要的目标作为约束, 这种方法文献上也认为是一种标量化方法。
- **多目标进化算法:** 使用进化算法(基于群体搜索的随机优化方法)去处理多目标优化问题。

文章所提方法与标量化方法相关, 现有的标量化多目标排序方法主要存在以下不足&挑战:

1. **多任务学习方法效率不高, 是次优解:** 不同目标的数据是不均衡的, 像取消订单的数据就要少很多。通常来说, 多目标学习会使用share bottom的多任务学习机制, 这会使数据量较少的任务可以从数据量较多的任务中收益, 但当目标间相关性比较低甚至有冲突时, 学习效率会比较低, 是一种次优解。当然, 即便是次优解, 也会比每个目标单独训练一个模型再做加权求和的模型融合方法要好很多。
2. **推理融合权重搜参效率不高, 且不稳定:** 多任务学习方法需要调整两组参数: 一是训练时多个目标的损失权重, 另一个是推理时在线分数的融合权重。推理时的融合权重网格搜索效率不高, 找到的权重系数也不能保证是最优的, 并且这个权重系数并不稳定, 当换一个模型时, 原来的融合权重可能就不适用了, 需要重新搜参调整。

3. **只适合于目标损失函数是可微的情况:** 深度学习方法要求模型的每个目标损失函数是可微的, 但在真实的业务场景, 有些业务目标可能只是一个既定的规则, 比如希望推荐中新内容的分发占比至少达到x%的目标。这类目标无法直接加到优化目标中, 只能在模型训练完之后再人工Boosting干预。

基于现有标量化多目标学习方法的不足, 作者观察到多目标优化的标量化方法和模型蒸馏的损失函数具有相似的结构，首次提出了基于模型蒸馏的多目标优化排序方法, 下面进行具体介绍。

2. 方法

作者首先基于一系列的理论推导, 建立起了多目标学习损失函数与模型蒸馏具有相似结构的联系, 再基于这种相似结构的发现, 进一步提出了基于模型蒸馏的多目标学习排序方法。

2.1 多目标优化->模型蒸馏

2.1.1 多目标优化问题的形式化描述

在Airbnb这样的双边在线市场, 通常会以一个业务目标(如CVR)作为主要目标，同时将其他次要业务目标视为约束(如订单取消率等)。这样的多目标优化问题与前面ϵ-约束方法一致, 即:

$$\begin{aligned} \min_{\theta} \quad & C_1(f(\theta, X)) \\ \text{s.t.} \quad & C_k(f(\theta, X)) \leq \varepsilon_k, k = 2, \ldots, K \end{aligned}$$

其中, θ为可学习的模型参数, X为召回之后, 待排序的n个Item所对应的各种特征, $C_k(f(\theta, X))$为第k个目标的损失函数, 这里$C_1(f(\theta, X))$是主要目标,其它为次要目标, 相应的ε_k为目标损失的约束上界。

2.1.2 多目标优化问题的推导转换

假设每个目标的损失函数是独立进行优化的, 也就是对于每个目标只需优化$C_k(f(\theta, X))$训练模型$f_k(\theta_k^*, X)$, 而不需要考虑其它目标的约束, 记相应的Loss为$C_k(f_k(\theta_k^*, X))$, 那么这个Loss可以看作是$C_k(f(\theta, X))$的下界限值, 因为多目标优化的模型在对应目标上的Loss肯定不会比单独目标优化的模型要好

$$C_k(f_k(\theta_k^*, X)) \leq C_k(f(\theta, X)) \leq C_k(f_k(\theta_k^*, X)) + \varepsilon_k^{'}$$

这样就有:

$$\Rightarrow | \ C_k(f(\theta, X)) - C_k(f_k(\theta_k^*, X)) \ | \leq \varepsilon_k^{'}$$

这意味着我们可以容忍第k个目标的损失值$C_k(X)$最多比$C_k^*(X)$差$\varepsilon_k^{'}$。

如果每个目标函数都是Lipschitz连续的，则有:

$$|C_k(f(\theta, X)) - C_k(f_k(\theta_k^*, X)) | \leq M \mid f(\theta, X) - f_k(\theta_k^*, X)|$$

其中M是一个常数，称为Lipschitz常数。结合上述两个式子, 我们可以找到代理约束(这一步推导过程笔者没有看懂, 看明白的同学欢迎留言讨论):

$$|f(\theta, X) - f_k(\theta_k^*, X)| \leq \varepsilon_k''$$

从而将原始的多目标优化函数改写成:

$$\begin{aligned} \min_\theta \quad & C_1(f(\theta, X)) \\ \text{s.t.} \quad & |f(\theta, X) - f_k(\theta_k^*, X)| \leq \varepsilon_k'', k = 2, \ldots, K \end{aligned}$$

对于上式, 使用拉格朗日松驰方法引入松驰变量ω_k, 上式多目标优化函数可以进一步改写成:

$$\min_\theta C_1(f(\theta, X)) + \sum_{k=2}^{K} (\omega_k \mid f(\theta, X) - f_k(\theta_k^*, X) \mid)$$

这里, LTR预估一般使用交叉熵作为每个目标的损失函数, 它是Lipschitz连续的, 欧式距离也可以用KL距离代替, 它与交叉熵是等价的, 因此, 上式就可以写成:

$$\min_\theta CE(f(\theta, X), L_1) + \sum_{k=2}^{K} \omega_k CE(f(\theta, X), f_k(\theta_k^*, X))$$

其中, L_1为主目标对应的Label, 而$CE(\cdot)$为交叉熵损失, 这里第2项可以进一步简化:

$$\begin{aligned} & \sum_{k=2}^{K} \omega_k CE(f(\theta, X), f_k(\theta_k^*, X)) \\ & = \sum_{k=2}^{K} \omega_k \sum_i -f_k(\theta_k^*, X_i) \cdot log f(\theta, X_i) \\ & = \sum_i -log f(\theta, X_i) \cdot \sum_{k=2}^{K} \omega_k f_k(\theta_k^*, X_i) \\ & = CE(f(\theta, X), \sum_{k=2}^{K} \omega_k f_k(\theta_k^*, X)) \end{aligned}$$

此外, 将主目标自己的优化解近似为附加约束也是很Make Sense的, 这样, 就可以得到如下优化函数:

$$\min_\theta CE(f(\theta, X), L_1) + CE(f(\theta, X), \sum_{k=1}^{K} \omega_k f_k(\theta_k^*, X))$$

2.1.3 多目标优化与模型蒸馏的联系

观察前面的损失(单目标)函数，可以发现, 它与模型蒸馏损失函数非常相似, 由两部分构成:

- **hard label对应的损失:** 预测结果与主目标Ground Truth之间的损失
- **soft label对应的损失:** 指前面通过聚合K个目标独立优化的结果$\sum_{k=1}^{K} \omega_k f_k(\theta_k^*, X)$, 再通过蒸馏损失推动模型逼近Soft Label。可以看出, 这里对各目标独立优化的结果做了模型融合, 实现了模型融合和模型蒸馏的有机统一。

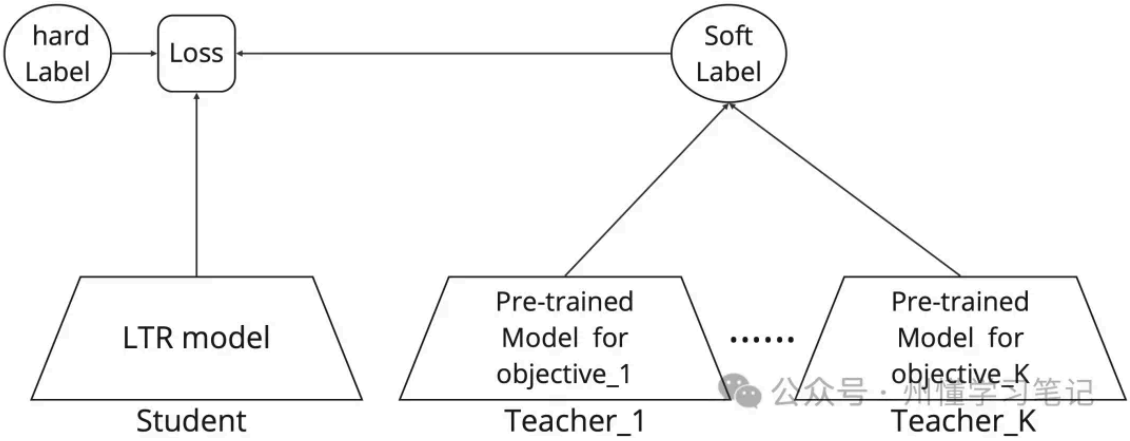
基于前面的理论分析, 为了优化多目标排序, 主要涉及以下几个步骤:

1. 分别为每个目标训练一个模型$f_k(\theta_k^*, X)$, 不考虑其它目标。

2. 对于每个训练样本, 生成soft label $\sum_{k=1}^{K} \omega_k f_k(\theta_k^*, X)$。
3. 同时考虑soft label和主要目标的hard label, 做模型蒸馏，从而得到一个在次要目标约束下优化主要目标的模型。

2.2 多目标排序系统整体框架

下图给出了基于知识蒸馏的多目标排序系统(multi-objective learning to rank system with model distillation, 论文简称为MO-LTR-MD)的整体框架。



可以看出, 它的Loss由hard label和soft label两部分组成, 这里, 作者使用α改写了前面的Loss:

$$Loss = -\alpha \sum_i l_i log f(\theta, X_i) - (1-\alpha) \sum_i \hat{l}_i log f(\theta, X_i)$$

- 第一部分是hard label loss, 可以是任意类型的learning-to-rank loss, 这里对于n个待排序的Item, 作者是使用Softmax Cross Entropy $f(\theta, X_i)\ |_p = \frac{e^{z_{i,p}}}{\sum_{j=1}^{n} e^{z_{i,j}}}$计算第$p$个Item的损失。
- 第二部分是soft label loss, 其中, $\hat{l}_i$是对应的soft label, 即有$\hat{l}_i = \sum_{k=1}^{K} \omega_k s_k$, 这里$s_k = f_k(\theta_k^*, X)$是从每个预训练模型计算的排名分数。需要注意的是, 和其它模型蒸馏一样, 这里$f(\theta, X_i)$会在Softmax那里会引入温度系数。

2.3 深度理解Soft Label

2.3.1 Soft Label与Hard Label的对比

首先, 先回顾下Hard Label是如何起作用的。

Hard Label是从用户行为数据中收集的, 它不仅反映了用户的偏好, 也反映了排序模型的偏好, 因为只有被排序模型认可的Item才会曝光。但是呢, Hard Label其实是高度稀疏的, 可能很多次推荐只会产生一次预订行为。这就导致Hard Label只能帮助我们获得部分基本事实：预订列表的排名高于用户未预订的列表，但未预定的列表的排序无从得知。

与Hard Label相反, Soft Label纯粹只反映了模型的偏好, 用户偏好倒是缺失的。Soft Label被定义为多个预训练模型的聚合，因此它实际上反映了一个简单的多目标排序系统的偏好，这个简单的系统可能不是最优的, 但通过Soft Label可以补充所有列表的排序。

2.3.2 Soft Label的优点

作者汇总了Soft Label的一些优点:

- **有助于缓解数据不平衡问题:** 为每个训练样本分配了一个dense型的Soft Label向量，有助于缓解数据不平衡, 避免再进行上采样/下采样的操作。
- **充当了正则化的作用:** Soft Label给出了多目标LTR模型的先验知识。
- **可以很好的处理不可微目标：** 像新内容曝光占比的业务目标是不可微的, 使用soft label可以很好的处理, 见下一章节的详细介绍.
- **可以有效地对知识排序，并传递给新版本的排序模型:** 通过Soft Label可以将模型学到的知识传递给新版本，有助于减少模型的不可再现性和不稳定性, 下面章节详细介绍。

2.4 不可微的特定业务目标

在真实业务场景中,常有一些不可微的业务目标, 比如新内容在推荐中的曝光占比, 这种目标本身就是模糊的，不能从过去的数据中学习优化。此外, 由于新内容历史行为交互稀疏, 一些与目标相关的特征(如CVR)缺失, 这使得传统的多目标排序优化难以处理这种不可微的业务目标。

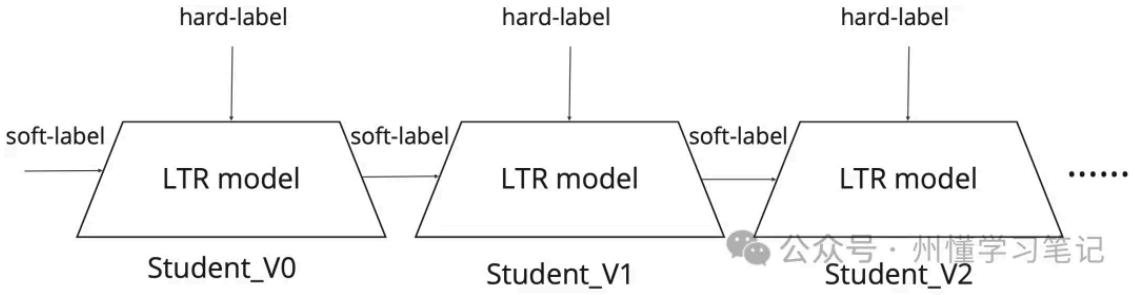
一种解决方案是通过对相似的成熟Item的特征值求平均, 来获取新Item的特征，但并不能保证学习效率, 且可能会损害用户体验。

另一种解决方案是直接修改训练数据的label, 新内容的hard label给一定的boost $\beta > 0$, 但这种将新内容都放在了其它未购买内容之上的方式并不合理。但这种方式boost直接用到soft label上就比较make sense了, 并且它也能像hard label一样非常高效。后面实验会详细介绍这里。

2.5 训练开销和不可再现性

论文强调了多目标排序模型的不可再现性和不稳定性问题, 这是说两个从头开始训练的模型, 即使它们使用的训练数据和模型结构都一样, 最终训练完的模型它们在指标上可能依然有差异。当模型使用最新数据训练时, 这种不可再现性会加剧模型的不稳定性。而多目标排序模型由于需要优化平衡多个目标, 稳定性无疑是非常重要的。

此外, Soft Label是基于预训练模型中得到的, 会存在训练开销的问题, 同时这些预训练模型也定期重新训练，也具有不可再现性。受Born-Again Neural Networks的启发, 作者提出了自蒸馏的方式来传递Soft Label到新版本模型中, 如下图所示:



- V_0版本的模型, 按前面多目标排序框架里的方式训练, 从多个预训练的模型中得到Soft Label用于模型蒸馏。
- 对于后续版本的模型, Soft Label不再依赖于预训练模型, 而是直接使用前一版本模型的Soft Label

通过这种设计，训练复杂度显着降低：不需要维护和更新预训练模型，而是将这些知识提炼成Soft Label并沿模型训练路径传递。当然，如果需要，这些预训练模型还是可以随时插入。作者做了一些实验，证实了这种自蒸馏对业务指标没有负面影响，这种创新方法可以帮助减少模型训练不可再现性和不稳定性。

3. 实验部分

3.1 实验设置上的对比

作者对比了baseline与论文所提的MO-LTR-MD方法在训练和推理阶段的一些优势, 如下图所示:

Model	Training Loss weights	Score fusion weights
Baseline	more than K weights	K weights
MO-LTR-MD	one distillation weight	0

Table 1: Model Training and Serving Comparison

概括起来就是:

- 更少的训练数据: 360M V.S. 500M
- 训练时超参更少, 只需要一个模型蒸馏的权重α, 论文里$\alpha = 0.2$
- 推理时不需要额外的融合权重

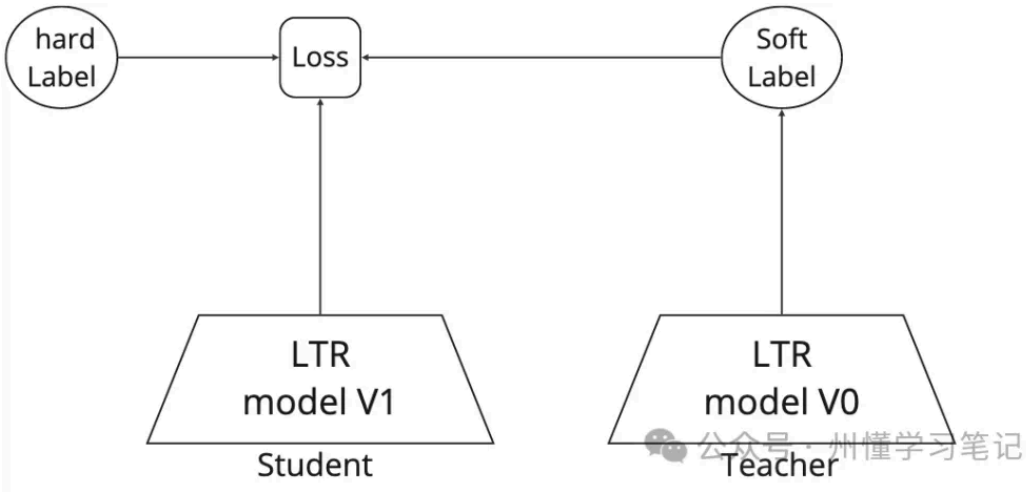
3.2 整体模型性能

- **离线评估:** 使用NDCG, 和baseline相比，NDCG显著提升+1.1%
- **在线AB测试:** CVR提升+0.37%(对应显著性p值=0.02), 而其他次要目标差异不大
- **训练成本&推理时延:** 虽然MO-LTR-MD需要加载多个预训练模型生成Soft Label, 但由于使用了更少的数据进行训练, 训练时间与baseline持平。由于只有一个Student模型, 推理时延显著减少1.6%。

3.3 自蒸馏测试

为了测试自蒸馏方式是稳定的, 作者设计了下面的流程:

- V_0模型是使用多个预训练模型的Soft Label训练的
- V_1模型使用V_0模型的Soft Label训练
- 对比V_0模型, V_1模型训练数据时间窗口后移了几个月



实验结果:

- 离线NDCG几乎相同, 这表明，即使没预训练模型，仅使用Soft Label也不会有什么影响, 且新训练数据并不会稀释Soft Label的作用
- 线上ab测试主目标和次要目标的也基本持平

3.4 模型不可再生性测试

使用不同的随机初始权重和相同的训练数据集训练了一组相同的baseline模型和MO-LTR-MD模型, 分别使用side-by-side评估结果的差异以及使用Relative Prediction Difference(PD)指标评估打分的差异。

Model	Change rate	PD
Baseline	77%	0.407
MO-LTR-MD	36%	0.363
Improvement	53%	11%

3.5 不可微的特定业务目标测试

作者模拟了一些从未在我们系统中应用过的业务目标, 比如在搜索结果靠前位置提高更高评论评级的item, 会对高评论内容做Boosting

- baseline模型对推理打分做Boosting:

$$s_i = \begin{cases} s_i & \text{for} \quad r_i < \rho \\ s_i + \alpha & \text{for} \quad r_i \geq \rho \end{cases}$$

- 作者所提方法MO-LTR-MD在Soft Label上做Boostring

$$\hat{l}'_i = \begin{cases} \hat{l}_i & \text{for} \quad r_i < \rho \\ \hat{l}_i + \beta & \text{for} \quad r_i \geq \rho \end{cases}$$

微调α和β, 使两个模型最终高评论内容的占比相差不大, 实验结果如下:

Methods	NDCG
directly boost score	-0.5%
Inject boost to soft-label	-0.1%

Table 3: Boosting Impact To NDCG

可以发现, 相比于直接在推理打分做Boosting的方法, 在Soft Label上做Boosting对数据伤害会更小。