案例分享 | TensorFlow 在贝壳找房中的实践

原创 贝壳找房技术团队 TensorFlow 2月12日

收录于话题 #案例展示 60个

文 / 潘昊、陈嘉远和袁彬，贝壳找房技术团队

贝壳找房作为行业领先的房产服务互联网平台，通过开放数据资源和技术能力，聚合和赋能全行业的服务者，打造产业互联网下的“新居住”品质服务生态，致力于为全国家庭的品质居住提供全方位服务连接，涵盖二手房、新房、租房、装修和社区服务等众多类目。

今天我们主要针对一个打算购买二手房的用户，来看一下 ta 来到贝壳 App 中所经历的一切，以及平台背后进行的策略优化。首先，用户会现在 App 上浏览一些 ta 感兴趣的房源。接着，当 ta 看到一套合适的房源时，会跟经纪人进行线上沟通，从而产生商机。随着聊天的深入，ta 对这个房子的兴趣越来越大，于是决定是时候去实地看一下这个房子了，就和经纪人约了线下带看。最后，ta 很满意这个房子，在贝壳达成了成交，并且在后续经纪人的帮助下成功过户，完成了购房。

图 1 用户在贝壳 App 上的行为流程图

这个流程囊括了用户在贝壳 App 上的几个重要环节：浏览、商机、带看、成交和售后。每个环节我们都有专门的团队来负责优化，其中，AI 选房团队通过结合 DNN 学习到静态信息与 LSTM 学习到的动态信息，帮助经纪人在作业过程中聚焦好房，极大的提高了去化率；推荐团队，通过实践 Embedding、WDL、DeepFM 等深度模型，持续提升场景的点击和商机转化效果，改善用户体验；图像团队通过提供一体化的 OCR 流程，使得交易流程得到改善的同时还大大地降低了人工成本。

而这些好效果的背后，都有两个共同的因素：深度学习和 TensorFlow 框架。随着数据量和算力的增加，选择深度学习早已顺理成章，而我们选择 TensorFlow 作为深度学习的框架，主要有以下几个原因：

1. 提供了很多拿来即用的 SOTA 模型，快速解决业务问题的同时也有效地提升了模型的迭代效率。
2. 支持 python、与 numpy 完美结合，降低数据预处理门槛。
3. 提供高性能、灵活的模型服务框架——TF serving，使得训练和线上预测可以无缝衔接。
4. 提供强大的可视工具——TensorBoard，帮助我们快速发现模型问题。
5. 完善的社区，遇到问题可以快速找到解决方案。

AI 选房

作为一家房产服务互联网平台，如何在海量房源中选出能够快速成交的房源是对平台和经纪人来说都是一件非常重要同时具有挑战的事情，它可以提升经纪人的工作效率和业绩，加速平台中房源的成交。

人工选房流程

选快速成交房源的工作在线下门店中是一直存在的，普遍做法是开会由门店中的经纪人提供经验投票选出。但是，这个方法存在如下的问题：

- 经纪人时间有限，开会选房成本高。
- 经纪人带有主观性。
- 房源数量过多，经纪人无法了解所有房源质量。

AI 选房模型介绍

为了帮助经纪人和平台更高效客观准确地选出能够快速成交的房源，我们提出了 AI 选房，将深度学习模型应用到选房中。

我们使用过去 90 天的房源数据训练深度学习模型，预测当前房源在未来 14 天的成交概率。

在特征选择上，从影响房源成交因素中选择了 6 大方向上百维特征，具体如下：

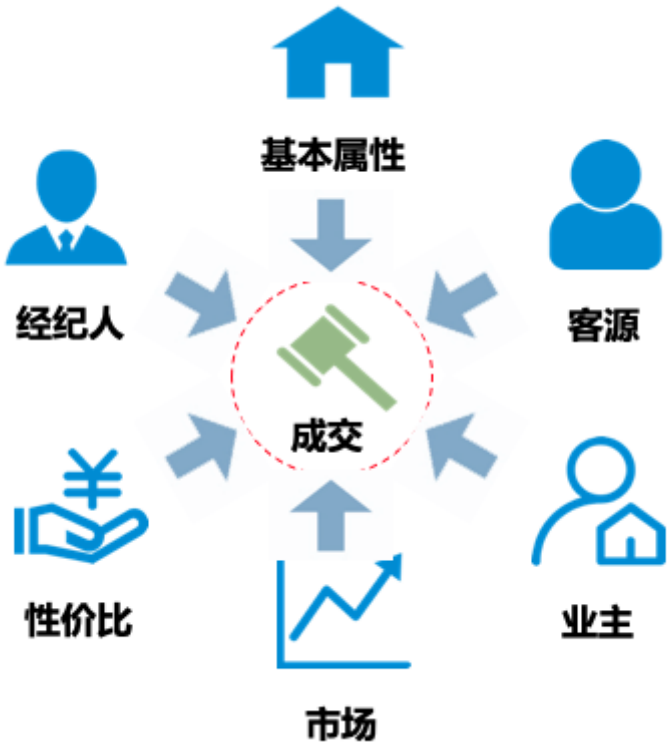


图 2 AI 选房模型特征

在模型上，我们同时采用了 DNN 和 LSTM 两种深度学习模型。其中，DNN 处理房源属性、价格等静态特征，LSTM 处理经纪人作业、客源关注量等时序特征。最终，将两种模型的输出进行融合后得到房源的成交概率。

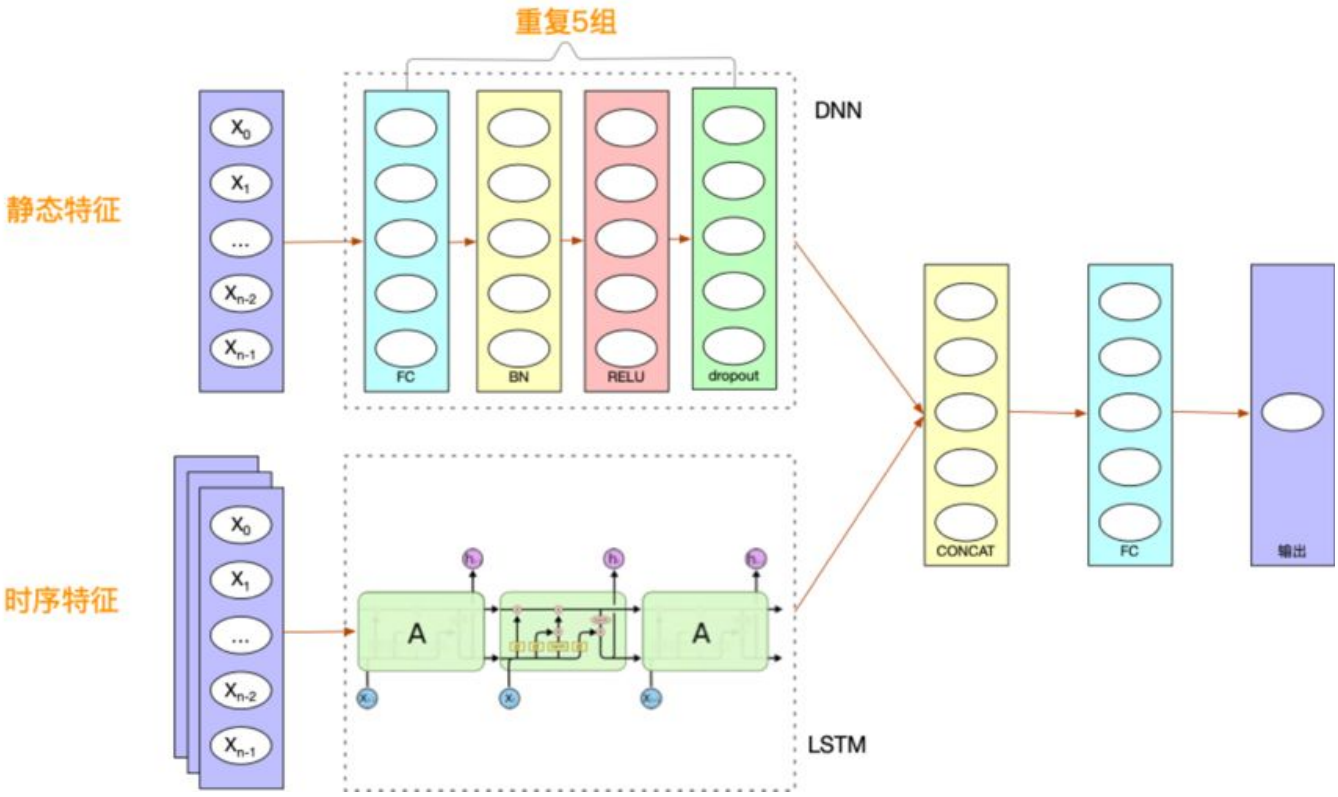


图3 AI 选房模型结构图

AI 选房上线后，选房效果大大超过人工选房的效果，准确率相比于人工提升了 56%，促进了更多房源的快速成交。同时，将经纪人从选房中解放，有效帮助经纪人盘点当前维护的房源，指导经纪人作业。

智能推荐

在贝壳，由于推荐场景、业务和物料类型众多，因此孕育而生了智能推荐平台。通过建设平台能力来快速支持和满足业务上的需求。推荐场景大大小小有上百个，比如一些典型的场景，如首页推荐、详情页推荐、搜索少无结果推荐等流量大的核心场景。也有如经纪人与用户聊天过程中的房源推荐、给用户推荐其关注房源相似的房源等。

我们把一个推荐场景的策略划分为业务策略和算法策略两部分。业务策略包括展示提权、同质打散、业务干预等。算法策略则包括黑名单、召回、融合、排序和理由这 5 个部分，如图 4 所示。

图 4 智能推荐平台策略架构图

其中，算法策略中各部分的具体含义如下：

- 黑名单。主要是对物料的过滤策略。
- 召回。可以理解为与搜索技术中心的匹配（match）一样，负责快速获取与此次推荐有关联的物料。
- 融合策略则是对多路召回结果的一次合并（merge 或理解为粗排）。
- 排序。就是通常理解意义上的精准排序（rank）。
- 理由。主要是为推荐物料生成展示给用户的推荐理由，增强推荐的可解释性。

我们在召回和排序中广泛使用了深度学习模型，下面来看一下我们是如何在这两部分中实践 TensorFlow 深度学习框架的。

召回策略

在召回策略中，我们通过 PySpark来进行热门、ItemCF、MF 等离线候选集（召回的中间或最终结果）的生成；通过 ES（ElasticSearch）来实现基于内容、用户偏好、线性加权等规则或简单模型的召回；而像 FM、Embedding 等复杂模型的召回，我们则是通过 TensorFlow 框架来实现。由于篇幅问题，这里主要讲一下我们如何得到以及利用二手房源 Embedding 进行召回的。

生成二手房源 Embedding的过程中我们采用 word2vec 的方式来得到房源的 Embedding，其中主要经历了两个版本的迭代。

V1 版：基于 word2vec 模型的房源 Embedding

第一个版本我们使用的是经典的 word2vec 模型，利用用户对二手房的行为来构建序列，一条序列就是一句话，而每一个房源就是句子中的一个词。并使用过采样（Oversampling）的方式，对不同类型的序列进行加权，使得各类序列得到的样本量比例相同。

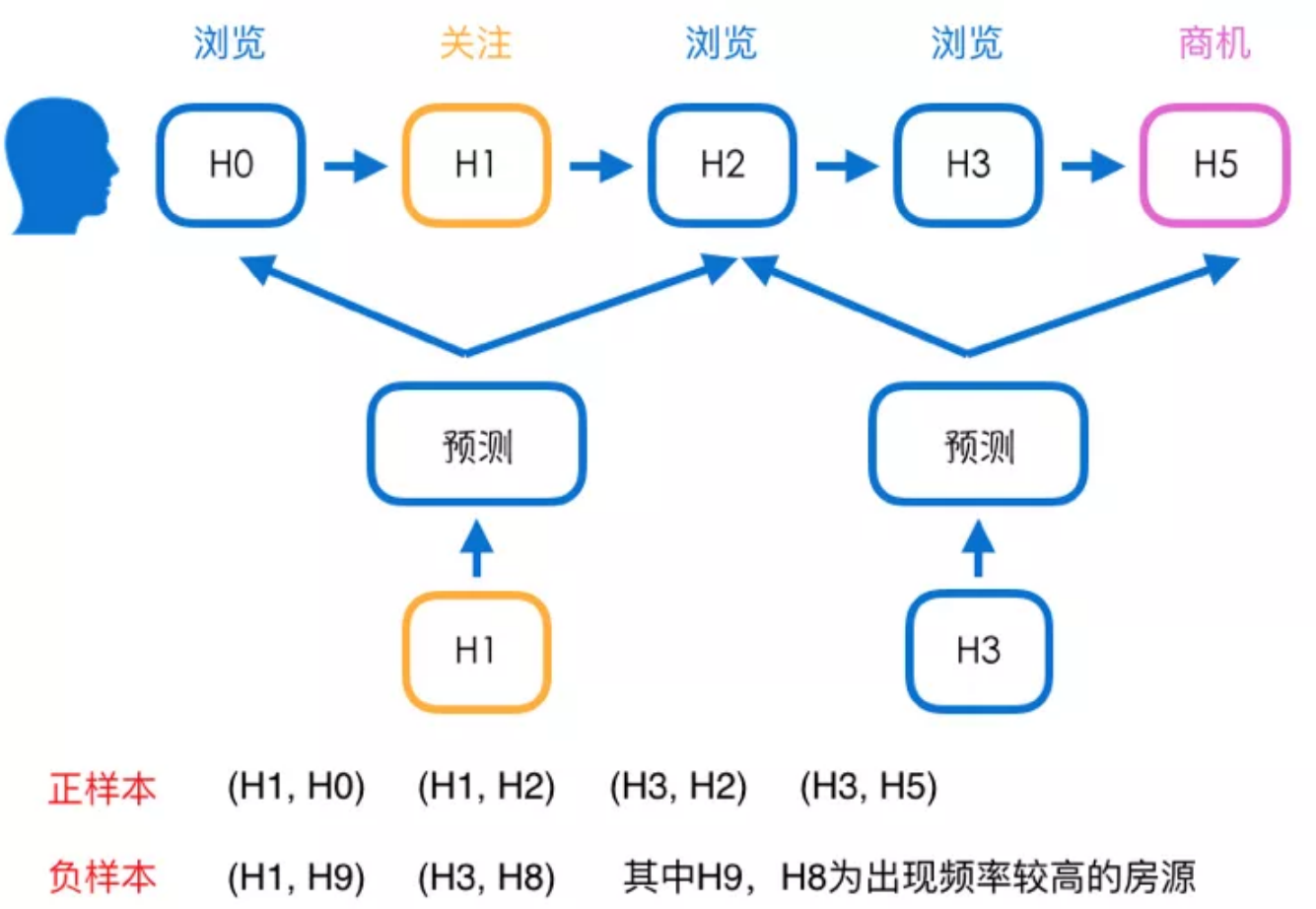


图 5 房源 Embedding V1 版样本构建示意图

由于 Embedding 是通过无监督的方式学习而来的，因此我们采用可视化的方式来进行离线的定性评估。这里，我们通过计算房源间两两 Embedding 的余弦相似度，并结合一般业务意义上理解的相似特征进行对比观察。如图 6 所示是在北京这个城市下，卧室数和小区间距离与相似度的关系。其中，卧室数部分横纵坐标都是卧室区间，色块越偏暖色说明相似度越高（越偏冷色相似度越低），可以看到随着卧室数差异的增大，相似度是越来越小的；小区距离部分，横坐标是小区间的距离，纵坐标是相似度，随着小区间距离的增加，相似度是先降低后升高的（城市通常为放射状发展）。可以看出，通过无监督得到的 Embedding 已经学习到了一般业务意义上的相似。

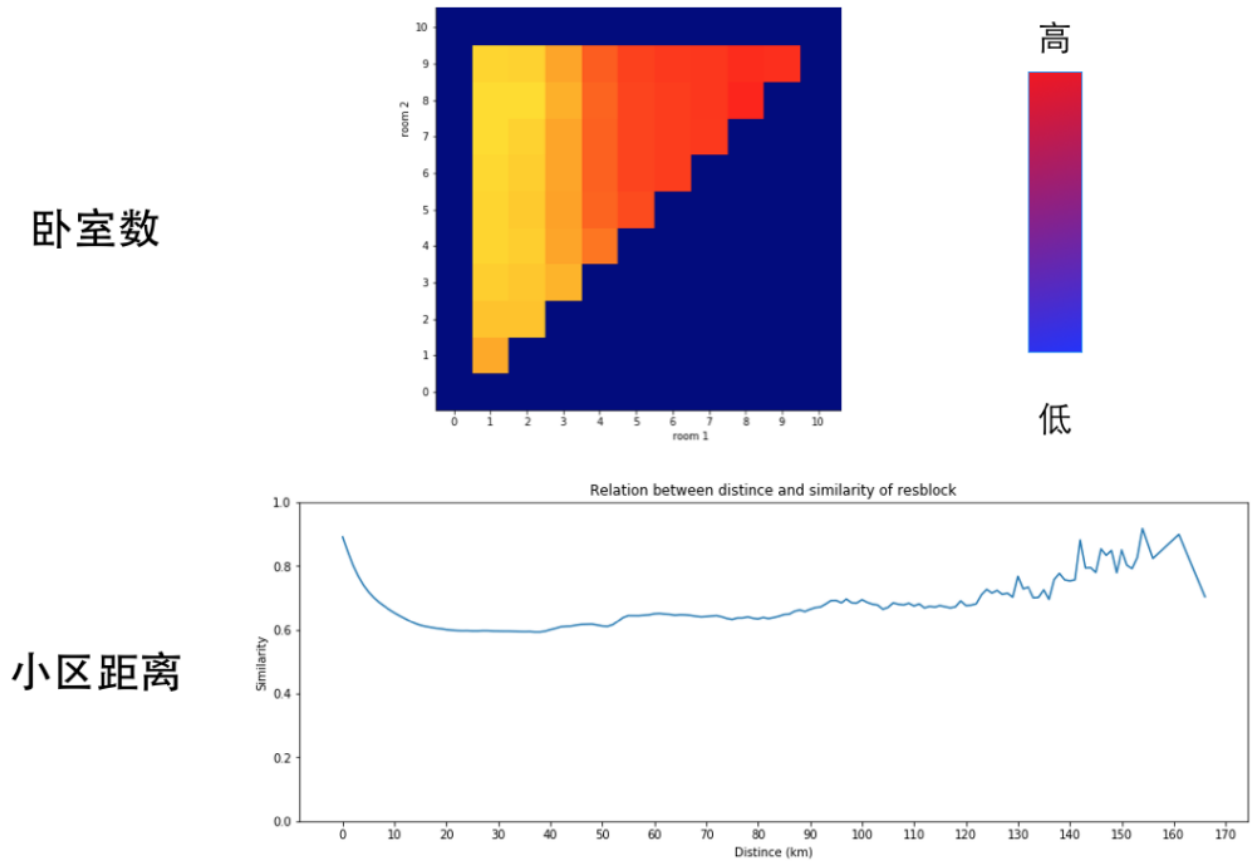


图 6 房源 Embedding V1 版可视化分析

通过城市粒度的可视化分析，我们也发现了一些问题。如图 7 所示，在样本量较少的城市，不同卧室数、小区间距离在相似度上反应的差异就没那么明显了。

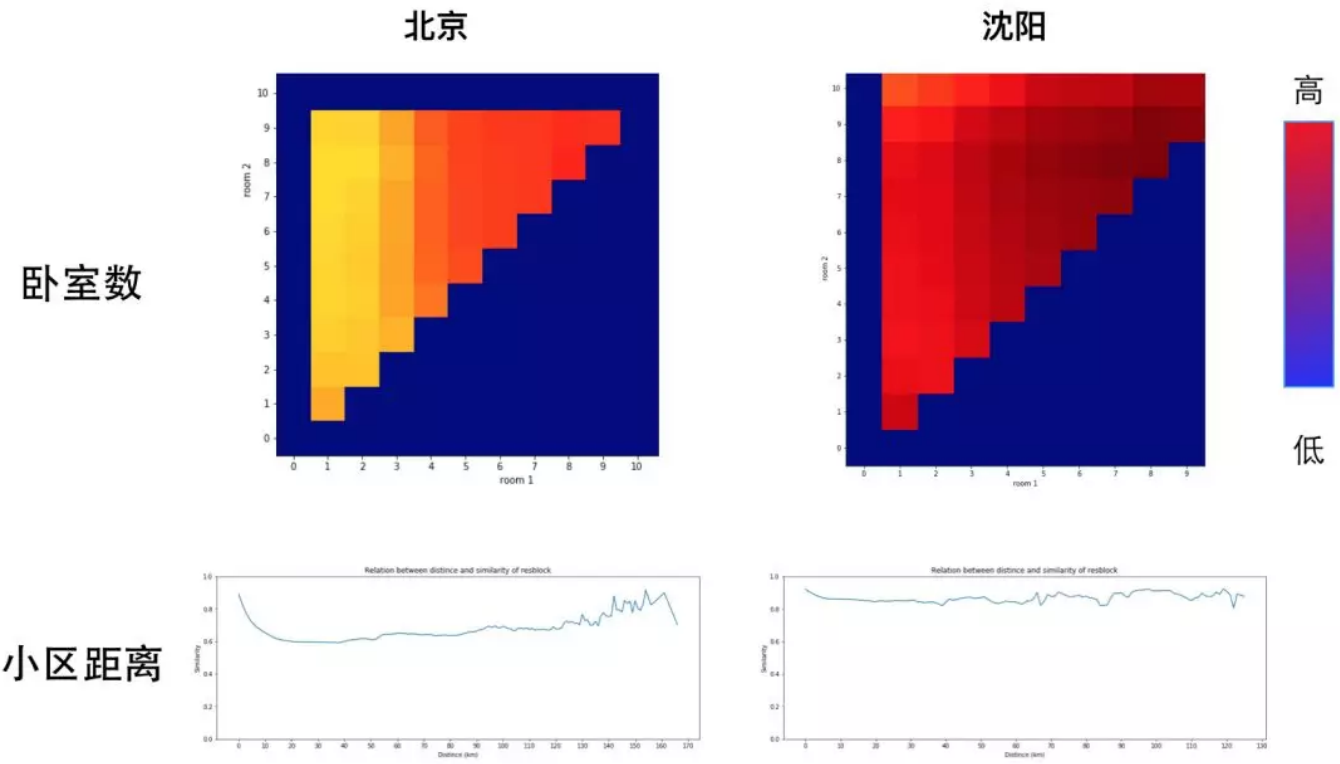


图 7 房源 Embedding V1 版不同城市可视化对比

我们在线上也进行了 AB 实验，相比于基于内容线性加权的召回策略，基于 Embedding 相似度的召回策略在详情页场景表现超过预期，相对提升了 27.7%。

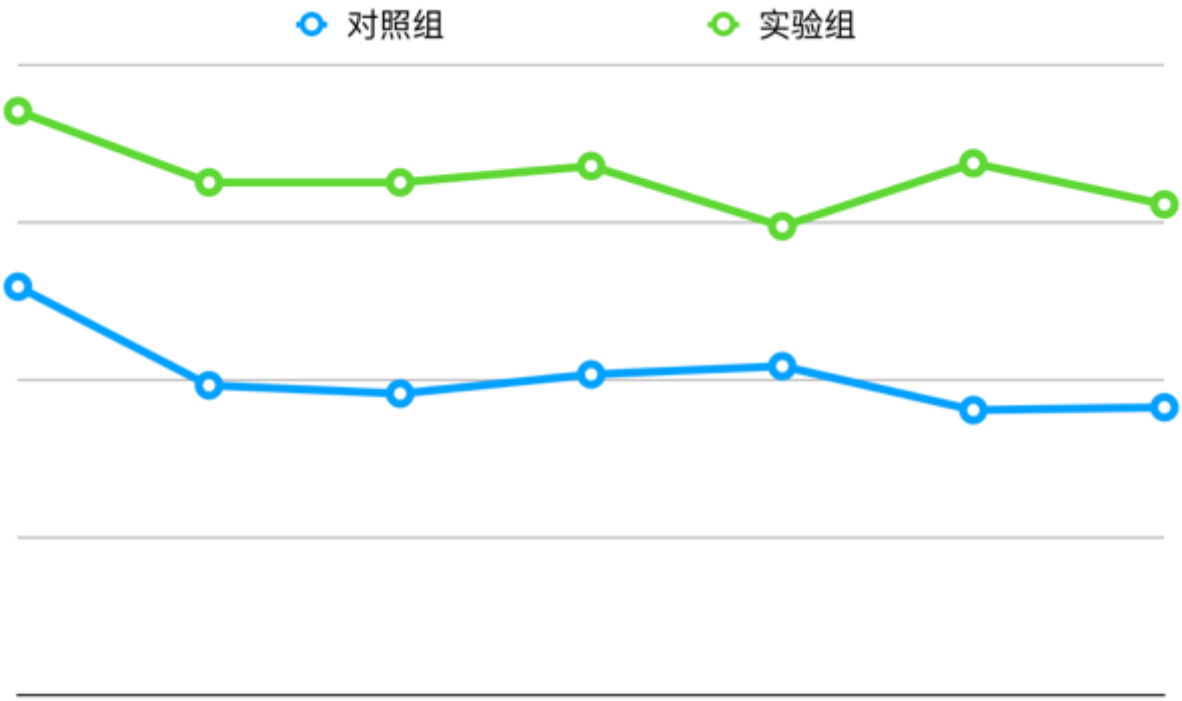


图 8 房源 Embedding V1 版线上 AB 实验

V2 版：合理的样本扩充

针对部分城市样本量不足的问题，我们参考 Airbnb[1] 论文中的方法。通过 TensorFlow 实现了 word2vec，并对损失函数进行了改进，增加了全局上下文和同城负采样。其中，全局上下文补充了正样本，而同城负采样则是补充负样本。

V1

$$\underset{\theta}{\operatorname{argmax}} \sum_{(l,c)\in\mathcal{D}_p} \log \frac{1}{1+e^{-\mathbf{v}'_c\mathbf{v}_l}} + \sum_{(l,c)\in\mathcal{D}_n} \log \frac{1}{1+e^{\mathbf{v}'_c\mathbf{v}_l}},$$

V2

$$\underset{\theta}{\operatorname{argmax}} \sum_{(l,c)\in\mathcal{D}_p} \log \frac{1}{1+e^{-\mathbf{v}'_c\mathbf{v}_l}} + \sum_{(l,c)\in\mathcal{D}_n} \log \frac{1}{1+e^{\mathbf{v}'_c\mathbf{v}_l}} + \log \frac{1}{1+e^{-\mathbf{v}'_{l_b}\mathbf{v}_l}} + \sum_{(l,m_n)\in\mathcal{D}_{m_n}} \log \frac{1}{1+e^{\mathbf{v}'_{m_n}\mathbf{v}_l}}$$

图 9 房源 Embedding V1 版与 V2 版优化目标对比

图 10 是优化前后的可视化结果。可以看到 V2 版在样本量较少的城市也能学习得很好。

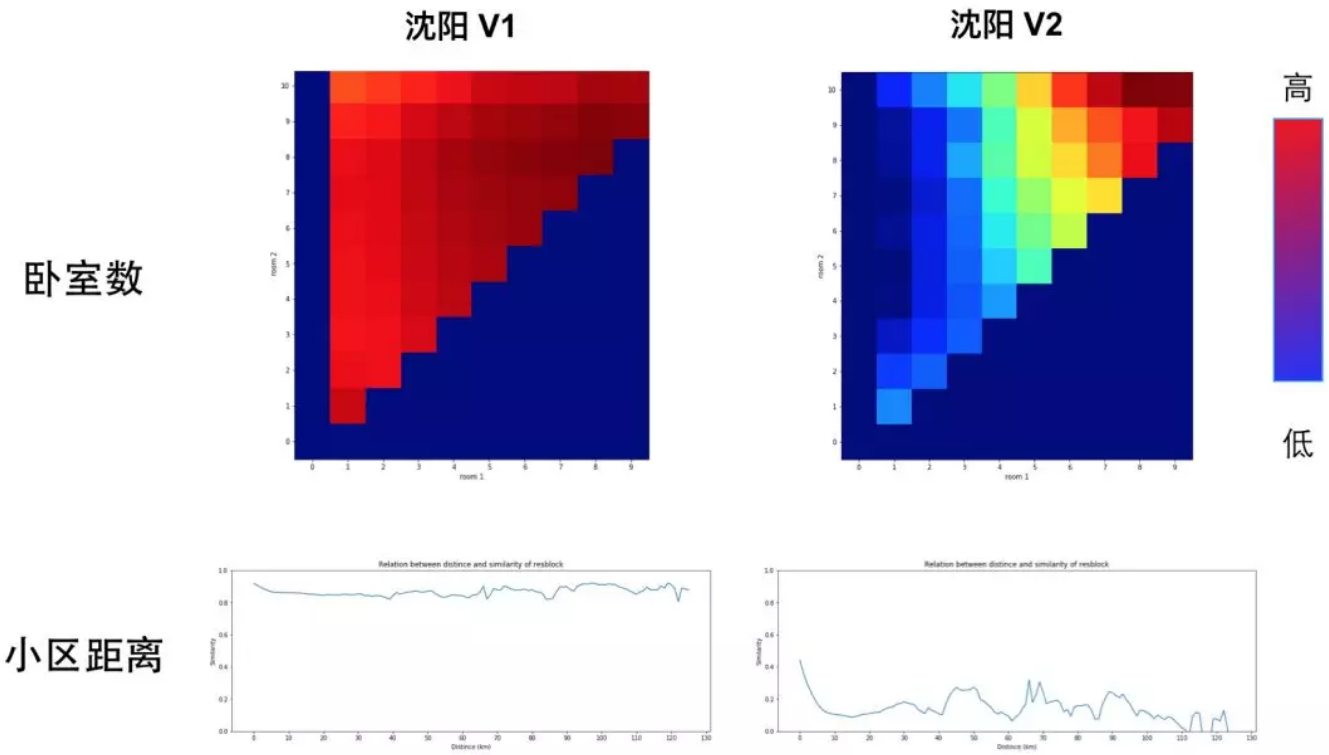


图 10 房源 Embedding V1 版与 V2 版在沈阳的可视化对比

最后，我们在线上对比了 V1 和 V2 版的效果，实验组用 V2 替换 V1 版，相比对照组，相对提升了 6.96%。

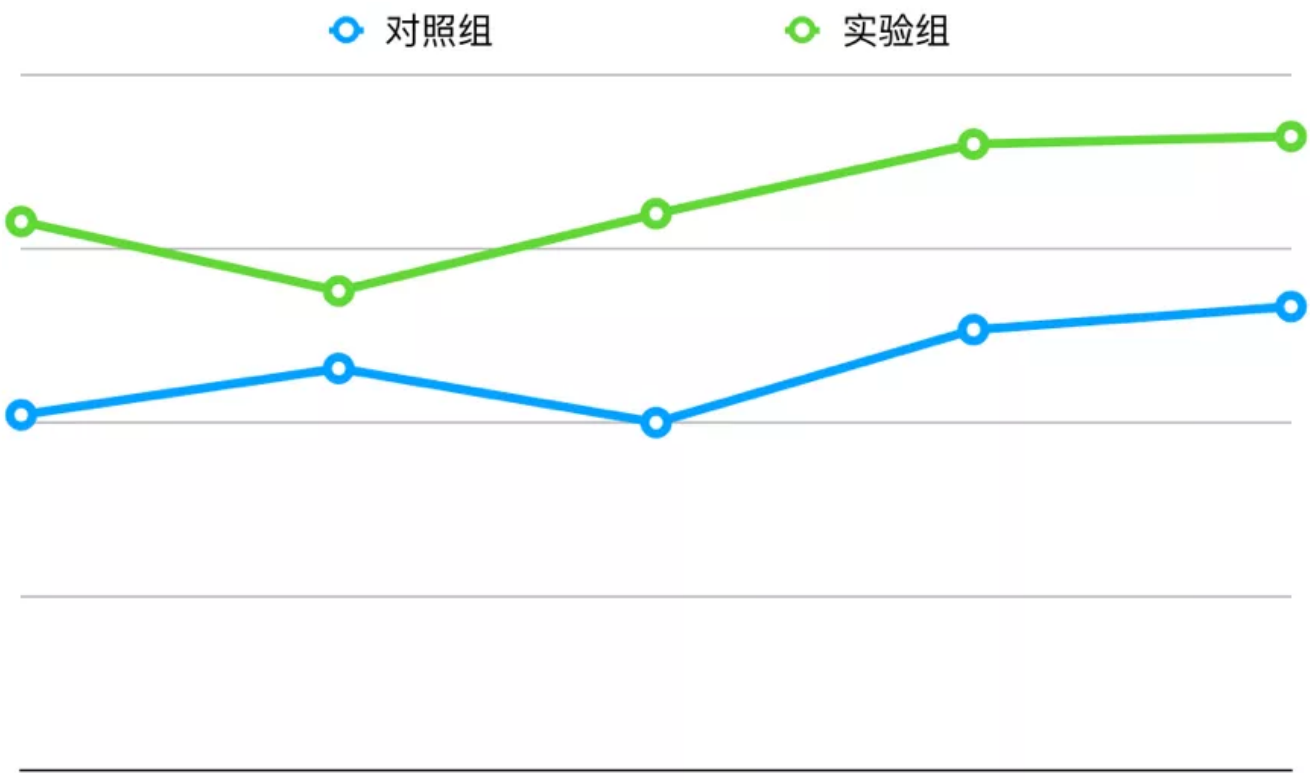


图 11 房源 Embedding V2 版与 V1 版线上 AB 实验

排序模型

一个公司的目标会随着它的发展阶段而改变，推荐系统也是如此。最初我们的目标是提高 pCTR（卡片点击次数 / 卡片曝光次数），因此我们在各个核心场景选择排序模型的时候，都是选择 LR、WDL[2]、DeepFM[3] 等以 pCTR 为优化目标建模的模型。我们没有选择 GBDT+LR 的原因除了深度模型的离线效果更好外，还有一个很大的原因就是需要额外的工程开发成本。这里不得不夸一下谷歌的 TensorFlow 深度学习框架，除了提供很多高阶API帮助开发者快速实现模型外，还提供了 TensorFlow Serving 这种模型服务的工具，大大减低了从离线模型到线上预测的实现成本。

这里以二手房详情页底部的猜你喜欢场景为例（产品 UI 如图 12），讲一下我们在排序模型上的优化过程。

图 12 贝壳 App 二手房详情页 - 猜你喜欢模块 UI

这个场景的 baseline 模型是 LR 模型，而在优化时我们选择了 DeepFM 模型，模型结构如图 13。选择这个模型的原因主要有两个，一是作为高低阶关系相互作用的深度模型，能够有效地挖掘复杂关系；二是端到端的 Embedding 学习以及 FM 的自动特征交叉，使得该模型在较低的特征工程成本下也能有不俗的效果。模型的实现方面，网上已经有很多基于 TensorFlow 框架实现 DeepFM 模型的优秀代码，再结合论文本身以及一些网上的参考资料，模型的实现变得事半功倍。

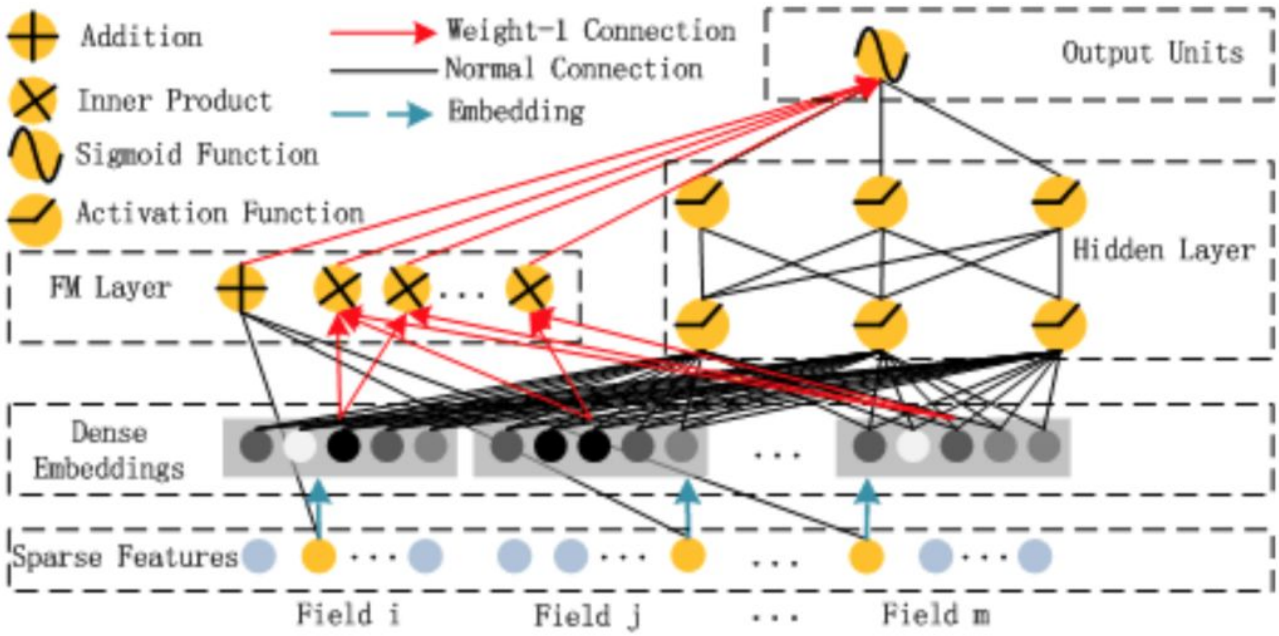


图 13 DeepFM 模型结构图

在样本构建方面，我们对点击、曝光样本做了更细粒度的划分，把同一次推荐的结果作为列表，并划分为包含点击和不包含点击的两类。其中，包含点击的列表中有以下 4 种：

1. skip expo，即用户最后一个点击之上曝光但未点击的样本。
2. click，用户点击的样本。
3. mid expo，在用户最后一个点击之后，模块最后一个展示位置之前，曝光未点击的样本。
4. last expo，最后一个展示位置，且曝光未点击的样本。

而未包含点击的列表中有以下 3 种：

1. first expo，是第一个展示位置，且曝光未点击。
2. last expo，最后一个展示位置，且曝光未点击的样本。
3. mid expo，除去第一个和最后一个展示位置，且曝光未点击的样本。

在保持测试集不变的前提下，通过尝试改变训练数据集中的样本组合，选取离线评估效果最好的组合方式，如图 14 所示。

图 14 训练样本构建示意图

在线上的 AB 实验中，以 LR 排序模型作为对照组进行对比，采用 DeepFM 排序模型的实验组相对提升了 12.65%。

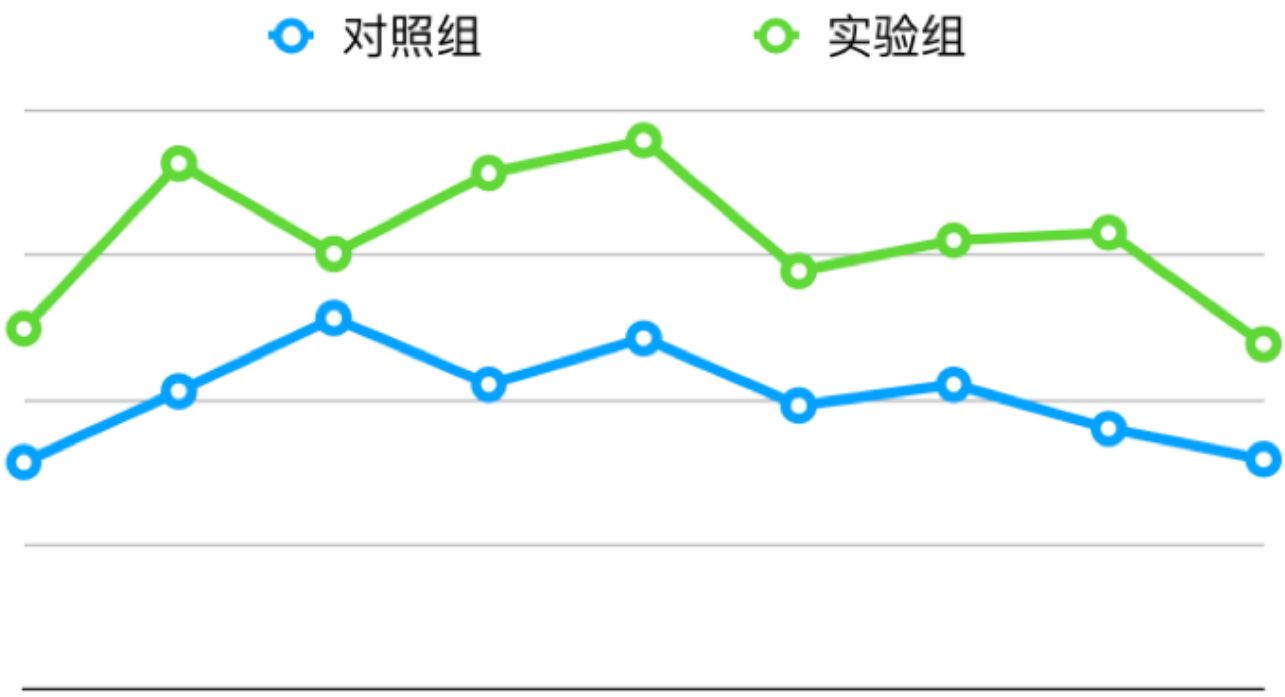


图 15 DeepFM 模型与 LR 模型的线上 AB 实验

贝壳 App 上的 UI 大多采用左右或上下滑动的形式展示推荐结果，因此不可避免的会带来样本的位置偏差问题。我们尝试将实际曝光位置作为特征加入到模型中，以此来吸收位置偏差对预测结果的影响。在详情页底部场景，一共有 6 个推荐卡片的展示位置（从 0 开始）。如图 16 所示，我们使

用相同的离线模型，在同一个测试集上，只改变曝光位置这维特征的填充值并对比得到的 AUC。其中，对照组是未加入曝光位置特征的模型。

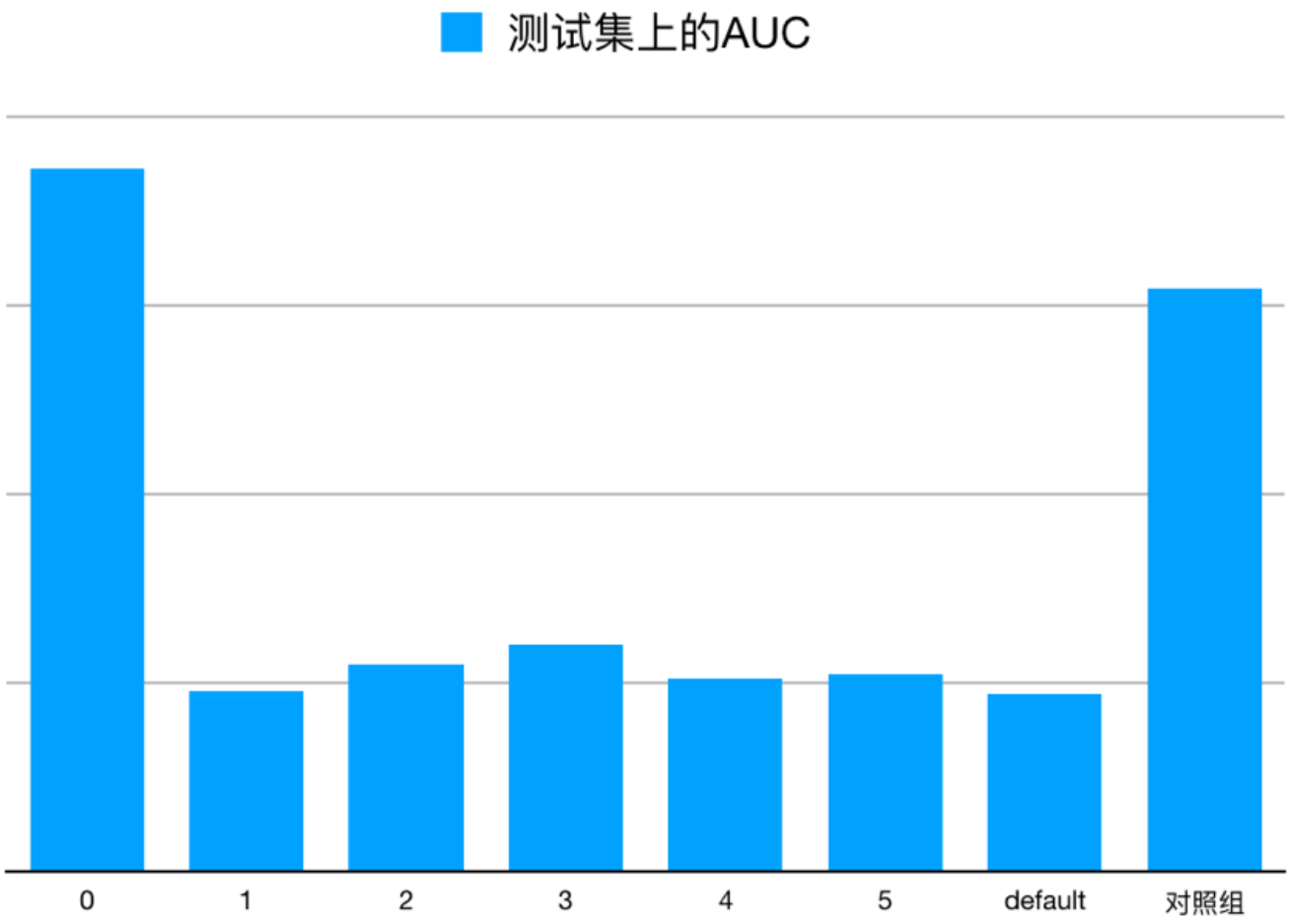


图 16 预测时填充不同位置在测试集上的 AUC 对比

我们选择了离线评估效果最好的一组（曝光位置填充 0）进行线上实验。在 AB 实验中，与未加入曝光位置特征的 DeepFM 模型相比，相对提升了 4.76%。

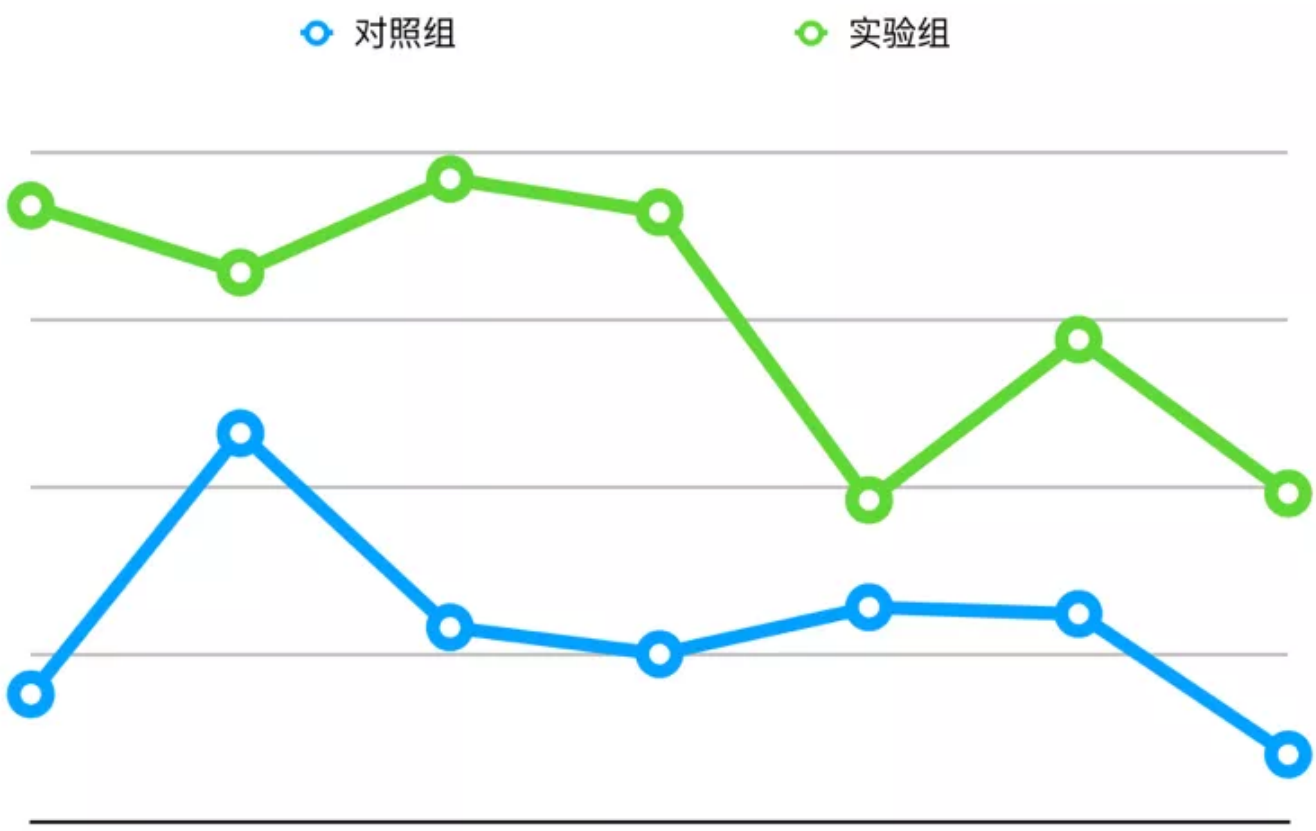


图 17 优化位置偏差前后版本的线上 AB 实验

除此之外，我们还尝试了 PAL[4] 论文中解决位置偏差问题的思路，在模型中增加结构来学习用户看到物料的概率，目前还在调试阶段。另外，随着公司规模的发展，对推荐系统的优化目标已经从 pCTR 逐渐过渡到 pCVR，组内也有同学在使用阿里的 ESMM[5] 模型，进行多目标优化的尝试。

房产证文字识别

房产证文字识别主要应用在房产交易环节。经纪人拍照上传成交房屋的产权证，后台需要人工录入房产证有效字段信息，如产权人、房屋坐落、房屋面积等，由此引入 OCR 自动识别系统来降低人工录入的人效成本，提升录入准确率。

OCR 房产证文字识别系统主要由五大模块组成，其流程如图 18 所示。其中预处理模块负责对无关图像进行过滤和校正图像方向；模板处理模块负责解决房本种类繁多的问题，采用模板匹配的方式进行结构化信息提取，提升系统的扩展性；OCR 检测和识别模块分别负责房本图像的文本行定位和识别；最后由后处理模块负责对识别结果进行合并和信息提炼。

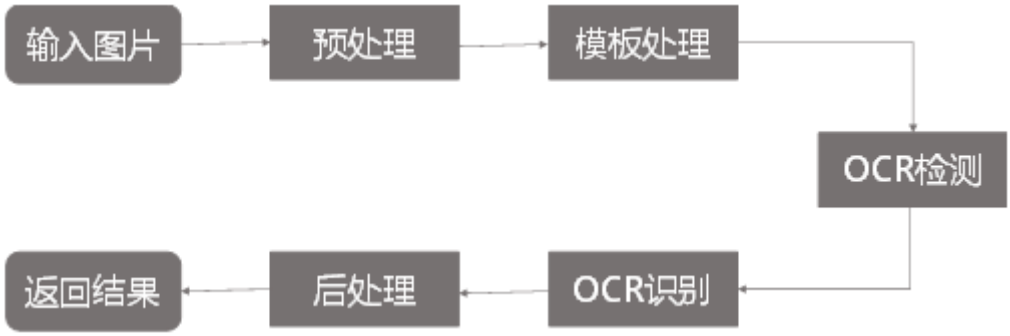
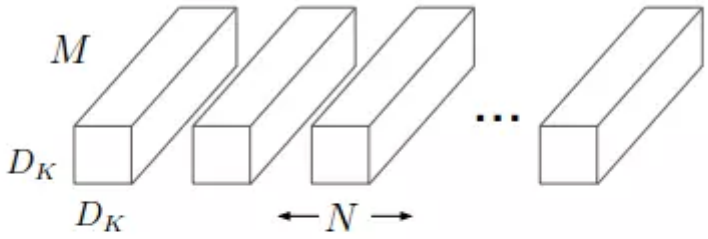


图 18 房产证文字识别流程图

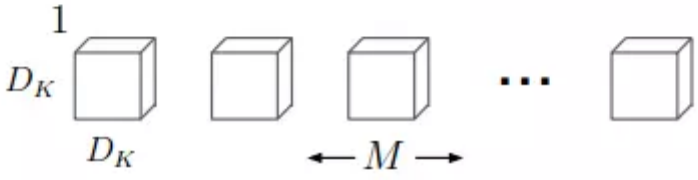
整个系统涉及多个深度学习模型，主要包括 3 个图像分类模型（无关图片过滤、图像方向判别和房产证模板分类）、1 个文字检测模型和 1 个文字识别模型，这些模型均采用 TensorFlow 框架搭建。下面重点介绍这三个模型的实现原理及效果。

分类模型

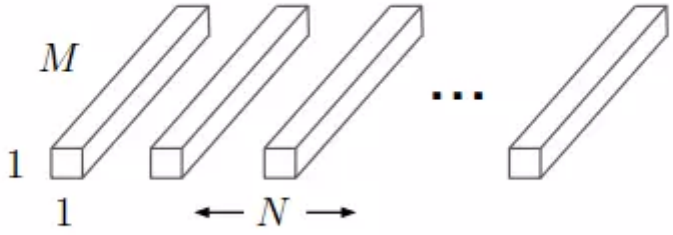
图像分类模型采用了已在 ImageNet 数据集上预训练的 MobileNet 模型进行迁移训练。MobileNet 是一种专门为移动和嵌入设备提出的高效模型，其最大的特点就是轻量级，参数量少，运算速率快。MobileNet 是基于深度可分离卷积实现的，通俗来说，深度可分离卷积就是把标准的卷积运算分解为深度卷积 (depthwise convolution) 和逐点卷积 (pointwise convolution)，从而达到减少参数量和降低计算量的目的。其卷积分解过程示意图如图 19。



(a) Standard Convolution Filters



(b) Depthwise Convolutional Filters



(c) 1×1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

图 19 深度可分离卷积示意图

图 19(a) 是标准的卷积运算，假设输入图片大小为（6，6，3），标准卷积运算用的是（4，4，3，5）的卷积核（4x4是卷积核大小，3 是卷积核通道数，5 是卷积核数量），stride=1，无 padding。那么输出的特征映射为（3，3，5），该过程参数量为 4x4x3x5=240。

图 19(b) 和图 19(c) 是深度卷积和逐点卷积运算过程。将标准卷积采用的（4，4，3，5）卷积核分解为（4，4，1，3）和（1，1，3，5）两个卷积核，首先将输入的图片经过深度卷积运算（卷积核 4x4x1x3）得到特征映射为（3，3，3），然后再经过逐点卷积（卷积核为 1x1x3x5）得到最终映射图为（3，3，5），该过程参数量为 4x4x1x3+1x1x3x5=63，因此深度可分离卷积明显减少了网络参数量，达到了降低计算量的作用。在贝壳房产证文字识别的预处理和模板分类阶段均采用了 MobileNet 网络分类模型，不仅训练推理过程快速，而且其分类精确度也均达到了 99% 以上。

OCR 检测模型

文字检测是一种特殊的目标检测场景，因此基于深度学习的 OCR 文字检测模型基本都是从通用目标检测算法演变而来。其中最具有代表性的模型包括有基于 faster-r-cnn 演变而来的 CTPN、基于 SSD 演变而来的 TextBoxes、TextBoxes++ 以及基于图像分割算法从全卷积网络 FCN 演变而来的 EAST、Advanced EAST。结合具体场景和综合实验对比，我们最终采用了 Advanced EAST 模型，下面对该模型做简单介绍。

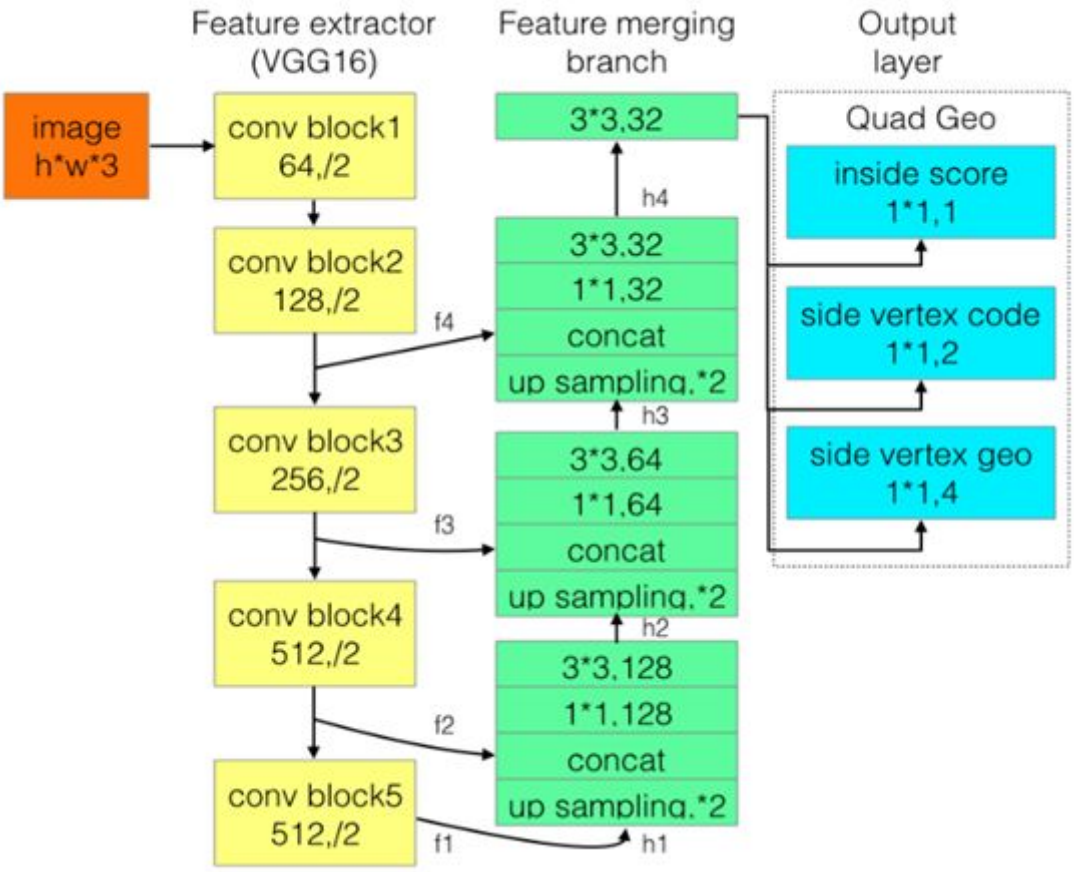


图 20 Advanced EAST 网络结构图

如图 20 所示，Advanced EAST 模型包括两阶段，第一阶段基于全卷积网络模型得到文本区域激活矩阵，主要由三部分组成——特征提取模块（图中黄色块）、特征融合模块（图中绿色块）和输出层（图中蓝色块）；第二阶段是采用基于非极大值抑制的后置处理得到最终文本框坐标。

特征提取模块采用 VGG16 网络模型提取图像特征，该网络利用在 ImageNet 数据集上预训练的卷积网络参数做初始化，主要提取四个级别的特征图（记为 fi）,其大小分别为输入图像的 1/32，1/16，1/8， 1/4。特征融合首先将来自上一级别的特征图通过上池化 (unpool) 扩大其尺寸大小，然后与当前层特征图进行通道合并，最后再通过 conv1x1 和 conv3x3 来减少通道数和融合局部信息以得到该层最终特征图，具体实现可参照下图 21 公式所示。另外，需要注意的是在所有特征图合并完成之后，需要再次通过 conv3x3 运算。

$$g_i = \begin{cases} unpool(h_i) & \text{if i}\leq 3 \\ conv_{3\times 3}(h_i) & \text{if i}=4 \end{cases}$$

$$h_i = \begin{cases} f_i & \text{if i}=1 \\ conv_{3\times 3}(conv_{1\times 1}([g_{i-1}; f_i])) & \text{otherwise} \end{cases}$$

图 21 最终特征图计算公式

输出层分别采用三个 conv1x1 卷积得到 inside score（1位，表示该区域是否在文本框内）、side vertex code（2位，表示该区域是否属于文本框边界以及属于文本框的头部还是尾部）和 vertex geo（4 位，表示头部（或尾部）的两个顶点坐标）。

第一阶段得到文本像素激活矩阵之后，开始进行后置处理，即通过查找左右邻接和上下邻接区域组成文本边框的头部和尾部集合，然后分别用头部和尾部的集合加权平均得到最终文本框的左顶点和右顶点坐标。在房产证文字识别的实践中发现，此模型存在长文本检测不全的问题，我们最终通过采用多次查找邻接区域，得到最远端的头部集合和尾部集合来预测坐标。最终在房产证图像上得到文字检测准召率分别为 p=95.89%，r=90.60%。

OCR 识别模型

OCR 文字识别采用了经典的文字检测模型 —— CRNN+CTC (Connectionist Temporal Classification，连接主义时序分类器)，该模型网络结构如图 22 所示。输入图像首先经过卷积神经网络提取特征图，然后经过双向 LSTM 网络提取时序特征，最后经过 CTC 网络计算文本序列概率，最大序列概率对应的文本序列即是预测文本结果。CTC 主要适应于解决输入特征和输出标签之间的对齐关系不确定的序列问题，常用于语音识别，而在 OCR 识别中同样存在图像像素点与文本序列对齐关系不确定的问题。

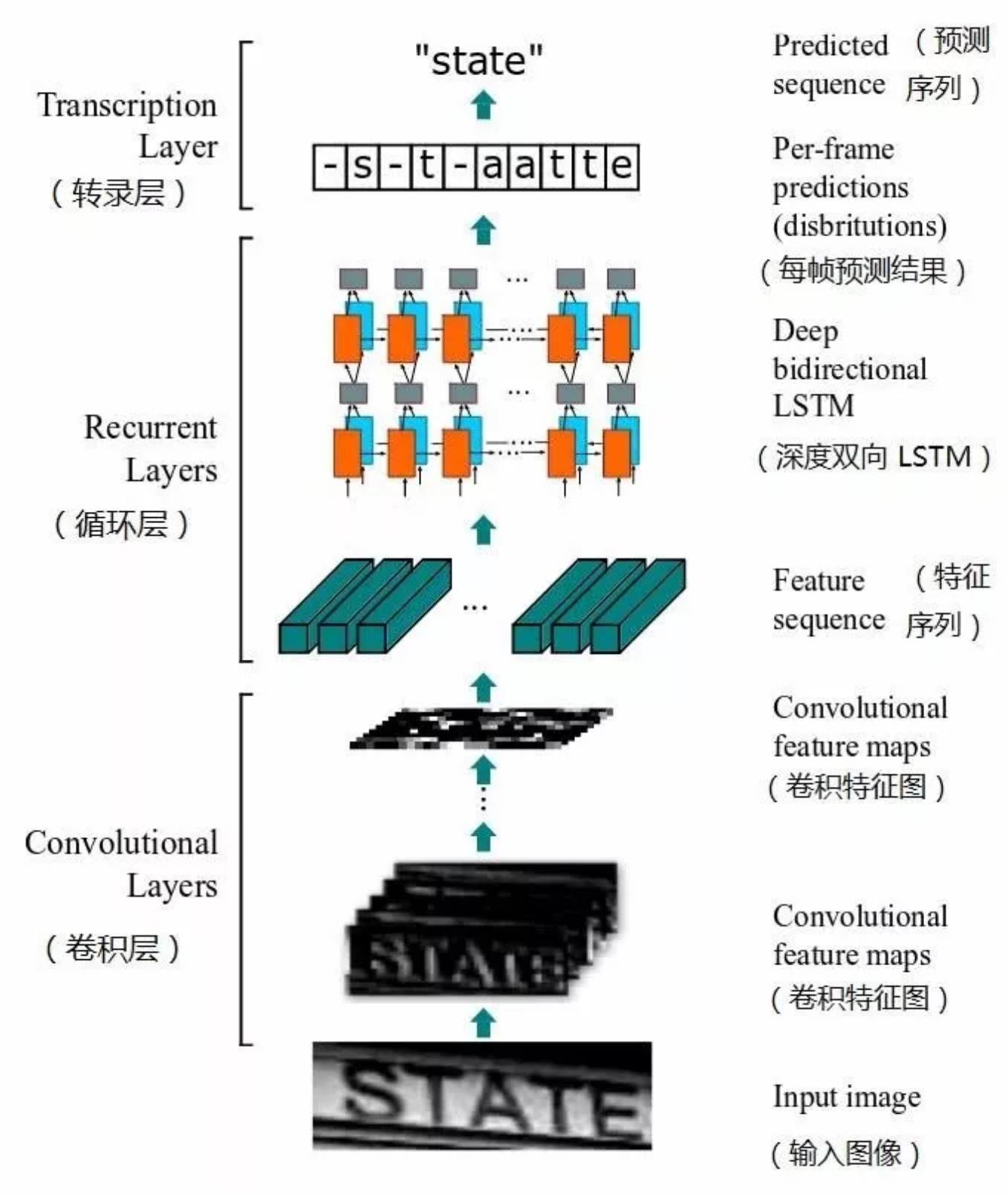


图 22 CRNN+CTC 网络结构图

贝壳房产证文字识别系统支持识别 7000 多个字符，基本包含了所有的常见字符和大量的生僻字符。在有些字段的提取中，我们还训练了专有模型来识别特定字段，最终房产证文字的整体识别率达到了 96% 以上。

结语

随着互联网、AI 技术的高速发展，好的模型如雨后春笋一般不断地涌现。但正如最优化理论中"没有免费午餐"定理(No Free Lunch，简称 NFL)所说的那样，不存在一个大而全的模型可以解决所有行业中的问题。

想要让一个模型在行业数据上得到最好的效果，需要算法工程师不断地精心探索和打磨。TensorFlow 可以将算法工程师从实现复杂前沿模型的繁杂工作中解放出来，更聚焦关注模型的优化和落地应用本身，从而大幅提升算法团队的工作效率。

关于作者

潘昊，现就职于贝壳找房数据智能中心策略算法部，担任高级算法工程师，从事房源策略算法相关工作

陈嘉远，现就职于贝壳找房数据智能中心策略算法部，担任资深算法工程师，专注图像视频算法相关工作

袁彬，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

参考文献

[1] Mihajlo Grbovic , Haibin Cheng. Real-time Personalization using Embeddings for Search Ranking at Airbnb. 2018

[2] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen. Wide & Deep Learning for Recommender Systems. 2016.

[3] Huifeng Guo , Ruiming Tang, Yunming Ye, et al. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. 2017.

[4] Huifeng Guo, Jinkai Yu, Qing Liu, Ruiming Tang, et al. PAL: A Position-bias Aware Learning Framework for CTR Prediction in Live Recommender Systems. 2019.

[5] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, et al. Entire Space Multi-Task Model: An Effective Approach for Estimating Post-Click Conversion Rate. 2018.

[6] Howard A G , Zhu M , Chen B , et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications[J]. 2017.

[7] Zhou X , Yao C , Wen H , et al. EAST: An Efficient and Accurate Scene Text Detector[J]. 2017.

[8] Graves A . Connectionist Temporal Classification[M]// Supervised Sequence Labelling with Recurrent Neural Networks. Springer Berlin Heidelberg, 2012.