这个点击率模型，值得用户托付终身

原创 十方 炼丹笔记 昨天

收录于话题
#搜索推荐前沿算法 53 #必读论文系列 4

↑↑↑关注后"星标"**炼丹笔记**

炼丹笔记干货

作者：十方，三品炼丹师

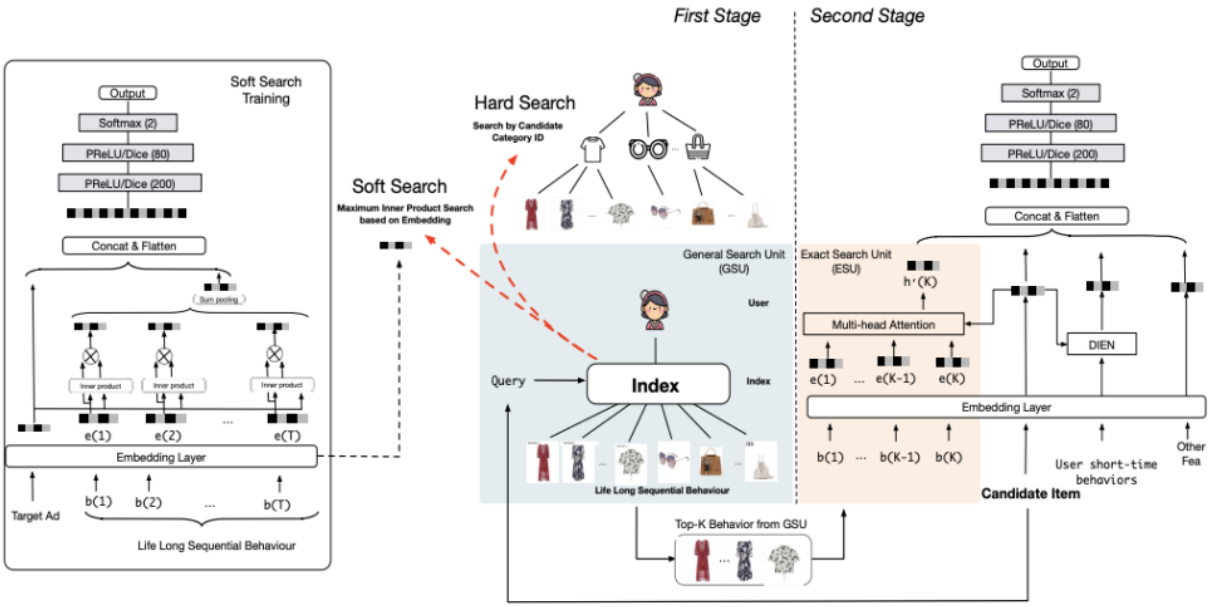
Search-based User Interest Modeling with Lifelong Sequential Behavior Data for Click-Through Rate Prediction

阿里对行为序列的研究可以说已经独领风骚了，前有DIN，后有MIMN，现在又出了这篇SIM。只能说行为序列确实对点击率预估很重要，阿里已经证明，丰富的用户行为数据对工业场景下推荐系统的点击率预估具有很大的价值。MINN已经把序列长度增加到了1000，然而当长度超过1000，MIMN很难准确捕捉用户兴趣了。淘宝23%的用户在过去5个月点击都超过1000个item，所以这篇论文想对任意长度的行为序列进行建模。

SIM长啥样

下图就是SIM，是个two-stage的策略，每个stage伴随着一个重要的unit，General Search Unit(GSU)和Exact Search Unit(ESU)。

- first-stage: 这一步用GSU在线性时间内，把原始的长序列提取出top-K的子序列，K远远小于原始序列长度。
- second-stage: 这一步用ESU把first-stage提取的top-K子序列作为输入，用一个类似DIN,DIEN的复杂结构精确的提取兴趣。



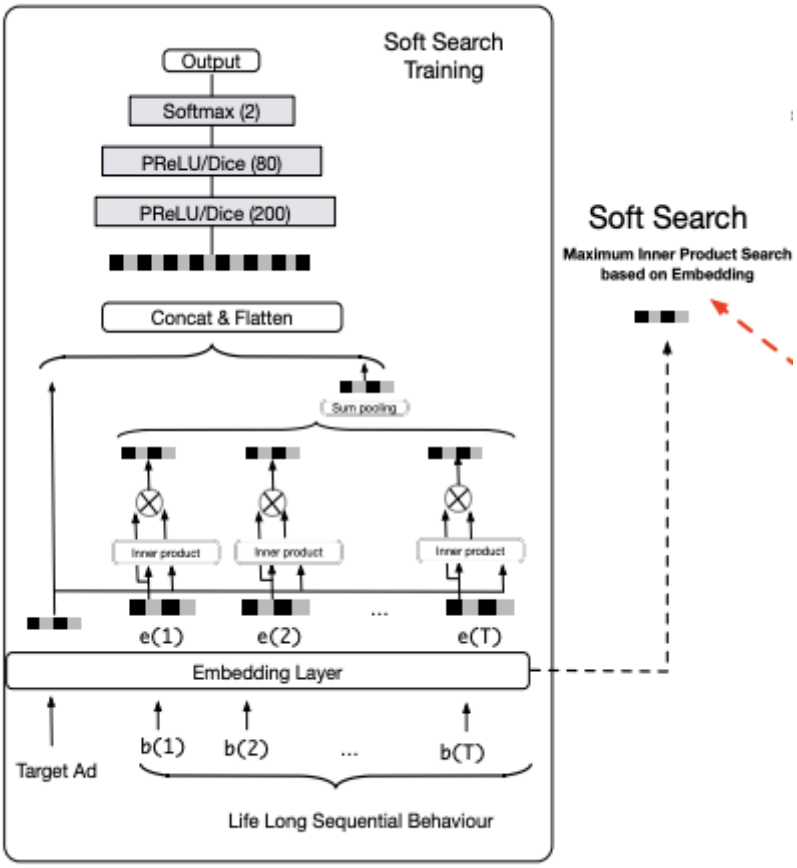
General Search Unit

用户大量的历史行为，对当前点击率预估真正有用的也就一部分行为，所以无需把原始行为序列全都塞到模型中去，这样资源消耗和性能瓶颈是无法承受的，所以干脆做个筛选，GSU就是这个功能。论文给出GSU两种实现方式Hard-search和Soft-search。假设我们现在拿到的原始用户行为序列B = [b1, b2, ..., bT]，GSU会计算每个行为bi的相关性ri，最后取相关性最高的K个行为作为新的序列。GSU两种实现方式本质就是ri计算方式不同，如下所示:

$$r_i = \begin{cases} Sign(C_i = C_a) & hard-search \\ (W_b \mathbf{e}_i) \odot (W_a \mathbf{e}_a)^T & soft-search \end{cases}$$

hard-search: 该方法没有任何参数，就是找到和目标item相同类目的历史序列的子序列。
soft-search: Wa和Wb是权重，ei和ea分别是序列第i个item和目标item的embedding，然后计算点积作为分数。论文提到可以用ALSH寻找top K的item，可以有个次线性的时间复杂度。

看到这里可能会有个疑问，ei和ea是怎么来的呢？见下图：



需要注意的是，b1~bT必须长期兴趣序列，因为GSU虽然是要从原始序列抽取K个兴趣，但是是为长期兴趣服务的，所以必须保证分布一致。图中的Sum pooling计算如下式:

$$\mathbf{U}_r = \sum_{i=1}^{T} r_i \mathbf{e}_i$$

然后Ur和目标向量concat在一起，接mlp预估点击率即可。

Exact Search Unit

通过GSU，我们已经获取了一个K长度的序列了，Exact Search Unit以该序列 B* 作为输入，训练一个基于attention的模型。考虑到B*序列每个item的权重是不同的，本能上都会觉得越靠近预估的item权重越大，论文里是把距离预估的item的时间差进行embeding为Et(长度为K，embsize为D的序列，类似bert中的位置编码)，B* 序列的也是一个长度为K的embeding序列E*。zb = concat((E*, Et), axis = 1)，是一个长度为K，embsize为2*D的序列，作为用户最终的兴趣序列，然后用多头attention:

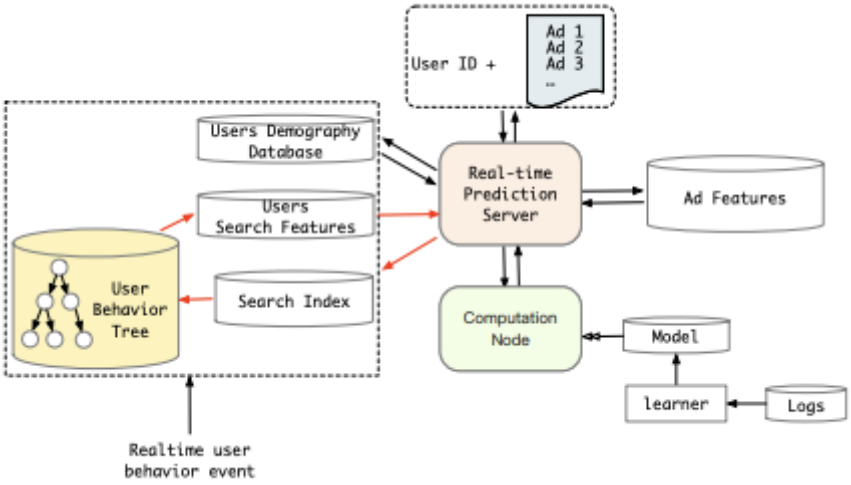
$$\mathbf{att}^{i}_{score} = Softmax(W_{bi}\mathbf{z}_b \odot W_{ai}\mathbf{e}_a)$$
$$\mathbf{head}_i = \mathbf{att}^{i}_{score}\mathbf{z}_b,$$

i表示第i个head，最后concat所有的head，输入到mlp中做ctr预估。如果我们用的是soft-search，loss就有两个了，LossGSU和LossESU，这两个loss权重都是1，如果用hard-search，只用LossESU即可。

$$Loss = \alpha Loss_{GSU} + \beta Loss_{ESU}$$

只会飞没用，要能落地

毕竟再好的模型，不能上线都是白做，特别是在线服务都必须是毫秒级别，论文提到实施预估系统延迟需要低于30ms，以阿里的体量，流量巅峰时每秒要处理百万用户，论文给出了实实在在的上线方案，如下图：



论文提到hard-search和soft-search选取的top-k序列，惊人的相似，所以考虑到性能资源，就采用hard-search的方式上线。既然用hard-search，就要找与目标item相同category的序列，所以论文提出了user behavior tree(UBT)，就是两层索引，第一层key为userid，第二层key为类目，最后value为行为序列。UBT用分布式系统实现，占22TB空间（有资源就是任性）。

实验

最后来看一下SIM的实验效果，首先看一下在阿里两个数据集的表现：

Table 2: Model performance (AUC) on public datasets

Model	Taobao (mean ± std)	Amazon (mean ± std)
DIN	0.9214 ± 0.00017	0.7276 ± 0.00051
Avg-Pooling Long DIN	0.9281 ± 0.00025	0.7280 ± 0.00012
MIMN	0.9278 ± 0.00035	0.7396 ± 0.00037
SIM (soft)[a]	0.9416 ± 0.00049	**0.7510 ± 0.00052**
SIM (soft) with Timeinfo	**0.9501 ± 0.00017**	-[b]

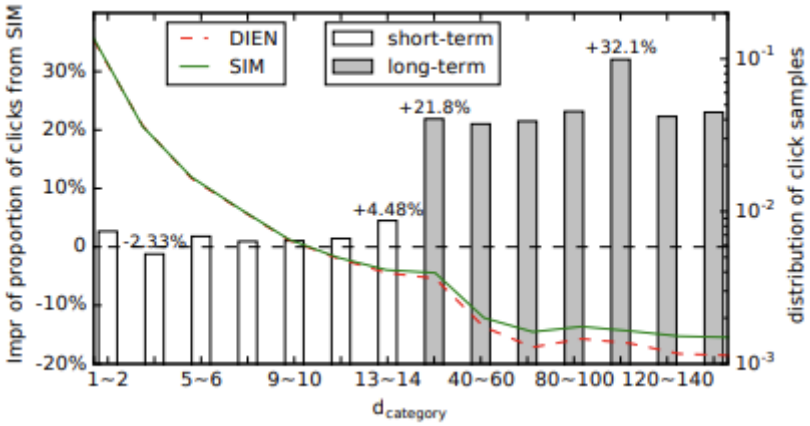
[a] SIM (soft) is SIM with soft search without time interval embeddings
[b] We didn't conduct the experiment on Amazon Dataset, as there are no timestamp features in it

Table 4: Model performance (AUC) on industrial dataset

Model	AUC
DIEN	0.6452
MIMN	0.6541
SIM (hard)	0.6604
SIM (soft)	0.6625
SIM (hard) with timeinfo[a]	0.6624

[a] The model has been deployed in our online serving system and is serving the main traffic now.

着重看下下图：



我们可以看到在短期行为的预估准确度DIEN和SIM是几乎一样的，但是长期行为上，SIM的优势就体现了。

收录于话题 #搜索推荐前沿算法·53个

下一篇 · 别扯高大上，告诉我哪些有用！

喜欢此内容的人还喜欢

Kaggle GM dott: 比赛很上瘾很耗时,我可没多余时间干其他的!

kaggle竞赛宝典

美团配送实时特征平台建设实践

DataFunTalk