

26 | 重排序：如何对排序后的内容进行微调？

2023-06-14 黄鸿波 来自北京

《手把手带你搭建推荐系统》

shikey.com转载分享

你好，我是黄鸿波。

前面我们学习了基于机器学习算法的 GBDT+LR 排序和基于深度学习 DeepFM 的排序模型，它们都是在工业界中比较常用的排序模型，今天我们抛开算法和模型，讲一讲推荐系统得到排序结果后的处理，也就是重排序。

今天的课程主要分为下面两大部分。

1. 什么是重排序以及为什么需要重排序。
2. 常见的重排序的方案有哪些。

重排序的概念及作用

我们先来看一看如果不使用重排序的方案会产生什么问题。

在现有的方案中，我们将排序之后的结果直接按照用户喜好程度来进行推荐。这就会产生一些多样性和局部最优解的问题。

比如推荐系统可能只关注与用户历史记录相似的商品，而忽略了其他有价值的商品。另外如果只按照排序结果进行推荐，推荐系统可能会出现某一类别的商品过于集中的情况。这样会导致用户没有机会体验到其他类别的商品，满意度下降。

还有一种情况，如果只按照排序结果进行推荐，推荐系统可能会过于依赖用户个人历史记录，从而忽略了其他有价值的推荐因素，例如热门商品和新品。

我们来考虑这样一个场景，假设在信息流推荐系统中一次性推出来了 15 条内容，这 15 条内容从展示形式来看，遍及了视频和图文两个类别，其中视频有 7 条，图文有 8 条。由于这个用户平时比较喜欢用 App 产品看视频（看图文相对较少），因此在排序阶段，会认为用户点击视频的概率更大。那么这 15 条内容直接就会按照前面 7 条都是视频，后面 8 条都是图文的形式进行排列。

把展现形式换成类别，实际上也是一样的道理。把这 15 条内容分别放在新闻、财经、体育这三个类别中，如果每一个类别的内容都挨在一起，也会造成不好的用户体验。

如果按照这样的形式进行推荐，最后的体验一定好不到哪里去，这时重排序的方案就登场了。

经典的 EE 问题

刚刚从内容的展现形式来说明了为什么需要重排序，我们再看看另外一个角度：排序后的处理。

前面说的重排序都是建立在排序层有输出结果的前提下，而排序层的内容又非常依赖于召回层。也就是说，如果在召回阶段中召回集就已经产生了问题，无论排序模型再怎么优秀都不会推出更好的结果。

我们拿基于 Item 的协同过滤来举例说明一下。基于 Item 的协同过滤算法推荐结果取决于用户过去评分过的 Items 集合，当前的推荐结果只是在该集合内寻找用户喜欢的 Items，并且没

有考虑用户可能有新的兴趣点的可能性，从而难以发现全新的 Items，导致系统陷入局部最优解。

具体来说，基于 Item 的协同过滤算法是以物品之间的相似度为基础，先构建一个物品相似度矩阵，然后通过计算用户已经评分过的物品与其余物品之间的相似度，计算待推荐物品的得分值。这里的问题在于，如果系统只根据已有的数据进行推荐，根本无法察觉到一些全新的物品，并且也可能会重复推荐已经推荐过的物品。

简单来说，如果用户在一段时间内经常看一类内容，就会导致使用基于 Item 的协调过滤算法最后只推荐那一类内容。

一个不好的召回集的另外一个极端：在召回集中什么类别都有，其实也就意味着和没有采用召回算法是一样的效果（随机推荐）。

这些问题就是推荐算法中比较经典的一个问题——EE 问题。

在推荐系统中，EE（Exploration-Exploitation）问题是指当推荐系统面对用户的时候，应该是“探索”还是“利用”。

在一个推荐系统中，用户可能会有某些个性化的偏好和兴趣，而这些偏好和兴趣在数据中未必能被准确发现。如果我们只根据已有的数据进行推荐，就可能会导致对一些用户的兴趣和偏好无法很好满足，从而失去兴趣。

另一方面，如果我们为了探索用户的深层次兴趣而随机推荐内容，用户可能会感到困惑、无聊或者失去耐心。因此，一个好的推荐系统应该在个性化推荐和探索用户更多兴趣之间找到一个合适的平衡点，从而实现用户兴趣最大化和平台利益最大化的双赢。

EE 问题就是要在探索和利用之间找到这个平衡点。在处理 EE 问题时，我们需要充分发掘用户的兴趣和偏爱，同时也需要随机进行探索性的推荐，并在每次推荐中根据用户的反馈增加或减少探索的概率，以达到更好的推荐效果。

一般来讲，处理 EE 问题常见的方法有下面这五种。

1. **探索性推荐**：在推荐用户可能喜欢的内容的时候，探索一些用户可能没有想到或者没有接触过的新内容。可以通过增加一些随机性和多样性等手段来实现探索性推荐，从而平衡探索和利用之间的关系。
2. **引入领域知识**：借鉴领域知识，为推荐系统提供更准确的信息，帮助系统更好地利用已有的信息，从而减少探索的需求。
3. **利用深度学习等技术提高模型准确性和效率**：深度学习技术可以帮助推荐系统更好地理解用户、内容和交互等特征，提高模型的准确性和效率。
4. **整合多种算法**：将不同算法的结果进行融合，从而得到更全面和准确的推荐结果。
5. **动态调整推荐策略**：根据用户反馈和平台目标不断修改推荐算法，动态调整推荐策略和探索率，从而达到更好地探索和利用平衡点。

总之，处理推荐系统中的 EE 问题需要综合考虑多种因素，并结合具体业务需求和算法模型，采取合适的探索策略，从而找到最好的平衡点，提高推荐的准确性和效率。


如果召回集已经产生了不好的效果，我们利用排序后的其他手段也可以解决。最主要的方式就是在这个阶段加入一些全新且多样化的内容，然后再穿插到最后的推荐列表中。

常见的重排序的方案

知道了为什么需要进行重排序后，接下来，我们就来简单介绍一些常见的重排序（排序后的处理）方案。

类别重排序

第一种就是针对刚刚我们说的，15 个列表中有很多重复的类别，如何进行重新排序。我们可以使用 Python 代码来模拟一下，代码如下。


 复制代码

```
1 import random
2
3 lst = ['a', 'a', 'a', 'a', 'b', 'b', 'b', 'b', 'b', 'c', 'c', 'c', 'c', 'c', 'c']
4 random.shuffle(lst)
5
6 while any([x==y for x, y in zip(lst, lst[1:])]):
7     random.shuffle(lst)
```

```
8
9 print(lst)
```

这个代码实现的非常简洁。首先使用 `random.shuffle()` 方法将列表 `lst` 打乱。然后使用 `zip()` 方法将列表中的相邻元素进行组合，然后遍历这些组合。如果有任意一对相邻元素完全相同，则说明列表 `lst` 有连续的两个元素相同，需要再次打乱。当所有的相邻元素都不相同时，输出结果即可。

执行代码后，会得到如下输出。

 复制代码

```
1 ['a', 'b', 'c', 'b', 'c', 'a', 'c', 'b', 'a', 'c', 'b', 'c', 'b', 'a', 'c']
```

如果把列表扩充一下，看看会有什么结果，假设现在总体的代码变成下面这样。

 复制代码

```
1 import random
2
3 lst = ['a', 'a', 'a', 'a', 'b', 'b', 'b', 'b', 'b', 'b', 'c', 'c', 'c', 'c', 'c', 'c', 'd', 'd', 'd', 'c']
4 random.shuffle(lst)
5
6 while any([x==y for x, y in zip(lst, lst[1:])]):
7     random.shuffle(lst)
8
9 print(lst)
```

也就是说我后面又加了 3 个字母 `d` 和一个字母 `c`，然后重新执行下这段代码，输出就会变成下面这样。

 复制代码

```
1 ['c', 'b', 'a', 'b', 'c', 'b', 'c', 'd', 'c', 'd', 'c', 'a', 'c', 'b', 'a', 'b',
```


由此可见，这个代码实际上是可扩展的。在实际的工程中，我们就可以利用这种方式来解决类别和内容的重排序的问题。

多样性处理

多样性处理也是常见的排序后处理方案。多样性处理的主要目的是在推荐结果中引入一些与用户历史兴趣相关但又不完全相同的推荐内容，从而增加推荐的多样性，提高用户对推荐结果的满意度。

首先可以设定一个多样性的阈值，例如某一个类别下至少有多少条内容（或者至少有多少个类别），然后取排序层的结果，去看这个排序层的结果是否满足这样的条件。如果满足这样的条件则进行输出，如果不满足这样的条件则结合热度值（或最新内容以及所在的类别）等取用户没有阅读过的内容，插入到排序列表中形成一个新的列表。

在排序后处理中利用多样性处理，可以为推荐系统的性能和用户体验带来多方面的提升。需要根据具体推荐场景选择合适的多样性处理方法，并综合考虑推荐的精度、效率和用户满意度等各方面需求，从多个角度来提升推荐系统的性能和效果。

随机化处理

随机化处理也是推荐系统后处理中比较常见的方式。一般来讲，我们可以在排序层后加入一些随机内容形成最后的推荐列表，给用户进行推荐，这样做有下面三个好处。

1. **提高推荐的多样性**：随机内容可以增加推荐列表中的多样性，从而避免推荐重复的或相似的内容，提高用户的满意度和使用体验。
2. **促进发现新内容**：随机内容的加入可以让用户发现一些他们之前没有想到过的内容，有助于推广产品和服务，增加用户对平台的粘性。
3. **对推荐算法进行调整和优化**：随机内容通过引入随机变量，可以用于推荐算法的优化，因为它可以帮助推荐算法发现更多的相似项和跨领域的推荐。

需要注意的是，在加入随机内容时不要过度，避免影响推荐的准确性和效果。另外，需要根据用户的兴趣偏好和行为习惯，进行个性化的随机内容推荐。

运营处理

运营处理也是排序后处理的常见操作。系统将由运营人员手动选择（或按照某种规则自动选择）的指定内容加入到推荐列表中。这些运营指定内容可以是广告、促销活动、某个品牌或商品以及其他支持商业化的内容等。通过引入这些特定内容，可以提高平台或企业的曝光率、营销效果、用户体验和收益。

shikey.com转载分享

加入运营指定内容需要基于用户的行为和兴趣偏好来进行个性化推荐，还需要平衡推荐列表的多样性和指定内容的曝光率，以避免过度干扰推荐系统排序过程。

具体来说有以下四种方式。

1. **加入固定的推荐内容**：在推荐列表中加入一些指定的固定内容，可以通过运营人员手动选择或者按照某种规则自动选择。
2. **利用 Slate 技术**：推荐系统可以使用 Slate 技术（也称作 Slot），将特定的推荐内容插入到特定的区域位置。比如在搜索页面里，可以将运营指定的内容插入到搜索结果中，提高用户的点击率。
3. **添加推荐条目优先级**：推荐系统可以添加优先级来对推荐内容进行调整。在最终的推荐列表中将运营指定的内容的优先级提高，以便其在排序过程中被优先推荐。
4. **基于标签的推荐**：推荐系统可以利用标签来进行内容的识别和分类。运营人员指定的内容可以根据标签进行识别，然后在相应的标签下加入推荐列表。

需要注意的是，为了提高推荐系统的效果和准确性，加入运营指定的内容应该尽量遵循用户的兴趣偏好和行为习惯。另外，在加入运营指定内容时，也需要注意避免干扰推荐系统本身的运行。

总结

到目前为止，本节课的内容就学完了，下面我们来做一个总结，本节课一共有下面四个重点。

1. 了解推荐系统中的重排序和排序后处理的概念及作用。

2. 在推荐系统中，EE 问题指的是在利用已知数据进行推荐时，是否应该通过收集更多数据来探索新的信息以提高推荐效果。也就是说，当推荐系统在进行已有策略（即已有的推荐算法或模型）和不确定性之间的权衡时，EE 问题成为了关键。
3. 解决 EE 问题的关键是要平衡探索和利用。在利用已有的推荐算法或模型进行推荐的同时，通过引入一些探索性推荐、试验和新的数据收集来提高推荐效果。需要注意的是，你需要合理地平衡探索和利用，才能有效提升推荐系统的效果和用户满意度。
4. 常见的重排序方案一般有类别的重排序、多样性处理、随机化处理、运营处理等，你需要熟悉它们各自的操作流程和注意事项。

思考题

本节课学完了，给你留两道思考题。

1. 你还能想到哪些重排序和推荐后排序的方法。
2. 了解一下 Bandit 方法如何解决 EE 问题。

期待你的分享，如果今天的内容让你有所收获，也欢迎你推荐给有需要的朋友！

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

精选留言 (2)



peter

2023-06-16 来自北京

在一个真实的中大型公司里，推荐系统的算法是怎么用的？公司的推荐系统会有多种推荐算法，对于某个推荐类型，会根据实际情况组合多种推荐算法，是这样吗？比如，推荐A内容，采用“算法1+算法4”；推荐B内容，采用“算法2+算法4 + 算法5”。是这样吗？



爱极客

2023-06-15 来自广东

请问老师，这几节课的理论，在课程项目里面能用到吗？



shikey.com转载分享