wide&deep 在贝壳推荐场景的实践

原创 翟丁丁 郭立星 壳算子 2019-12-18

本文主要向大家介绍 wide & deep 模型在贝壳首页二手房推荐场景中的实践。

"推荐" 对于我们来说并不陌生，已经渗透到了我们生活中的方方面面，比如淘宝的商品推荐，网易云的音乐推荐，抖音的短视频推荐等。在房产O2O领域，同样也需要推荐。无论在哪个推荐领域，推荐系统面临的一个共同挑战是如何同时满足推荐结果的准确性和多样性。准确性要求推荐的内容与用户高度相关，推的精准；多样性则要求推荐内容更加新颖，让用户具有新鲜感。**设计合理的推荐策略，兼顾内容准确性和多样性，提升线上推荐效果**，一直是我们算法同学的工作重点。本文向大家介绍wide & deep 模型的设计理念与基本原理，以及该 模型在我们的业务场景中的实践与效果。本文分享的结构如下：

- wide & deep 模型概述
- wide & deep 模型在贝壳首页二手房推荐场景中的实践
- 阶段总结与未来规划

wide & deep 模型概述

wide & deep模型是Google在2016年发布的一类用于分类和回归的模型。该模型应用到了Google Play的应用推荐中，有效的增加了Google Play的软件安装量。目前wide & deep模型已经开源，并且在TensorFlow上提供了高级API。下面将主要围绕Google发布的论文《Wide & Deep Learning for Recommender Systems》来介绍wide&deep 模型的原理和框架。

文中设计了一种融合浅层模型（wide）和深层模型（deep）进行联合训练的框架，综合利用浅层模型的记忆能力和深层模型的泛化能力，实现单模型对推荐系统准确性和多样性的兼顾。模型框架如图 Figure 1 所示。

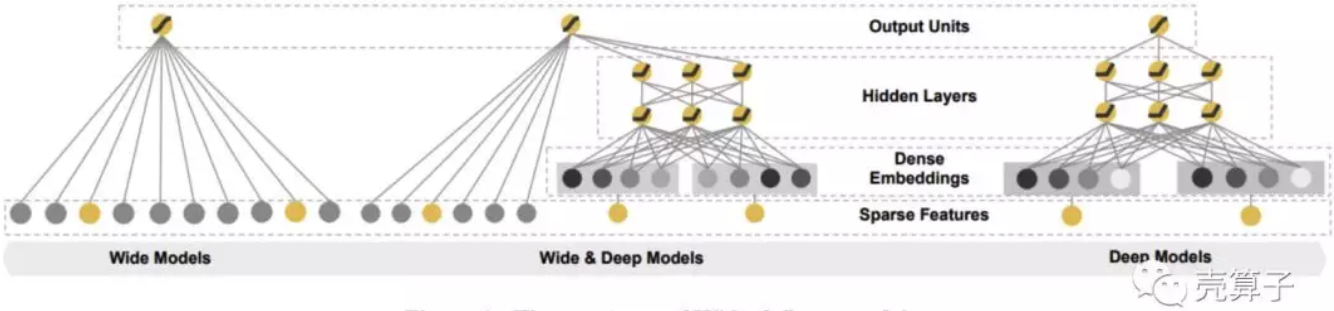

Figure 1: The spectrum of Wide & Deep models

在wide & deep模型中包括两部分，分别为wide模型和deep模型，其中wide部分是一个广义线性模型， 上图中的左侧部分，deep模型是一个前馈神经网络，如上图右侧所示。wide & deep模型的思想来源是，模仿人脑有不断记忆并且泛化的过程，将线性模型（用于记忆）和深度神经网络模型（用于泛化）相结合，汲取各自优势，从而达到整体效果的最优。

wide & deep模型旨在使得训练得到的模型能够同时兼顾记忆（Memorization）与泛化（Generalization）能力：

- Memorization：模型能够从历史数据中学习到高频共现的特征组合，发掘特征之间的相关性，通过特征交叉产生特征相互作用的"记忆"，高效可解释。但要泛化，则需要更多的特征工程。
- Generalization：代表模型能够利用相关性的传递性去探索历史数据中从未出现过的特征组合，通过embedding的方法，使用低维稠密特征输入，可以更好的泛化训练样本中从未出现的交叉特征。

我们分别来看一下wide部分和deep部分:

Wide部分 就是我们熟知的LR线性模型，这部分主要用作学习样本中特征的共现性，达到 "记忆" 的目的。该模型的表达式如下:

$$y = w^T x + b$$

其中，y是预估值，x表示特征向量，w为模型参数， b是偏置项。

特征集合包括原始特征和转换后的特征，其中交叉特征在wide部分十分重要，能够捕捉到特征间的交互，因此通常会对稀疏的特征进行交叉特征转换，定义如下：

$$\phi_k(x) = \prod_{i=1}^{n} x_i^{c_{ki}} \qquad c_{ki} \in \{0, 1\}$$

其中c_{ki}是一个布尔变量，当第i个特征是第k个转换的一部分时为1，否则为0。举个二元特征来说，交叉特征转换(e.g：$AND(gender = female, language = en)$)，只有当$gender = famele$和$language = en$同时为1时，转换结果才为1，否则为0。因此，通过特征之间的交叉变换，便捕获了二元特征之间的相关性，为广义线性模型增加了非线性。

Deep部分是一个前馈神经网络。该部分的输入，通常是一些稠密的连续特征，对于一些高维的稀疏特征（id类等特征），首先会转换成低维且稠密的向量，就是大家常说的embeding vector，然后将这些特征拼接成一个大的稠密矩阵，再喂入第一层。在训练过程中通过优化损失函数不断迭代更新，每个隐藏层执行下面的运算：

$$a^{l+1} = f(W^{(l)}\alpha^{(l)} + b^{(l)})$$

其中l是隐层的标号；f是激活函数，Google的论文中使用的是ReLU。

最后通过 **联合训练** 的方式将 wide 模型的输出和 deep 模型的输出合并到一起。这里要重点区别联合训练和集成学习的差别。集成学习是多模型分别独立训练，最后再将结果进行融合；而联合训练会将wide和deep模型组合在一起，在训练时同时优化所有参数，并且进行加权求和，根据最终的loss计算出gradient，反向传播到Wide和Deep两部分中，分别训练自己的参数。也就是说，wide & deep 模型的权重更新会受到 wide 侧和 deep 侧对模型训练误差的**共同影响**。在论文中，wide部分是使用L1正则化的**Follow-the-regularized-leader**(FTRL)算法进行优化，deep部分使用的是**AdaGrad**完成优化。

到此对 wide & deep 模型结构的讲解就结束了，它的核心思想是结合线性模型的记忆能力和 DNN 模型的泛化能力，从而提升模型的整体性能，也因此能够同时兼顾推荐的准确性和多样性。

wide & deep 模型在贝壳首页二手房推荐场景中的实践

我们的推荐系统中，主要分为两大阶段：召回和排序。如图 Figure 2 所示，其中召回阶段负责从数据存储系统中筛选出与用户相关的一些房源，排序阶段负责对这些召回的房源进行精准的排序和打分。

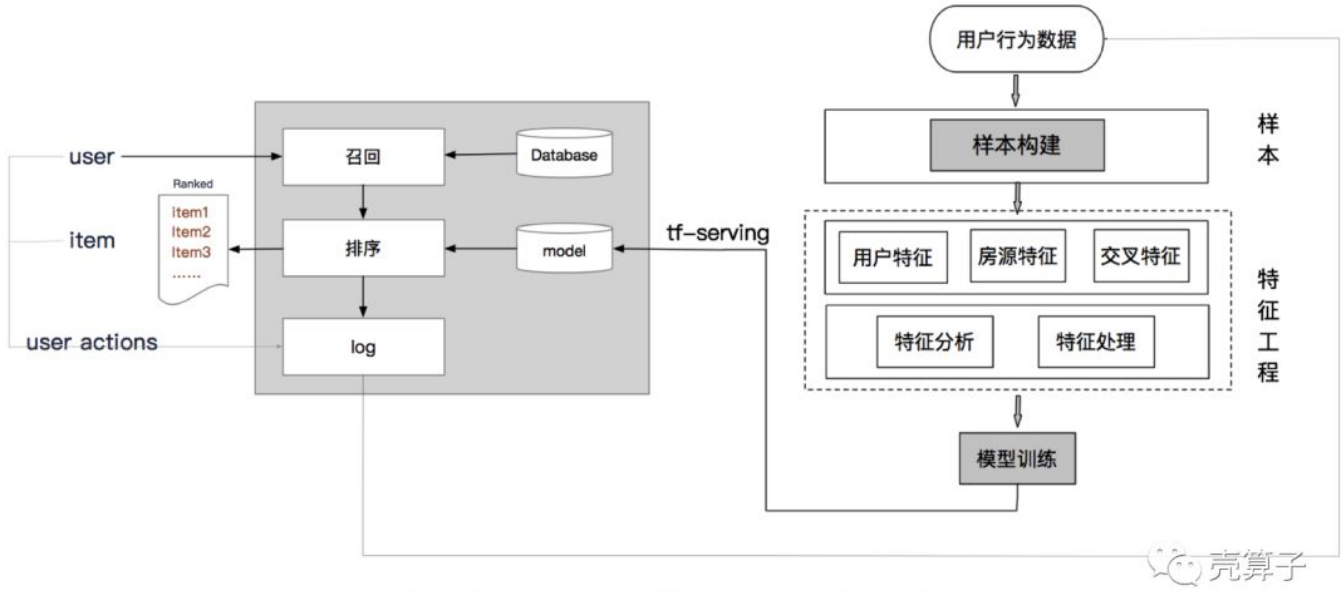

Figure 2: Architecture of the recommendation system

我们将wide&deep模型应用在了贝壳首页推荐场景下的排序阶段，优化目标定义为推荐 点击率。排序阶段模型的搭建主要包括了样本构建、特征工程、模型离线训练及线上化，其中样本与特征是整个机器学习中最重要的两个环节，直接决定了模型效果的上限。接下来分别介绍下这几个部分：

样本构建

推荐排序模型的目标是预测用户是否点击，我们的训练样本采样自用户在首页场景下对房源卡片的浏览和点击数据。

原始样本是一个三元组，包括userid，itemid和label。其中label的定义如下：

- 正样本：用户点击的房源。
- 负样本：用户点击最大位置以上曝光未点击的房源；从未点击的用户部分曝光未点击的房源。

label为1时，代表用户点击，为0时代表用户曝光未点击。

在首页场景下的正负样本比例在1:12，为了解决正负样本的不均衡问题，我们对负样本进行下采样，将正负比例控制在了1:8。在离线训练时，我们也尝试对于商机样本进行一定程度的加权，使得模型能够更充分的学到这部分样本的特征。

此外，我们还对训练样本进行了清洗，去除掉 **Noise样本**，这里的Noise样本指的是特征值近似或相同的情况下，分别对应正负两种样本。在我们的业务场景下，用户在不同时间对同一房源可能会存在不同的行为，导致采集的样本中有正有负，而特征只存在微小的差异，模型是很难学到微小的特征差异对结果带来的改变的，此类样本送入到模型中会产生 "歧义"。因此，我们将Noise样本进行清洗，去掉冲突，减少对模型的干扰。

特征工程

特征工程主要包括特征构建，特征分析，特征处理，以及特征选择等。特征构建也主要是围绕用户和房源这两个角色构建相关的特征。我们主要用到了以下的特征：

- **用户特征**：注册时长、上一次访问距今时长等基础特征，最近3/7/15/30/90天活跃/浏览/关注/im数量等行为特征，以及画像偏好特征和转化率特征。
- **房源特征**：价格、面积、居室、楼层等基础特征，是否地铁房/学区房/有电梯/近医院等二值特征，以及热度值/点击率等连续特征。
- **交叉特征**：将画像偏好和房源的特征进行交叉，主要包含这几维度：价格、面积、居室、城区、商圈、小区、楼层级别的交叉。交叉值为用户对房源在该维度下的偏好值。

在构建完特征，拿到一份数据集后，需要先对数据进行EDA分析，包括查看数据的统计分布、类别概况与取值范围等。对原始数据有了初步的认识与了解后，我们会进行特征处理将特征转化为更容易被模型学习的数据形式，主要的特征分析和处理过程如下：

缺失值与异常值处理：首先我们会进行特征的缺失值分析，分析其缺失、异常、为0的比例，根据特征的类型（离散/连续）与缺失的比例，进行不同方式的缺失值填充。此

外，通过对样本特征进行EDA分析可知，有些特征存在异常值，导致整体的数据分布出现严重偏差，这里通过四分位距（IQR）对异常值进行检测，并将异常值进行截断。

等频分桶处理：对于一些连续特征，例如价格、面积等，我们对其分布进行画图分析。如图 Figure 3 所示， 这里以价格特征为例：

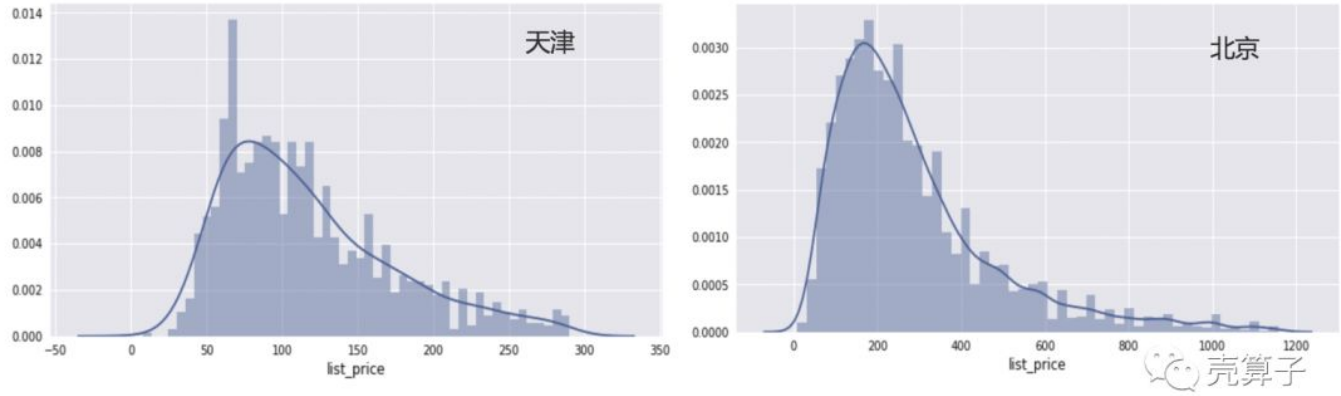

Figure 3: Distribution of continuous features

通过画图分析可知，价格整体呈长尾分布，这就导致大部分样本的特征值都集中在一个小的取值范围内，使得样本特征的区分度减小。因此，我们将这类连续特征进行等频分桶，保证样本的分布在每个区间是均匀的，从而保证样本之间的区分度和数值的稳定性。

此外，不同城市下的分布区间存在较大差异，如上图中，天津的价格区间主要是分布在50-150万，北京的价格区间主要是分布在100-350万，如果对整体的训练集做等频分桶，会导致不同城市下的特征存在偏差。针对这一问题，我们做特征处理时，是分城市进行处理的，从而保证每个城市下的特征都是均匀分布的。

归一化处理：对取值较大的连续值特征使用了最大最小归一化方法，将特征值都转化为0-1之间的数值。该方法能够使得特征之间拥有统一的标准，提升特征的区分度，并且将数据归一化到0到1之间的数据，能够加快模型的训练速度，实验也证明模型效果确实有显著提升。

低频过滤：对于离散型特征，我们对其取值分布进行统计，图 Figure 4 中，以居室和楼层特征为例：

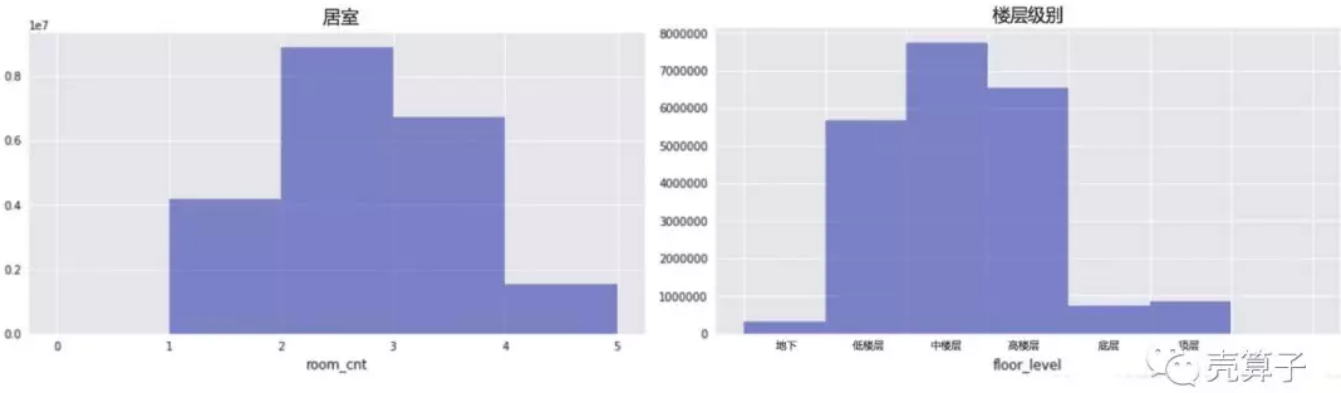

Figure 4: Distribution of category features

通过对居室和楼层级别进行分析可知，卧室数量大部分为两居、三居。中楼层的房源最多，而底层和顶层的房源非常少。通过对此类特征的统计，我们能够清楚的看出特征值

的频率分布情况，有些类别的取值比较低频，存在分布不均的情况，因此我们将这部分特征进行了低频过滤处理，将出现频次小于阈值的值归为一类。

embedding：对于一些高维稀疏的id类特征，例如小区，商圈id等，可以作为deep模型的embedding输入，通过embeding层转换成低维且稠密的向量。降维后的特征具有较强的“扩展能力”，能够探索历史数据中从未出现过的特征组合，增强模型的表达能力。

这里需要注意的是，一些id类特征维度很高，呈长尾分布，可以将这部分特征进行低频过滤，降维后再放入embedding中。

模型离线训练与线上化

离线训练 经过上述的样本构建与特征工程，生成了训练集，作为wide & deep的训练样本。我们使用的模型结构如图Figure 5所示，采用了三层的网络结构。

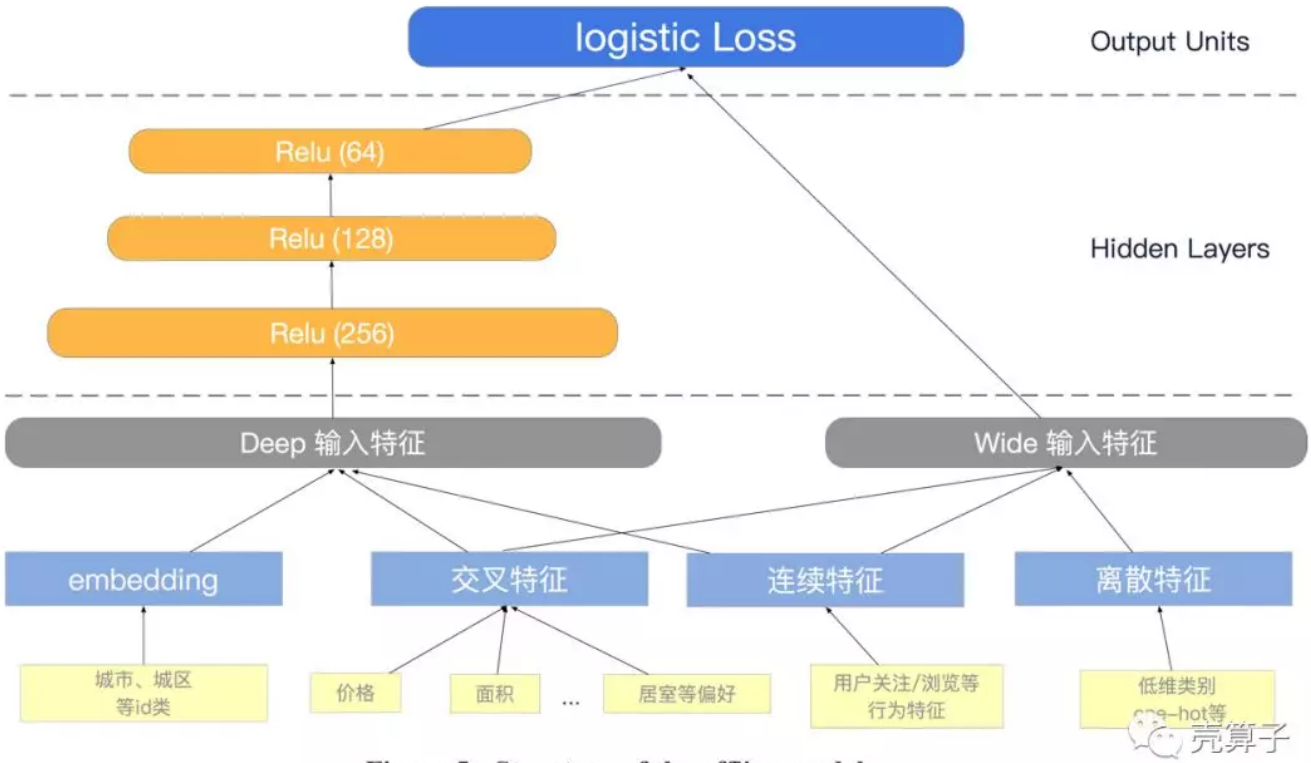

Figure 5: Structure of the offline model

image

图中的Hidden Layers为三个全连接层，每个layer后面连接Relu激活函数，训练过程中，wdl是采用联合训练的方式，将wide和deep模型组合在一起，在训练时同时优化所有参数，并且在训练时进行加权求和，根据最终的loss计算出gradient，反向传播到Wide和Deep两部分中，分别训练自己的参数。

模型训练过程中，我们将数据集按照时间顺序切分，采用7天的数据作为训练集，1天的作为测试集。

在训练期间，wide侧的输入包含了一些交叉特征、离散特征及部分连续特征。deep侧的输入，主要是一些稠密的连续特征，我们也尝试了将城市、城区、商圈等id特征进行embedding，但没有取得效果上的提升，猜测是由于模型对这部分特征没有学到很好的表达，无法发挥出embedding的优势，目前我们还在研究和探索好的embedding特征，也期待能和大家一起讨论。

模型调优 为了防止模型过拟合。我们加入了dropOut 与 L2正则；为了加快模型的收敛，引入了Batch Normalization（BN）；并且为了保证模型训练的稳定性和收敛性，尝试不同的learning rate（wide侧0.001，deep侧0.01效果较好）和batch_size（目前设置的2048）；在优化器的选择上，我们对比了SGD、Adam、Adagrad等学习器，最终选择了效果最好的Adagrad。实验证明，通过参数的调整，能够明显的影响模型的训练效果。

线上化 模型训练和验证后，需要将它进行线上化，我们采用了TensorFlow Serving服务器。对于线上特征和离线特征的一致性问题，采取了redis存储的方式，将离线特征处理的参数存储在redis中，并且在上线之前，我们会进行特征一致性检验。为了提升模型的预测性能，我们使用多线程并行化的方式构造tf-serving请求，最终能在10ms左右返回120个房源的预测结果。

线上效果

我们在10月中旬上线第一版wdl排序模型，线上效果如下：

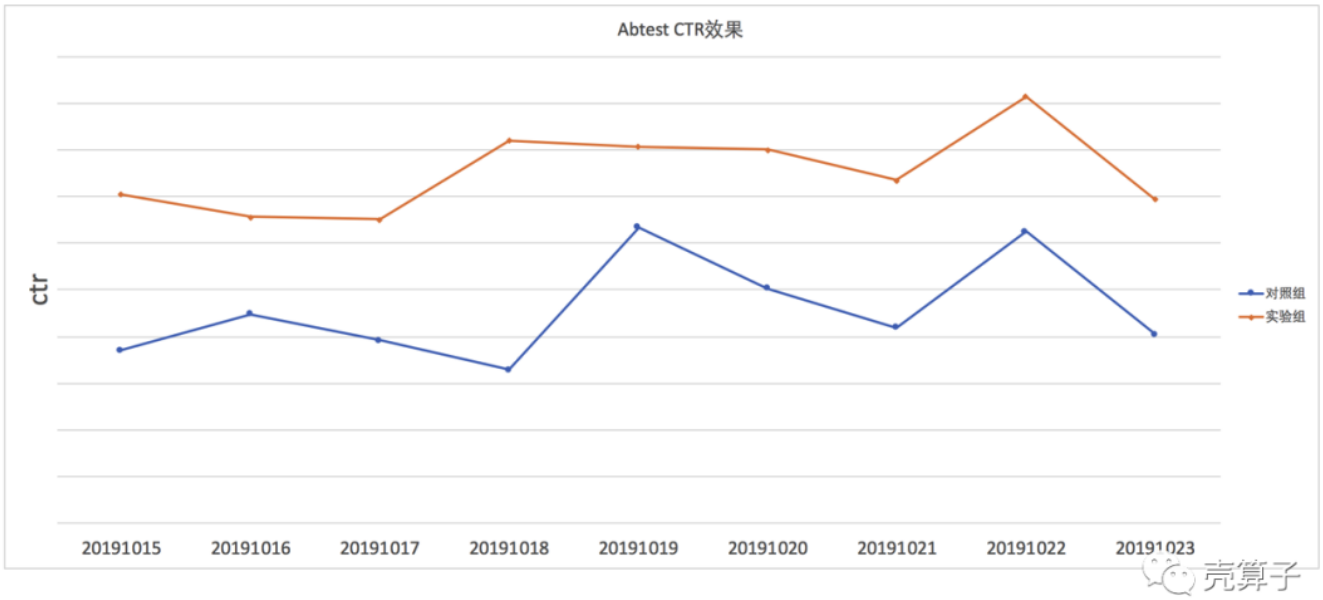

Figure 6: Promotion of online CTR

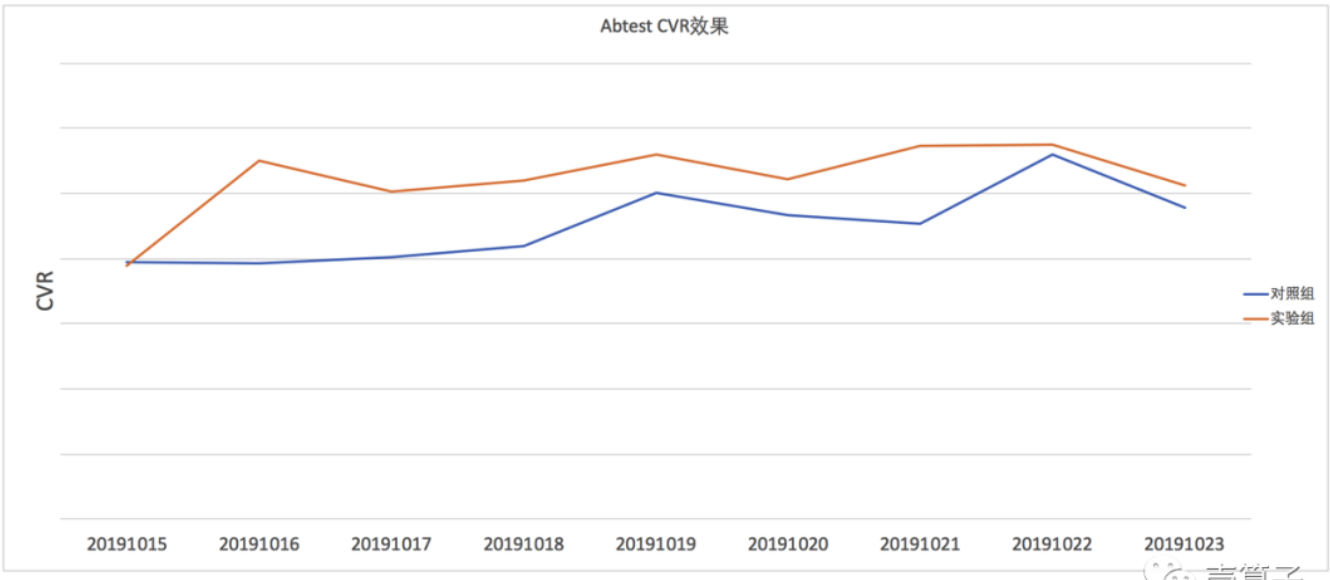

Figure 7: Promotion of online CVR

image

Figure 6 和Figure 7 分别对应线上CTR 和CVR的提升效果。第一版模型上线后，对比对照组CTR相对提升6.08%，CVR相对提升15.65%。目前我们正在探索新的特征，以及在样本和特征工程上尝试不同的方法，进行模型的二期优化，提升模型的离线auc，争取给线上效果带来进一步的提升。

阶段总结和未来规划

本文主要介绍了深度学习模型 wide&deep 在贝壳排序场景中的实践，包括根据业务场景构建样本、特征，以及搭建模型等工作，在线上也取得了正向收益。

目前我们正在进行模型的二期优化，也在的不断总结和尝试。对于接下来的优化方向，主要总结为以下几个方面：

- 样本精细化采样代替随机采样（包括分城市、分用户群体的采样）。
- 为wide侧挖掘更多有区分能力的特征，deep侧加入有效的embedding特征。
- 引入实时特征，确保房源和用户特征最新和最全。
- 优化网络结构，提升网络学习能力。
- 优化模型的训练效率，加快线上模型的更新速度，提升线上效果。
- 尝试gAUC、MAP等评估指标，保证离线效果提升与线上效果提升的一致性。

此外，深度学习模型在推荐系统中的应用还有更多价值等待我们一起去发掘，这条算法之路，还是十分漫长的，也希望能和大家一起讨论，共同学习。

参考资料

[1] https://arxiv.org/pdf/1606.07792.pdf

[2] https://www.tensorflow.org/serving

作者介绍

- 翟丁丁，2018年校招加入贝壳找房，主要从事推荐算法相关工作。
- 郭立星，先后在去哪儿网、一起作业网从事算法相关工作，现就职于贝壳找房数据智能中心，担任资深算法工程师，专注推荐算法相关工作。

文章已于2019-12-30修改