京东电商搜索中的语义检索与商品排序

原创 王松林、唐国瑜 DataFunTalk 2020-06-03

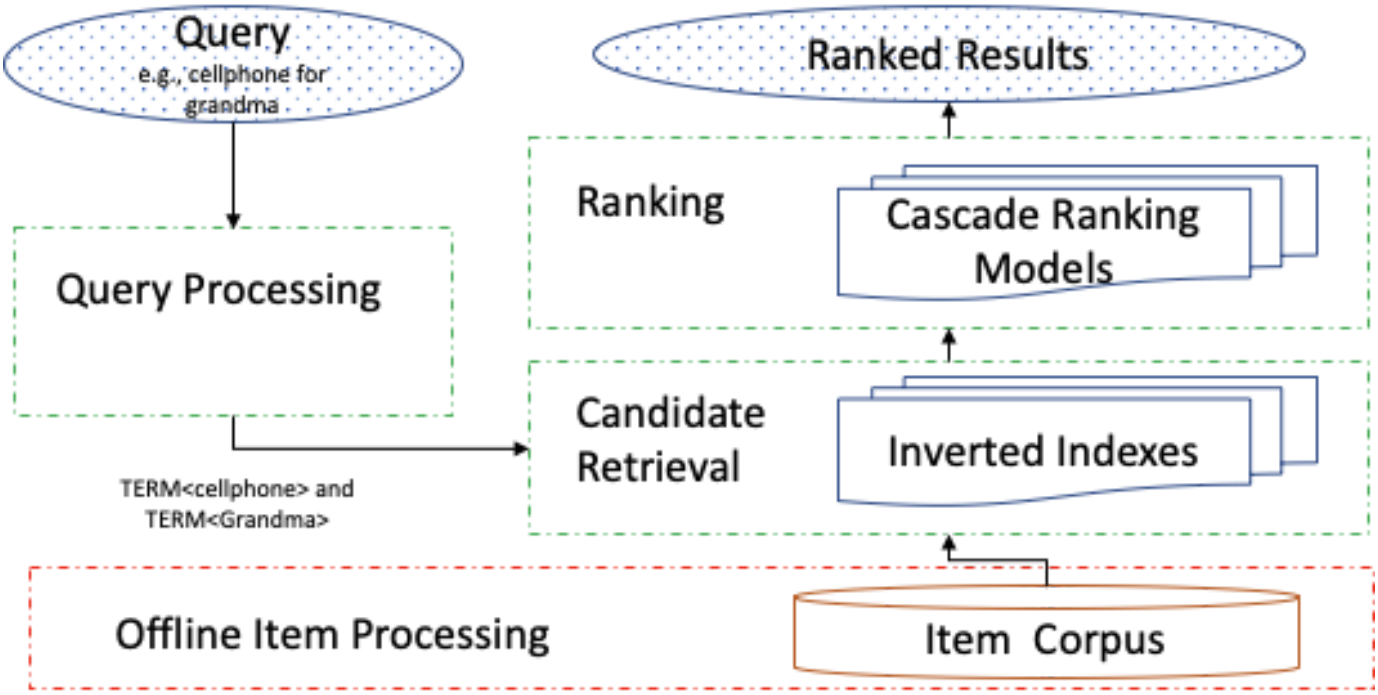
文章作者：王松林、唐国瑜 京东算法工程师
编辑整理：Hoh
内容来源：作者授权
出品平台：DataFunTalk
注：欢迎转载，转载请留言。

导读：本文将介绍京东搜索场景中的两块技术，**语义检索与商品排序。**在业界检索算法基础上，我们提出一系列更适用于电商场景的检索排序算法，在业务上取得了显著收益。其中的多篇论文已被 KDD/SIGIR 等收录。

01

背景介绍

电子商务搜索是京东等电商重要组成部分，用户通过搜索找到自己需要的商品，然后下单购买。一个典型电商搜索引擎的架构，包括三个重要组成部分：**query 理解、召回和排序。**



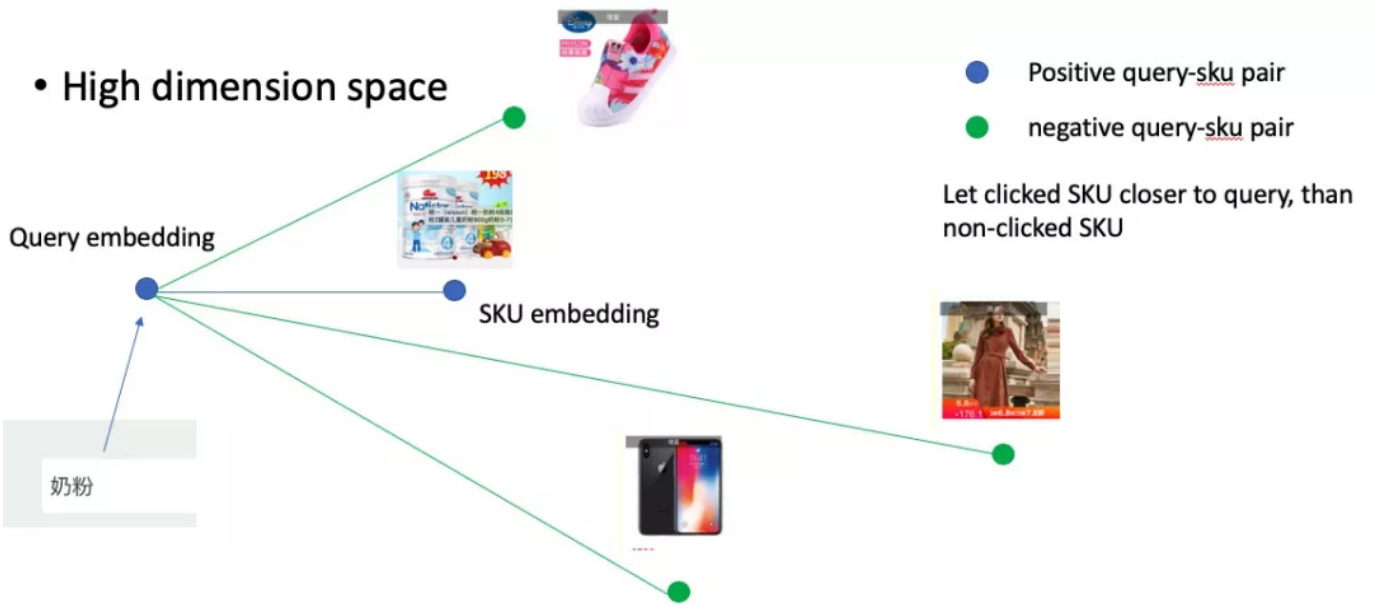
- Query 理解：包括 query 的纠错、改写、扩展、分词等。
- 召回阶段：给定一个查询词，从商品库中召回有效正确的商品候选集，并将结果返回给排序。召回方式有很多种，这里我们只介绍基于向量检索的召回。
- 排序阶段：给定召回商品的候选集合，根据众多因子对这些商品进行排序，挑选出最好的候选商品展示给用户。

下面我们分别介绍，基于向量检索召回和商品排序：

02

向量召回

向量检索作为一种信息检索方式在工业界已经被广泛应用，它能解决传统倒排检索不能解决的问题。倒排通过字面匹配方式召回商品，这种方式存在一种缺陷，不能召回字面不匹配但语义层面相近的商品， 如 query='2-3周岁宝宝玩具'是无法召回 sku='托马斯小火车'的。



通俗的讲就是训练一个模型，该模型通过将 query 和 sku 映射到统一维度空间，在该空间中，相似的商品距离近，不相近的商品距离较远。如上图例子，query=奶粉，在高纬空间里，相对鞋子、服装、手机，奶粉商品距离 query 更近。这是建模过程，生成 query 和 sku 的向量数据。

我们得到了 query 和 sku 的向量，接下来就是做检索，返回与 query 距离近的 topK 个 sku。而数据库的商品量非常多，通常是十亿级，不可能做线性遍历，考虑到时效性，会引入快速向量近似检索方法，如 KDTree、TDM、LSH、PQ、HNSW 等等，我们采用的是 PQ 算法，这里不再赘述，网上有很多材料介绍其算法。下面重点介绍我们的模型及在线检索框架。

模型方面不仅要考虑 query-sku 的相关性，我们也对用户行为进行建模，同一 query 针对不同用户、同一用户不同时刻检索出更具有个性化的商品。我们使用的是 DPSR（Deep Personalized and Semantic Retrieval）算法，模型融合个性化和搜索语义信息，我们的论文已被 SIGIR2020 收录。

1. 检索系统 overview

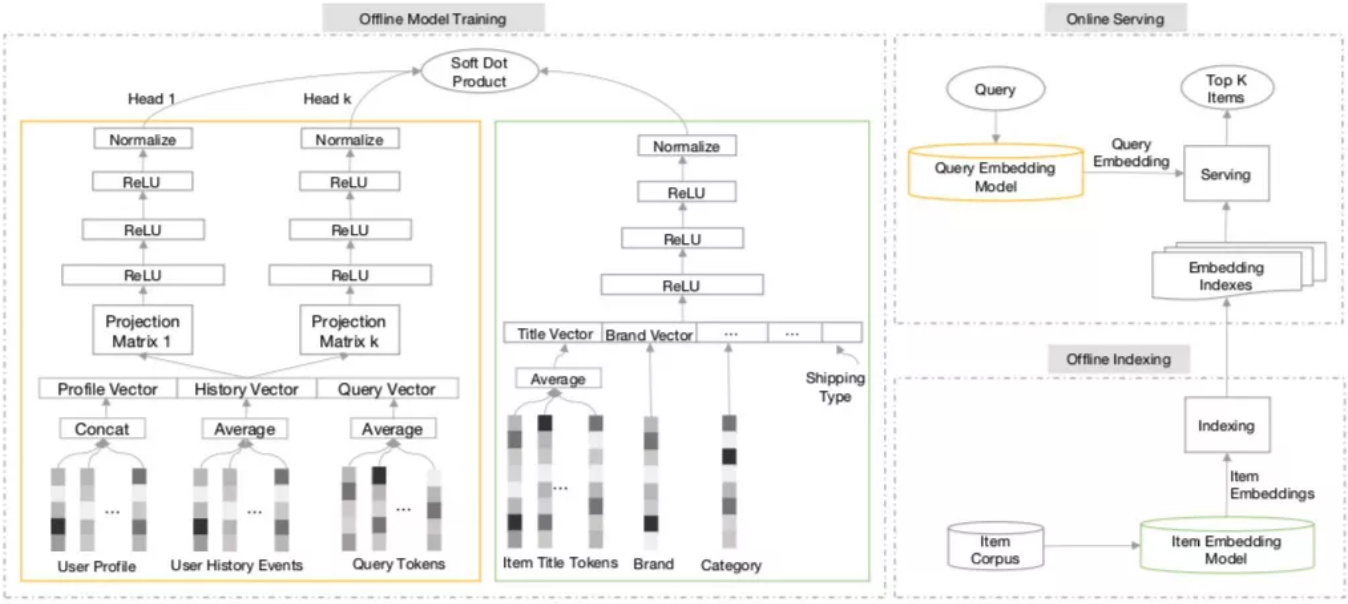


Figure 3: Overview of our DPSR retrieval system.

从整体看，离线模型是一个双塔模型结构，query 和 sku 分别有一个 model tower。Query 端包括了 query 包括 query tokens、user profile、user history events 等特征。Sku 端包括 title tokens、brand、category、shopid 等特征。

离线索引（offline indexing），使用的是 sku tower，导出 sku 的 embedding 构建 QP 索引。

在线服务（online serving）使用的是 query tower，模型加载在 tensorflow service，在线 predict query 的 embedding。

2. 模型详细设计

① Two tower model architecture

上面介绍了模型结构，一个 query tower Q，一个 sku tower S，对于给定的 query=q，sku=s，模型计算过程为：

$$f(q,s)=G(Q(q),S(s))$$

$Q(q)\in R^{d\times m}$ 表示 query 的 embedding

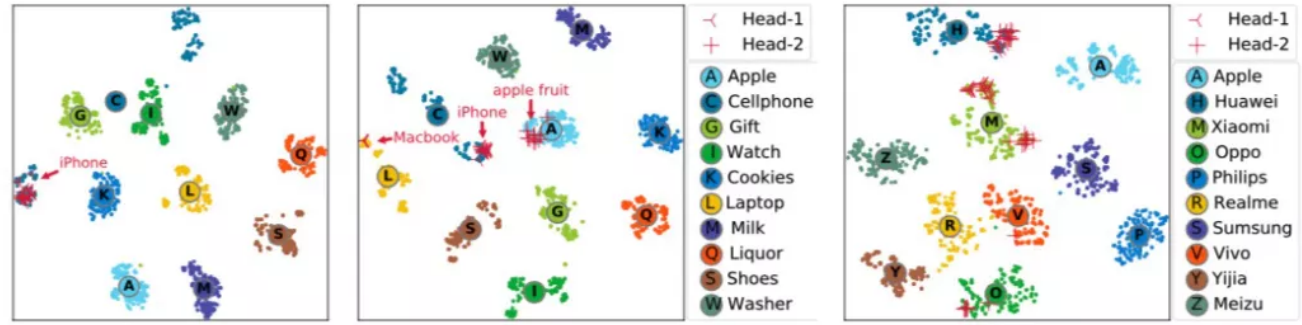
$S(s) \in R^{d \times m}$ 表示 sku 的 embedding

G 表示打分计算函数，比如 inner product、L2 distance 等

双塔模型训练完后，query 和 sku 的模型相对独立，我们可以分别计算他们。所有的 sku embedding 都在离线计算，以便快速构建向量检索索引。虽然 model 是相互独立的，但 query 和 sku 之间使用简单的点积计算，理论上 query 和 sku embedding 仍然在同一个几何空间中，具有可比性。

② Query tower with multi heads

我们看到左侧的 tower 和右侧有两点不一样：Projection layer 和 mutli heads，目的是为了丰富 query 侧的信息。如下图所示，不同的 head 可以捕获 query 不同的语义（query=苹果，语义可以是手机和水果），捕获不同的品牌属性（query=手机，品牌可以是华为、小米），捕获不同的产品属性（query=三星，产品属性可以是笔记本、手机）等等。



(a) One head for query "apple". (b) Two head retrieval for query "apple". (c) Two head retrieval for query "cellphone".

Figure 6: t-SNE visualizations of retrieval results for polysemous queries.

③ Attention Loss

Multi heads 让 query 可以生成多个 embedding 与 sku embedding 计算 score。我们采用 attenion loss 做模型优化。

我们标记 query 的多个 embeding 为 $Q(q)=\{e_1,e_2,...,e_m\}$，其中 $e_i \in R^d$，Sku 的 embedding 为 $S(s)=g$，$g \in R^d$，Query 和 sku 的打分计算如下：

$$G(Q(q), S(s)) = \sum_{i=1}^{m} w_i e_i^T g$$

$$w_i = \frac{\exp(e_i^T g/\beta)}{\sum_j^m \exp(e_j^T g/\beta)}$$

其中 β 是 softmax heat 参数。假设 D 表示训练预料，$r(q_i, s_i^+)$ 为正样本，$r(q_i, s_i^-)$ 为负样本，模型优化的 loss 可表示为：

$$\mathcal{D} = \{(q_i, s_i^+, \mathcal{N}_i) \mid i, r(q_i, s_i^+) = 1, r(q_i, s_i^-) = 0 \ \forall s_i^- \in \mathcal{N}_i\}$$

$$\mathcal{L}(\mathcal{D}) = \sum_{(q_i, s_i^+, \mathcal{N}_i) \in \mathcal{D}} \sum_{s_i^- \in \mathcal{N}_i} \max(0, \delta - f(q_i, s_i^+) + f(q_i, s_i^-))$$

④ Negative Sampling

我们采用的是用户点击数据，数据量在10亿级作为正样本。负样本并未使用同 session 未点击的样本，因为搜索手机，展示了小米和华为手机，不能说未点击就是不相关商品。负例分为两部分：random negatives、batch negatives。我们增加了一组超参来调整两者的比例，观察发现 random negatives 越多，召回商品的 popularity 越高，更能吸引用户点击下单，但会降低商品与检索 query 的相关性。

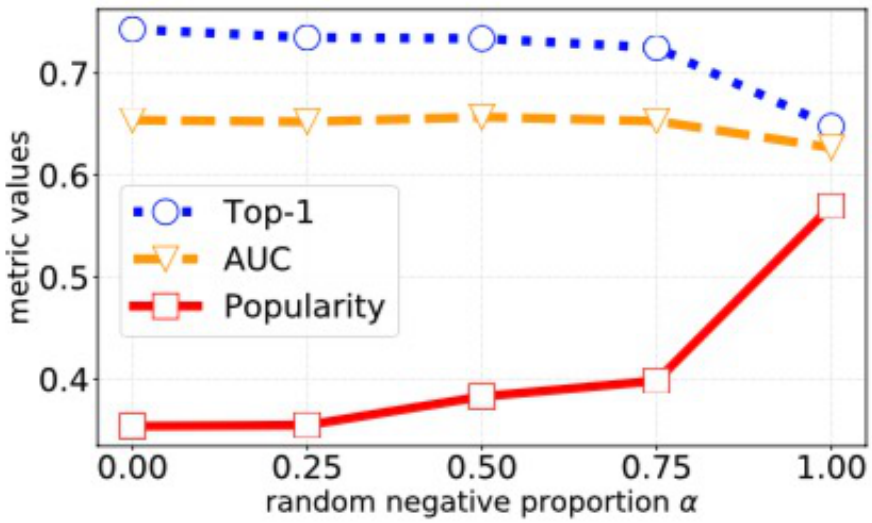


Figure 7: Effect with different mixing ratio of negatives

模型训练算法具体如下：

Algorithm 1 DPSR training algorithm

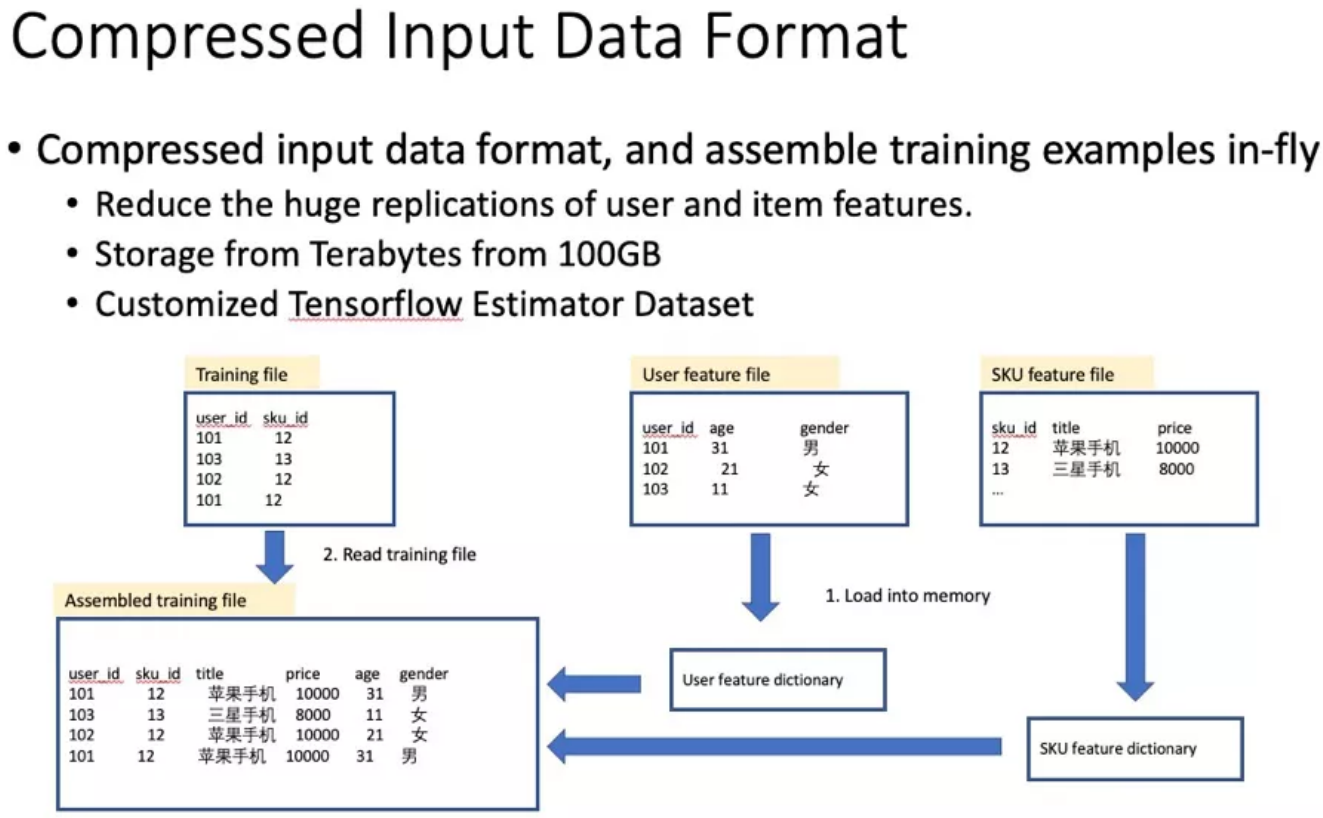
input: Dataset $\mathcal{D}$, batch size b, max number of steps T, mixing ratio α
for $t = 1 \ldots T$ **do**
  Sample a batch of b examples $\mathcal{B} \subseteq \mathcal{D}^+$.
  Sample a set of random negatives $\mathcal{N}^{rand}$ for this batch. Note that all examples in the batch shares this set.
  Compute query head embeddings $Q(q)$ from query tower.
  Compute all item embeddings $S(s)$ for all item s_i in the batch, and in the random negative set $\mathcal{N}^{rand}$.
  Compute loss function value $\mathcal{L}(\mathcal{B})$ for this batch $\mathcal{B}$. The batch negatives $\mathcal{N}^{batch}$ are implicitly computed and included in the loss.
  Update towers Q and S by back propagation.
end for
return query tower Q and item tower S.

3. 训练优化

我们也尝试过更强大的神经网络，如 RNN、transform 等，得到的效果类似或稍好一些。然而一个短延时的模型更适用于工业生产建模，这样可以使用更少的服务器做有效的离线训练和在线服务。

模型系统方面，我们也做了一系列训练优化，简单描述其中的几点：

- 实现 c++ tokenizer，以 custom operator 方式加载到 tensorflow，离线训练和在线服务共用，保证 token 的一致性。
- 训练数据压缩，修改训练数据格式，把共用的特征数据加载内存，训练时展开从而降低数据存储。也便于训练时做负例采样。



- 可伸缩分布式，切分大的 embedding，并将 sum up 放到 ps 以解决 worker/ps 带宽瓶颈。
- 模型 servable 服务，我们将向量检索和 tfs 合成一个服务，不仅减少一次网络访问，降低系统 3-5ms 的平响，而且将模型分片部署，从而可以支持上百个模型同时服务或者 A/B 实验。同时 servable 服务是 cpu 和 gpu 混合部署。

	indexing (sec.)	search (ms)	QPS
CPU	3453	9.92	100
GPU	499	0.74	1422

Table 3: Comparison of GPU and CPU for indexing and serving.

4. 语义检索效果展示

语义检索上线后获得了很好的体验效果，不仅提升了转化，长尾流量降低了近10%的 query 改写率，也就是说用户不需要多次改写 query，就能获得想要的商品结果。

JD.com

03

商品排序

下面介绍下商品排序：

商品排序主要是根据用户的输入对商品进行打分排序。商品排序的传统方法使用 xgboost 等基于决策树的方法从数据中进行学习，但是这些模型通常有成百乃至上千的数值型人工特征，不能有效的从原始特征比如用户历史点击购买数据、商品文本和图像中直接学习。近年来，深度学习在各种应用中验证了从原始特征中学习的有效性，在业界被广泛使用，比如 wide&Deep、DIN 等。下面介绍一个我们在商品搜索排序中尝试的方法。

1. 双胞胎网络

我们的训练数据来自于用户的搜索日志，通过将同一个 session 中用户购买的商品（商品a）和没有购买的商品（商品b）配对起来，并把购买未购买作为最终学习的 label，从而构造了用户查询-商品对训练集。

根据训练数据，我们首先设计了双胞胎网络结构：

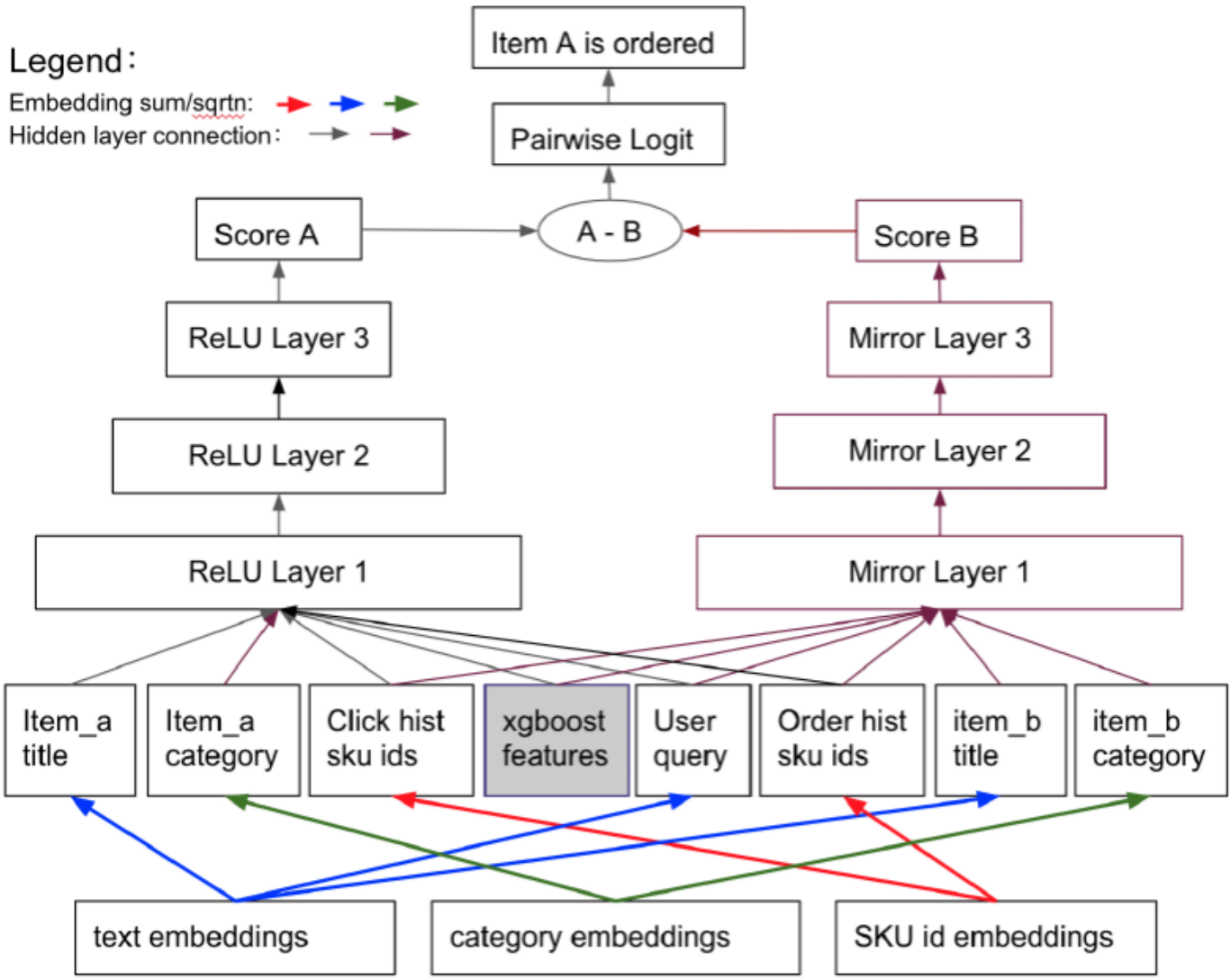


Figure 3: Simplified Siamese ranking model.

双胞胎网络结构有两个共享参数的模块，每个模块分别输入用户、查询和商品特征，每个模块采用 ReLU 作为激活函数，最终层的输出一个分数，两个模块的差值和数据 label 作为交叉熵损失函数的输入。

在特征方面，我们使用以下几种不同类型的特征：

- 数值型特征：包括商品销量、用户购买力和用户是否点过、购买过商品等。
- 文本特征：包括用户输入的查询和商品名称等。
- 用户历史行为：包括历史点击、购买、加购商品 list 等

- 商品、用户 id 等

文本特征可以学习到一定的相关性信息，用户历史行为可以学习到个性化信息，id 类特征我们做了 pretrain。

2. 个性化升级

在第一版双胞胎模型中，我们简单的对用户的历史行为做 sum pooling，但是这样缺乏和搜索商品的交互，无法精准的表示用户的兴趣；为了加强用户的交互，我们升级了模型的结构，用候选商品和用户历史商品做 attention，从而将静态的 user embedding 升级为随 query 和当前商品变化的 user embedding。

我们还加入了 Graph 学习方法对 id 类特征 embedding 进行 pretrain，然后加入到模型训练中。具体方法使用用户的高质量点击行为生成商品 graph，通过 Random Walk 生成训练数据，然后利用 Skip-gram 进行训练，加入 id embedding 可以提高模型离线指标和收敛速度。

	SESSION AUC@5	SESSION AUC
DNN	0.656	0.857
DNN+history_attention	0.667	0.864
DNN+ history_attention+gnn_pretrain	0.675	0.867

3. 时效性优化

值得一提的是，为了增强排序捕捉变化的能力，提升排序的流动性，我们从三个方面：特征时效性、模型时效性、线上预估校准进行了优化。

- **提升特征时效性**：接入商品小时级的点击加购订单等实时信号，训练模型学习实时变化
- **实时在线校准**：根据商品全站的点击订单等实时反馈信号，对模型原来的预测分数及时校准

- **提升模型的更新频率**：优化训练数据生产流程，推动训练平台升级，提升模型训练速度

搜索排序是商品检索最重要的模块之一，我们在个性化、时效性、多目标等方向不断迭代，提升了排序体验，也提升了商品成交量。

04

总结

我们介绍了语义检索召回和商品排序，在京东搜索服务上部署并取得了良好效果。我们还在尝试一些业内其他流行的方法，比如 GNN、KG、MMoE 等方向，也获得了不错的成绩。

文章作者：

王松林、唐国瑜，京东算法工程师。

团队介绍：

京东搜索应用科学部，负责京东商城商品搜索排序算法，覆盖京东主站，京喜，微信一级入口的京东搜索。团队成员有来自国内外大厂，也有来自中清北的优秀毕业生。我们致力于用技术驱动产品，用行业前沿的先进技术落地业务场景；从实际需求出发，用技术解决实际问题，做有用并且有趣的算法，我们也乐于把实践经验通过论文分享给业界。欢迎有技术情怀、有创新活力的你加入我们！

投递方式：

邮件名：姓名-学校/公司-算法工程师，将简历发送至：

wangsonglin3@jd.com

今天的分享就到这里，谢谢大家。

如果您喜欢本文，欢迎点击右上角，把文章分享到朋友圈~~
