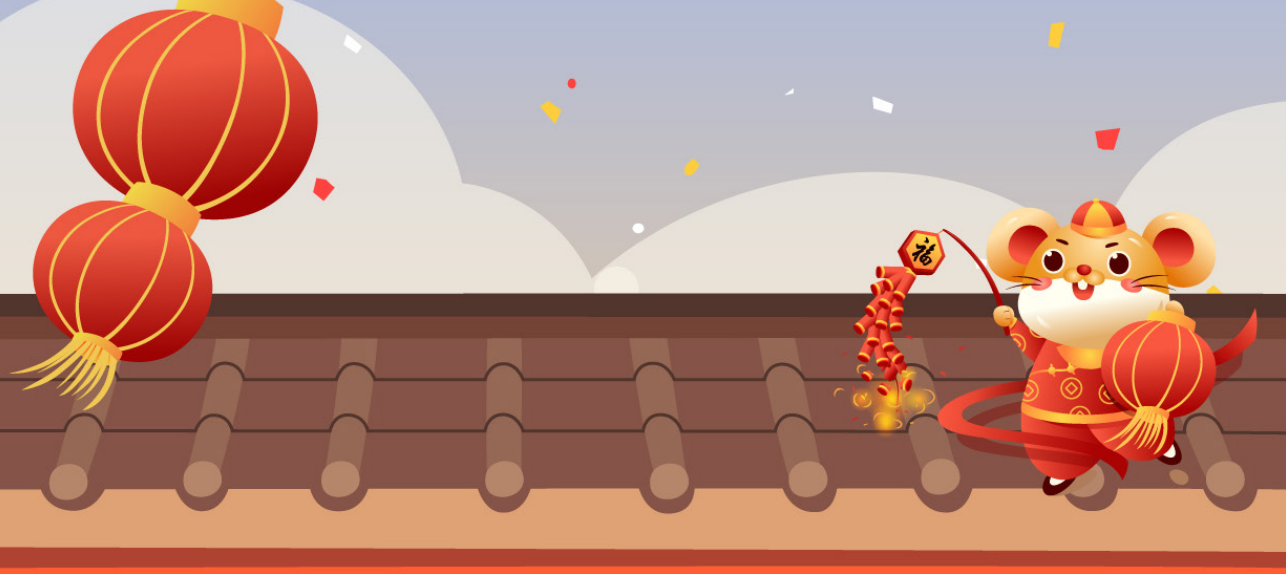


美团点评
2019 技术年货

CODE A BETTER LIFE

【算法篇】

美团点评

微信扫码关注技术团队公众号

tech.meituan.com
美团技术博客

新年
快乐

目录

算法

用能力支撑了每天数十亿次的交易量；

用实力攀登算法领域最高学术殿堂；

用技术绘制美团点评最美的 AI 全景图。

美团算法团队正在构建的 AI 相关技术囊括了视觉、语音、自然语言处理、机器学习、知识图谱等。以美团 / 大众点评 App 搜索、推荐为核心，面向外卖配送的策略、调度算法、定价系统，延伸到无人配送的自动驾驶、智能耳机里的语音识别、人脸识别，再到连接用户端的客服系统，连接商家端的金融体系、供应链系统也汇聚了美团点评正在构建的庞大知识图谱……

探索无止境，实践出真知。美团点评算法团队孜孜不倦的学习与探索，并在实际业务场景中引入新技术，总结新领悟。

美团 BERT 的探索和实践

杨扬　佳昊　金刚

2018 年，自然语言处理（Natural Language Processing，NLP）领域最激动人心的进展莫过于**预训练语言模型**，包括基于 RNN 的 ELMo[1] 和 ULMFiT[2]，基于 Transformer[3] 的 OpenAI GPT[4] 及 Google BERT[5] 等。下图 1 回顾了近年来预训练语言模型的发展史以及最新的进展。预训练语言模型的成功，证明了我们可以从海量的无标注文本中学到潜在的语义信息，而无需为每一项下游 NLP 任务单独标注大量训练数据。此外，预训练语言模型的成功也开创了 NLP 研究的新范式 [6]，即首先使用大量无监督语料进行语言模型预训练（Pre-training），再使用少量标注语料进行微调（Fine-tuning）来完成具体 NLP 任务（分类、序列标注、句间关系判断和机器阅读理解等）。

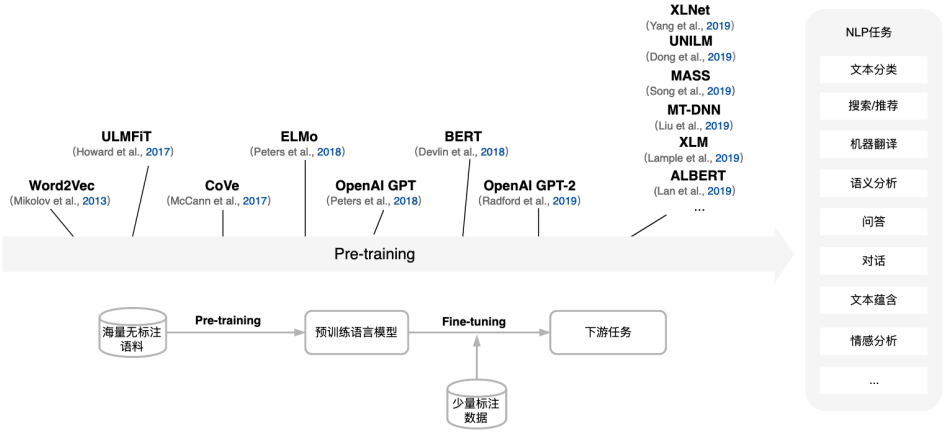


图 1　NLP Pre-training and Fine-tuning 新范式及相关扩展工作

所谓的“预训练”，其实并不是什么新概念，这种“Pre-training and Fine-tuning”的方法在图像领域早有应用。2009 年，邓嘉、李飞飞等人在 CVPR 2009 发布了 ImageNet 数据集 [7]，其中 120 万张图像分为 1000 个类别。基于 ImageNet，以图像分类为目标使用深度卷积神经网络（如常见的 ResNet、VCG、Inception 等）

进行预训练，得到的模型称为预训练模型。针对目标检测或者语义分割等任务，基于这些预训练模型，通过一组新的全连接层与预训练模型进行拼接，利用少量标注数据进行微调，将预训练模型学习到的图像分类能力迁移到新的目标任务。预训练的方式在图像领域取得了广泛的成功，比如有学者将 ImageNet 上学习得到的特征表示用于 PSACAL VOC 上的物体检测，将检测率提高了 20%[8]。

他山之石，可以攻玉。图像领域预训练的成功也启发了 NLP 领域研究，深度学习时代广泛使用的词向量（即词嵌入，Word Embedding）即属于 NLP 预训练工作。使用深度神经网络进行 NLP 模型训练时，首先需要将待处理文本转为词向量作为神经网络输入，词向量的效果会影响到最后模型效果。词向量的效果主要取决于训练语料的大小，很多 NLP 任务中有限的标注语料不足以训练出足够好的词向量，通常使用跟当前任务无关的大规模未标注语料进行词向量预训练，因此预训练的另一个好处是能增强模型的泛化能力。目前，大部分 NLP 深度学习任务中都会使用预训练好的词向量（如 Word2Vec[9] 和 GloVe[10] 等）进行网络初始化（而非随机初始化），从而加快网络的收敛速度。

预训练词向量通常只编码词汇间的关系，对上下文信息考虑不足，且无法处理一词多义问题。如“bank”一词，根据上下文语境不同，可能表示“银行”，也可能表示“岸边”，却对应相同的词向量，这样显然是不合理的。为了更好的考虑单词的上下文信息，Context2Vec[11] 使用两个双向长短时记忆网络（Long Short Term Memory，LSTM）[12] 来分别编码每个单词左到右（Left-to-Right）和右到左（Right-to-Left）的上下文信息。类似地，ELMo 也是基于大量文本训练深层双向 LSTM 网络结构的语言模型。ELMo 在词向量的学习中考虑深层网络不同层的信息，并加入到单词的最终 Embedding 表示中，在多个 NLP 任务中取得了提升。ELMo 这种使用预训练语言模型的词向量作为特征输入到下游目标任务中，被称为 Feature-based 方法。

另一种方法是微调（Fine-tuning）。GPT、BERT 和后续的预训练工作都属于这一范畴，直接在深层 Transformer 网络上进行语言模型训练，收敛后针对下游目标任务进行微调，不需要再为目标任务设计 Task-specific 网络从头训练。关于

NLP 领域的预训练发展史，张俊林博士写过一篇很详实的介绍 [13]，本文不再赘述。

Google AI 团队提出的预训练语言模型 BERT（Bidirectional Encoder Representations from Transformers），在 11 项自然语言理解任务上刷新了最好指标，可以说是近年来 NLP 领域取得的最重大的进展之一。BERT 论文也斩获 NLP 领域顶会 NAACL 2019 的最佳论文奖，BERT 的成功也启发了大量的后续工作，不断刷新了 NLP 领域各个任务的最佳水平。有 NLP 学者宣称，属于 NLP 的 ImageNet 时代已经来临 [14]。

美团点评作为中国领先的生活服务电子商务平台，涵盖搜索、推荐、广告、配送等多种业务场景，几乎涉及到各种类型的自然语言处理任务。以大众点评为例，迄今为止积累了近 40 亿条文本 UGC，如何高效而准确地完成对海量 UGC 的自然语言理解和处理是美团点评技术团队面临的挑战之一。美团点评 NLP 团队一直紧跟业界前沿技术，开展了基于美团点评业务数据的预训练研究工作，训练了更适配美团点评业务场景的 MT-BERT 模型，通过微调将 MT-BERT 落地到多个业务场景中，并取得了不错的业务效果。

BERT 是基于 Transformer 的深度双向语言表征模型，基本结构如图 2 所示，本质上是利用 Transformer 结构构造了一个多层双向的 Encoder 网络。Transformer 是 Google 在 2017 年提出的基于自注意力机制（Self-attention）的深层模型，在包括机器翻译在内的多项 NLP 任务上效果显著，超过 RNN 且训练速度更快。不到一年时间内，Transformer 已经取代 RNN 成为神经网络机器翻译的 State-Of-The-Art（SOTA）模型，包括谷歌、微软、百度、阿里、腾讯等公司的线上机器翻译模型都已替换为 Transformer 模型。关于 Transformer 的详细介绍可以参考 Google 论文《Attention is all you need》[3]。

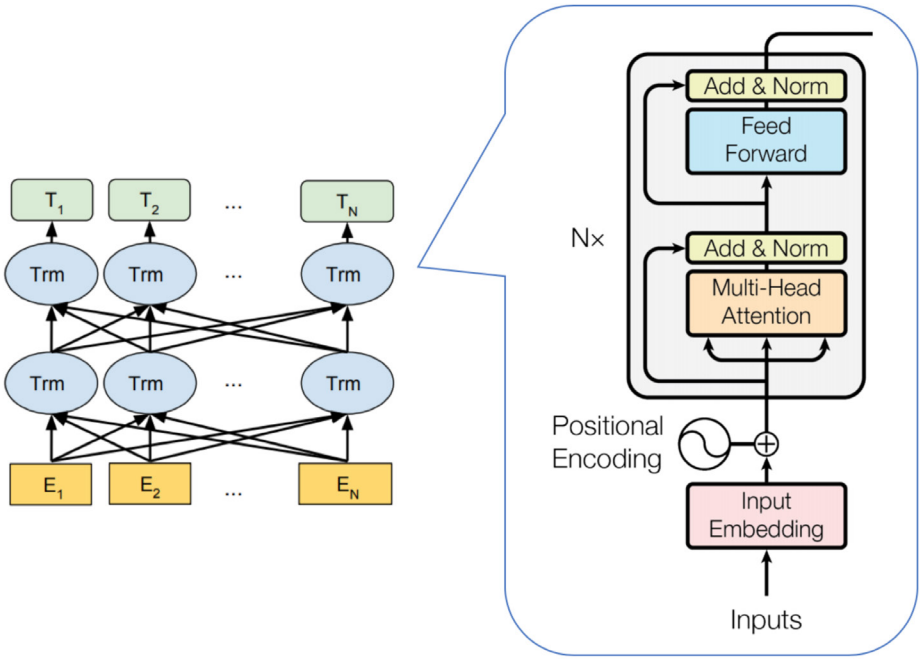


图 2　BERT 及 Transformer 网络结构示意图

模型结构

如表 1 所示，根据参数设置的不同，Google 论文中提出了 Base 和 Large 两种 BERT 模型。

表1　BERT Base和Large模型参数对比

模型	Layers	Hidden Size	Attention Head	参数数量
Base	12	768	12	110M
Large	24	1024	16	340M

输入表示

针对不同的任务，BERT 模型的输入可以是单句或者句对。对于每一个输入的 Token，它的表征由其对应的词表征（Token Embedding）、段表征（Segment Embedding）和位置表征（Position Embedding）相加产生，如图 3 所示：

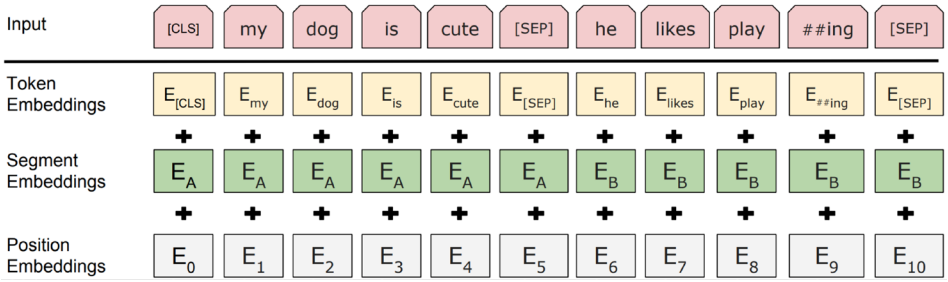


图 3　BERT 模型的输入表示

- 对于英文模型，使用了 Wordpiece 模型来产生 Subword 从而减小词表规模；对于中文模型，直接训练基于字的模型。
- 模型输入需要附加一个起始 Token，记为 [CLS]，对应最终的 Hidden State（即 Transformer 的输出）可以用来表征整个句子，用于下游的分类任务。
- 模型能够处理句间关系。为区别两个句子，用一个特殊标记符 [SEP] 进行分隔，另外针对不同的句子，将学习到的 Segment Embeddings 加到每个 Token 的 Embedding 上。
- 对于单句输入，只有一种 Segment Embedding；对于句对输入，会有两种 Segment Embedding。

预训练目标

BERT 预训练过程包含两个不同的预训练任务，分别是 Masked Language Model 和 Next Sentence Prediction 任务。

Masked Language Model（MLM）

通过随机掩盖一些词（替换为统一标记符 [MASK]），然后预测这些被遮盖的词来训练双向语言模型，并且使每个词的表征参考上下文信息。

这样做会产生两个缺点：（1）会造成预训练和微调时的不一致，因为在微调时 [MASK] 总是不可见的；（2）由于每个 Batch 中只有 15% 的词会被预测，因此模型的收敛速度比起单向的语言模型会慢，训练花费的时间会更长。对于第一个缺点的解

决办法是，把 80% 需要被替换成 [MASK] 的词进行替换，10% 的随机替换为其他词，10% 保留原词。由于 Transformer Encoder 并不知道哪个词需要被预测，哪个词是被随机替换的，这样就强迫每个词的表达需要参照上下文信息。对于第二个缺点目前没有有效的解决办法，但是从提升收益的角度来看，付出的代价是值得的。

Next Sentence Prediction（NSP）

为了训练一个理解句子间关系的模型，引入一个下一句预测任务。这一任务的训练语料可以从语料库中抽取句子对包括两个句子 A 和 B 来进行生成，其中 50% 的概率 B 是 A 的下一个句子，50% 的概率 B 是语料中的一个随机句子。NSP 任务预测 B 是否是 A 的下一句。NSP 的目的是获取句子间的信息，这点是语言模型无法直接捕捉的。

Google 的论文结果表明，这个简单的任务对问答和自然语言推理任务十分有益，但是后续一些新的研究 [15] 发现，去掉 NSP 任务之后模型效果没有下降甚至还有提升。我们在预训练过程中也发现 NSP 任务的准确率经过 1–2 个 Epoch 训练后就能达到 98%–99%，去掉 NSP 任务之后对模型效果并不会有太大的影响。

数据 & 算力

Google 发布的英文 BERT 模型使用了 BooksCorpus（800M 词汇量）和英文 Wikipedia（2500M 词汇量）进行预训练，所需的计算量非常庞大。BERT 论文中指出，Google AI 团队使用了算力强大的 Cloud TPU 进行 BERT 的训练，BERT Base 和 Large 模型分别使用 4 台 Cloud TPU（16 张 TPU）和 16 台 Cloud TPU（64 张 TPU）训练了 4 天（100 万步迭代，40 个 Epoch）。但是，当前国内互联网公司主要使用 Nvidia 的 GPU 进行深度学习模型训练，因此 BERT 的预训练对于 GPU 资源提出了很高的要求。

美团 BERT（MT–BERT）的探索分为四个阶段：（1）开启混合精度实现训练加速；（2）在通用中文语料基础上加入大量美团点评业务语料进行模型预训练，完成领域迁移；（3）预训练过程中尝试融入知识图谱中的实体信息；（4）通过在业务数据上进行微调，支持不同类型的业务需求。MT–BERT 整体技术框架如图 4 所示：

图 4　MT-BERT 整体技术框架

基于美团点评 AFO 平台的分布式训练

正如前文所述，BERT 预训练对于算力有着极大要求，我们使用的是美团内部开发的 AFO[16]（AI Framework On Yarn）框架进行 MT-BERT 预训练。AFO 框架基于 YARN 实现数千张 GPU 卡的灵活调度，同时提供基于 Horovod 的分布式训练方案，以及作业弹性伸缩与容错等能力。Horovod 是 Uber 开源的深度学习工具 [17]，它的发展吸取了 Facebook《一小时训练 ImageNet》论文 [18] 与百度 Ring Allreduce[19] 的优点，可为用户实现分布式训练提供帮助。根据 Uber 官方分别使用标准分布式 TensorFlow 和 Horovod 两种方案，分布式训练 Inception V3 和 ResNet-101 TensorFlow 模型的实验验证显示，随着 GPU 的数量增大，Horovod 性能损失远小于 TensorFlow，且训练速度可达到标准分布式 TensorFlow 的近两倍。相比于 Tensorflow 分布式框架，Horovod 在数百张卡的规模上依然可以保证稳定的加速比，具备非常好的扩展性。

Horovod 框架的并行计算主要用到了两种分布式计算技术：控制层的 Open

MPI 和数据层的 Nvidia NCCL。控制层面的主要作用是同步各个 Rank（节点），因为每个节点的运算速度不一样，运算完每一个 Step 的时间也不一样。如果没有一个同步机制，就不可能对所有的节点进行梯度平均。Horovod 在控制层面上设计了一个主从模式，Rank 0 为 Master 节点，Rank1-n 为 Worker 节点，每个 Worker 节点上都有一个消息队列，而在 Master 节点上除了一个消息队列，还有一个消息 Map。每当计算框架发来通信请求时，比如要执行 Allreduce，Horovod 并不直接执行 MPI，而是封装了这个消息并推入自己的消息队列，交给后台线程去处理。后台线程采用定时轮询的方式访问自己的消息队列，如果非空，Woker 会将自己收到的所有 Tensor 通信请求都发给 Master。因为是同步 MPI，所以每个节点会阻塞等待 MPI 完成。Master 收到 Worker 的消息后，会记录到自己的消息 Map 中。如果一个 Tensor 的通信请求出现了 n 次，也就意味着，所有的节点都已经发出了对该 Tensor 的通信请求，那么这个 Tensor 就需要且能够进行通信。Master 节点会挑选出所有符合要求的 Tensor 进行 MPI 通信。不符合要求的 Tensor 继续留在消息 Map 中，等待条件满足。决定了 Tensor 以后，Master 又会将可以进行通信的 Tensor 名字和顺序发还给各个节点，通知各个节点可以进行 Allreduce 运算。

混合精度加速

当前深度学习模型训练过程基本采用单精度（Float 32）和双精度（Double）数据类型，受限于显存大小，当网络规模很大时 Batch Size 就会很小。Batch Size 过小一方面容易导致网络学习过程不稳定而影响模型最终效果，另一方面也降低了数据吞吐效率，影响训练速度。为了加速训练及减少显存开销，Baidu Research 和 Nvidia 在 ICLR 2018 论文中 [20] 合作提出了一种 Float32（FP32）和 Float16（FP16）混合精度训练的方法，并且在图像分类和检测、语音识别和语言模型任务上进行了有效验证。Nvidia 的 Pascal 和 Volta 系列显卡除了支持标准的单精度计算外，也支持了低精度的计算，比如最新的 Tesla V100 硬件支持了 FP16 的计算加速，P4 和 P40 支持 INT8 的计算加速，而且低精度计算的峰值要远高于单精浮点的计算峰值。

为了进一步加快 MT-BERT 预训练和推理速度，我们实验了混合精度训练方式。混合精度训练指的是 FP32 和 FP16 混合的训练方式，使用混合精度训练可以加速训练过程并且减少显存开销，同时兼顾 FP32 的稳定性和 FP16 的速度。在模型计算过程中使用 FP16 加速计算过程，模型训练过程中权重会存储成 FP32 格式（FP32 Master-weights），参数更新时采用 FP32 类型。利用 FP32 Master-weights 在 FP32 数据类型下进行参数更新可有效避免溢出。 此外，一些网络的梯度大部分在 FP16 的表示范围之外，需要对梯度进行放大使其可以在 FP16 的表示范围内，因此进一步采用 Loss Scaling 策略通过对 Loss 进行放缩，使得在反向传播过程中梯度在 FP16 的表示范围内。

为了提高预训练效率，我们在 MT-BERT 预训练中采用了混合精度训练方式。

加速效果

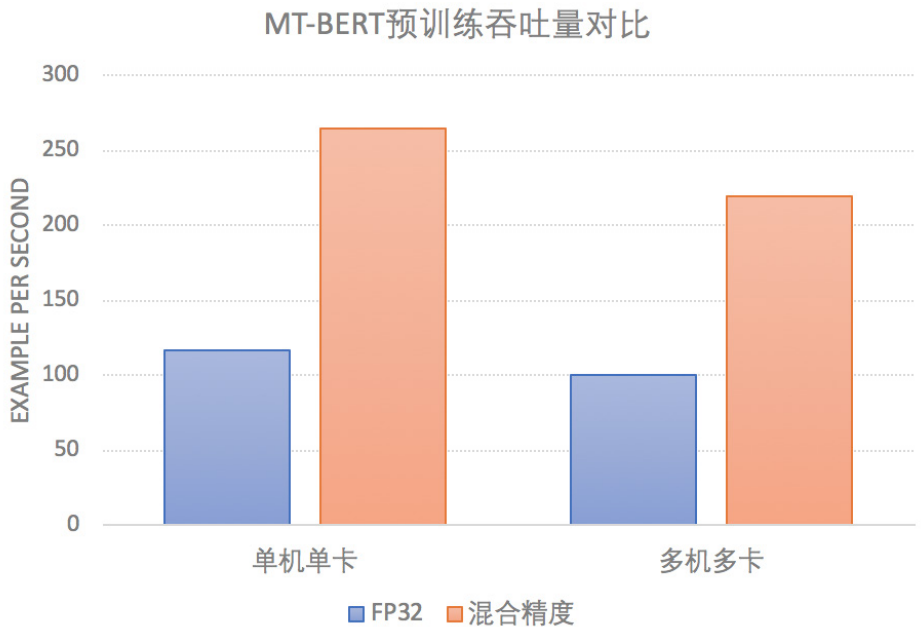


图 5　MT-BERT 开启混合精度在 Nvidia V100 上训练吞吐量的对比
（Tensorflow 1.12；Cuda 10.0；Horovod 0.15.2）

如图 5 所示，开启混合精度的训练方式在单机单卡和多机多卡环境下显著提升了训练速度。为了验证混合精度模型会不会影响最终效果，我们分别在美团点评业务和通用 Benchmark 数据集上进行了微调实验，结果见表 2 和表 3。

表2　开启混合精度训练的MT-BERT模型在美团点评业务Benchmark上效果对比

数据集	Metric	MT-BERT FP32	MT-BERT 混合精度	Google BERT
细粒度情感分析	Macro-F1	72.04%	72.25%	71.63%
Query 意图分类	F1	93.27%	93.13%	92.68%
Query 成分分析（NER）	F1	91.46%	91.05%	90.66%

表3　开启混合精度训练的MT-BERT模型在中文通用Benchmark上效果对比

数据集	Metric	MT-BERT FP32	MT-BERT 混合精度	Google BERT
MSRA-NER	F1	95.89%	95.75%	95.76%
LCQMC	Accuracy	86.74%	85.87%	86.06%
ChnSentiCorp	Accuracy	95.00%	94.92%	92.25%
NLPCC-DBQA	MRR	94.07%	93.24%	93.55%
XNLI	Accuracy	78.10%	76.57%	77.47%

通过表 2 和表 3 结果可以发现，开启混合精度训练的 MT-BERT 模型并没有影响效果，反而训练速度提升了 2 倍多。

领域自适应

Google 发布的中文 BERT 模型是基于中文维基百科数据训练得到，属于通用领域预训练语言模型。由于美团点评积累了大量业务语料，比如用户撰写的 UGC 评论和商家商品的文本描述数据，为了充分发挥领域数据的优势，我们考虑在 Google 中文 BERT 模型上加入领域数据继续训练进行领域自适应（Domain Adaptation），使得模型更加匹配我们的业务场景。实践证明，这种 Domain-aware Continual Training 方式，有效地改进了 BERT 模型在下游任务中的表现。由于 Google 未发布中文 BERT Large 模型，我们也从头预训练了中文 MT-BERT Large 模型。

我们选择了 5 个中文 Benchmark 任务以及 3 个美团点评业务 Benchmark 在内的 8 个数据集对模型效果进行验证。实验结果如表 4 所示，MT-BERT 在通用 Benchmark 和美团点评业务 Benchmark 上都取得了更好的效果。

表4 MT-BERT模型和Google BERT模型在8个Benchmark上的效果对比

Benchmark	Metric	Google BERT	MT-BERT
MSRA-NER	F1	95.76%	95.89%
LCQMC	Accuracy	86.06%	86.74%
ChnSentiCorp	Accuracy	92.25%	95.00%
NLPCC-DBQA	MRR	93.55%	94.07%
XNLI	Accuracy	77.47%	78.10%
细粒度情感分析	Macro-F1	71.63%	72.04%
Query 意图分类	F1	92.68%	93.27%
Query 成分分析（NER）	F1	90.66%	91.46%

知识融入

BERT 在自然语言理解任务上取得了巨大的成功，但也存在着一些不足。其一是常识（Common Sense）的缺失。人类日常活动需要大量的常识背景知识支持，BERT 学习到的是样本空间的特征、表征，可以看作是大型的文本匹配模型，而大量的背景常识是隐式且模糊的，很难在预训练数据中进行体现。其二是缺乏对语义的理解。模型并未理解数据中蕴含的语义知识，缺乏推理能力。在美团点评搜索场景中，需要首先对用户输入的 Query 进行意图识别，以确保召回结果的准确性。比如，对于“宫保鸡丁”和“宫保鸡丁酱料”两个 Query，二者的 BERT 语义表征非常接近，但是蕴含的搜索意图却截然不同。前者是菜品意图，即用户想去饭店消费，而后者则是商品意图，即用户想要从超市购买酱料。在这种场景下，BERT 模型很难像正常人一样做出正确的推理判断。

为了处理上述情况，我们尝试在 MT-BERT 预训练过程中融入知识图谱信息。知识图谱可以组织现实世界中的知识，描述客观概念、实体、关系。这种基于符号语义的计算模型，可以为 BERT 提供先验知识，使其具备一定的常识和推理能力。在我们团队之前的技术文章 [21] 中，介绍了 NLP 中心构建的大规模的餐饮娱乐知识图

谱——美团大脑。我们通过 Knowledge-aware Masking 方法将“美团大脑”的实体知识融入到 MT-BERT 预训练中。

BERT 在进行语义建模时，主要聚焦最原始的单字信息，却很少对实体进行建模。具体地，BERT 为了训练深层双向的语言表征，采用了 Masked LM（MLM）训练策略。该策略类似于传统的完形填空任务，即在输入端，随机地“遮蔽”掉部分单字，在输出端，让模型预测出这些被“遮蔽”的单字。模型在最初并不知道要预测哪些单字，因此它输出的每个单字的嵌入表示，都涵盖了上下文的语义信息，以便把被“掩盖”的单字准确的预测出来。

图 6 左侧展示了 BERT 模型的 MLM 任务。输入句子是“全聚德做的烤鸭久负盛名”。其中，“聚”，“的”，“久”3 个字在输入时被随机遮蔽，模型预训练过程中需要对这 3 个遮蔽位做出预测。

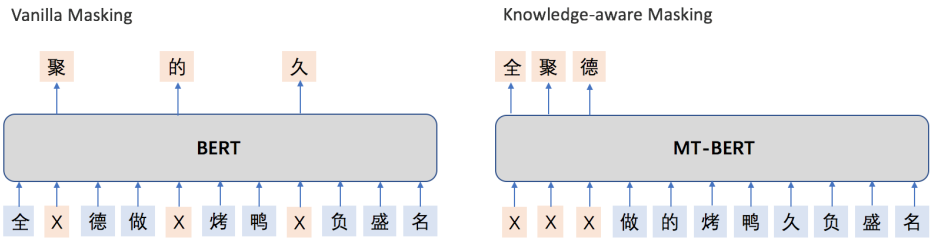


图 6　MT-BERT 默认 Masking 策略和 Whole Word Masking 策略对比

BERT 模型通过字的搭配（比如“全 X 德”），很容易推测出被“掩盖”字信息（“德”），但这种做法只学习到了实体内单字之间共现关系，并没有学习到实体的整体语义表示。因此，我们使用 Knowledge-aware Masking 的方法来预训练 MT-BERT。具体的做法是，输入仍然是字，但在随机”遮蔽”时，不再选择遮蔽单字，而是选择“遮蔽”实体对应的词。这需要我们在预训练之前，对语料做分词，并将分词结果和图谱实体对齐。图 6 右侧展示了 Knowledge-aware Masking 策略，“全聚德”被随机“遮蔽”。MT-BERT 需要根据“烤鸭”，“久负盛名”等信息，准确的预测出“全聚德”。通过这种方式，MT-BERT 可以学到“全聚德”这个实体的语义

表示，以及它跟上下文其他实体之间的关联，增强了模型语义表征能力。基于美团大脑中已有实体信息，我们在 MT-BERT 训练中使用了 Knowledge-aware Masking 策略，实验证明在细粒度情感分析任务上取得了显著提升。

表5 MT-BERT在细粒度情感分析数据集上效果

模型	Macro-F1
BERT (Vanilla masking)	72.04%
MT-BERT (Knowledge-aware Masking)	72.48%

模型轻量化

BERT 模型效果拔群，在多项自然语言理解任务上实现了最佳效果，但是由于其深层的网络结构和庞大的参数量，如果要部署上线，还面临很大挑战。以 Query 意图分类为例，我们基于 MT-BERT 模型微调了意图分类模型，协调工程团队进行了 1000QPS 压测实验，部署 30 张 GPU 线上卡参与运算，在线服务的 TP999 高达 50ms 之多，难以满足上线要求。

为了减少模型响应时间，满足上线要求，业内主要有三种模型轻量化方案。

- 低精度量化。在模型训练和推理中使用低精度（FP16 甚至 INT8、二值网络）表示取代原有精度（FP32）表示。
- 模型裁剪和剪枝。减少模型层数和参数规模。
- 模型蒸馏。通过知识蒸馏方法 [22] 基于原始 BERT 模型蒸馏出符合上线要求的小模型。

在美团点评搜索 Query 意图分类任务中，我们优先尝试了模型裁剪的方案。由于搜索 Query 长度较短（通常不超过 16 个汉字），整个 Sequence 包含的语义信息有限，裁剪掉几层 Transformer 结构对模型的语义表征能力不会有太大影响，同时又能大幅减少模型参数量和推理时间。经过实验验证，在微调过程中，我们将 MT-BERT 模型裁剪为 4 层 Transfomer 结构（MT-BERT-MINI，MBM），实验效果如图 7 所示。可以发现，Query 分类场景下，裁剪后的 MBM 没有产生较大影响。由

于减少了一些不必要的参数运算，在美食和酒店两个场景下，效果还有小幅的提升。

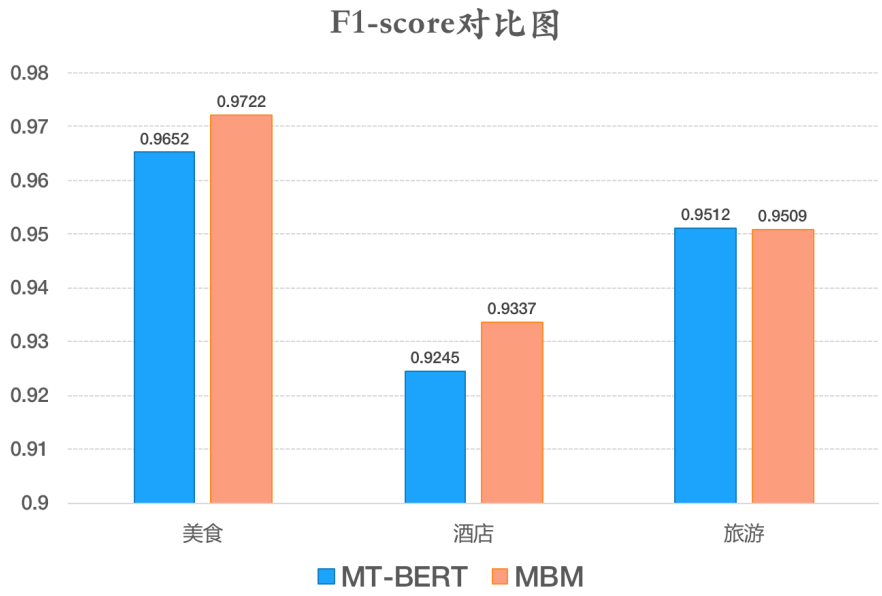


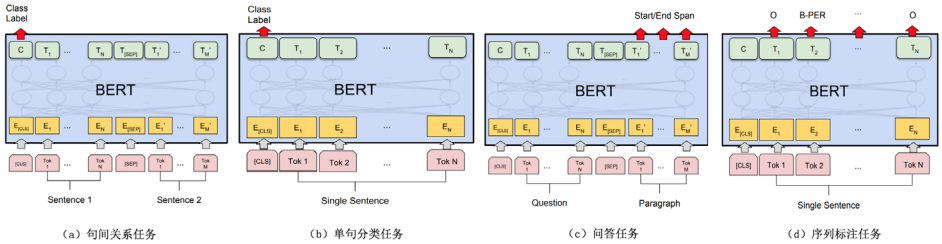
图 7　裁剪前后 MT-BERT 模型在 Query 意图分类数据集上 F1 对比

MBM 在同等压测条件下，压测服务的 TP999 达到了 12-14ms，满足搜索上线要求。除了模型裁剪，为了支持更多线上需求，我们还在进行模型蒸馏实验，蒸馏后的 6 层 MT-BERT 模型在大多数下游任务中都没有显著的效果损失。值得一提的是，BERT 模型轻量化是 BERT 相关研究的重要方向，最近 Google 公布了最新 ALBERT 模型（A Lite BERT）[23]，在减少模型参数量的同时在自然语言理解数据集 GLUE 上刷新了 SOTA。

图 8 展示了基于 BERT 模型微调可以支持的任务类型，包括句对分类、单句分类、问答（机器阅读理解）和序列标注任务。

1. 句对分类任务和单句分类任务是句子级别的任务。预训练中的 NSP 任务使得 BERT 中的“[CLS]”位置的输出包含了整个句子对（句子）的信息，我们利用其在有标注的数据上微调模型，给出预测结果。
2. 问答和序列标注任务都属于词级别的任务。预训练中的 MLM 任务使得每个

Token 位置的输出都包含了丰富的上下文语境以及 Token 本身的信息，我们对 BERT 的每个 Token 的输出都做一次分类，在有标注的数据上微调模型并给出预测。



（a）句间关系任务　（b）单句分类任务　（c）问答任务　（d）序列标注任务

图 8　BERT 微调支持的任务类型

基于 MT-BERT 的微调，我们支持了美团搜索和点评搜索的多个下游任务，包括单句分类任务、句间关系任务和序列标注任务等等。

单句分类

细粒度情感分析

美团点评作为生活服务平台，积累了大量真实用户评论。对用户评论的细粒度情感分析在深刻理解商家和用户、挖掘用户情感等方面有至关重要的价值，并且在互联网行业已有广泛应用，如个性化推荐、智能搜索、产品反馈、业务安全等领域。为了更全面更真实的描述商家各属性情况，细粒度情感分析需要判断评论文本在各个属性上的情感倾向（即正面、负面、中立）。为了优化美团点评业务场景下的细粒度情感分析效果，NLP 中心标注了包含 6 大类 20 个细粒度要素的高质量数据集，标注过程中采用严格的多人标注机制保证标注质量，并在 AI Challenger 2018 细粒度情感分析比赛中作为比赛数据集验证了效果，吸引了学术界和工业届大量队伍参赛。

针对细粒度情感分析任务，我们设计了基于 MT-BERT 的多任务分类模型，模型结构如图 9 所示。模型架构整体分为两部分：一部分是各情感维度的参数共享层（Share Layers），另一部分为各情感维度的参数独享层（Task-specific Layers）。其中参数共享层采用了 MT-BERT 预训练语言模型得到文本的上下文表征。MT-BERT 依赖其深层网络结构以及海量数据预训练，可以更好的表征上下文信息，尤其

擅长提取深层次的语义信息。 参数独享层采用多路并行的 Attention+Softmax 组合结构，对文本在各个属性上的情感倾向进行分类预测。通过 MT-BERT 优化后的细粒度情感分析模型在 Macro-F1 上取得了显著提升。

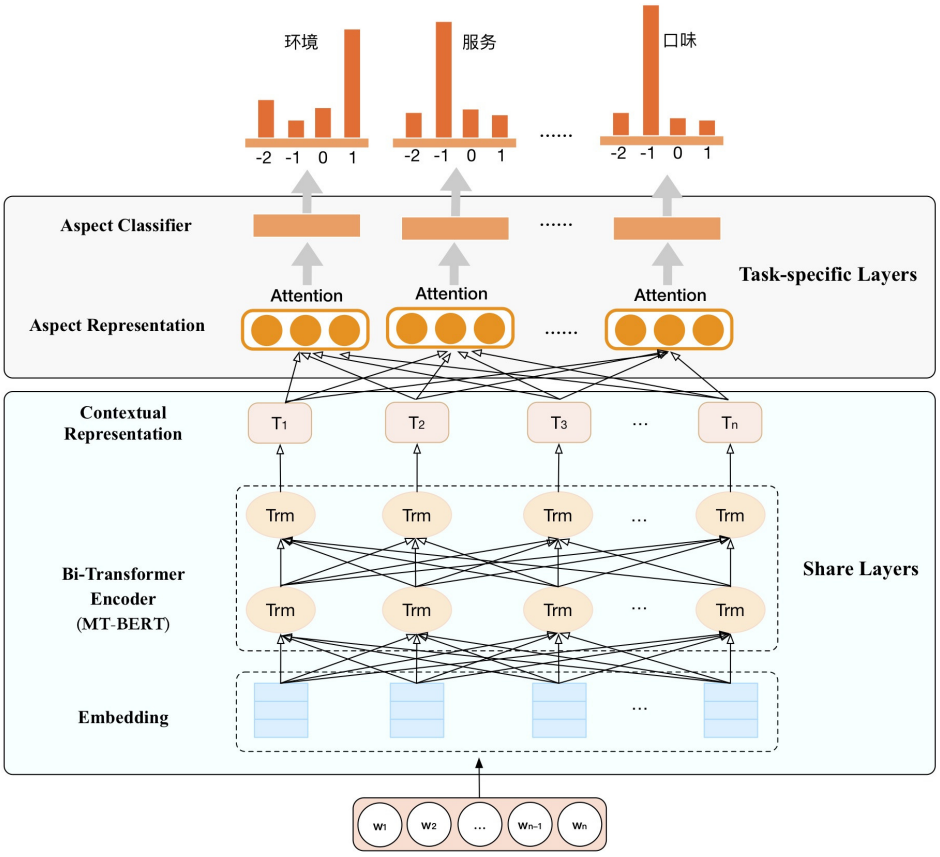


图 9　基于 MT-BERT 的多任务细粒度情感分析模型架构

细粒度情感分析的重要应用场景之一是大众点评的精选点评模块，如图 10 所示。精选点评模块作为点评 App 用户查看高质量评论的入口，其中精选点评标签承载着结构化内容聚合的作用，支撑着用户高效查找目标 UGC 内容的需求。细粒度情感分析能够从不同的维度去挖掘评论的情感倾向。基于细粒度情感分析的情感标签能够较好地帮助用户筛选查看，同时外露更多的 POI 信息，帮助用户高效的从评论中获取消费指南。

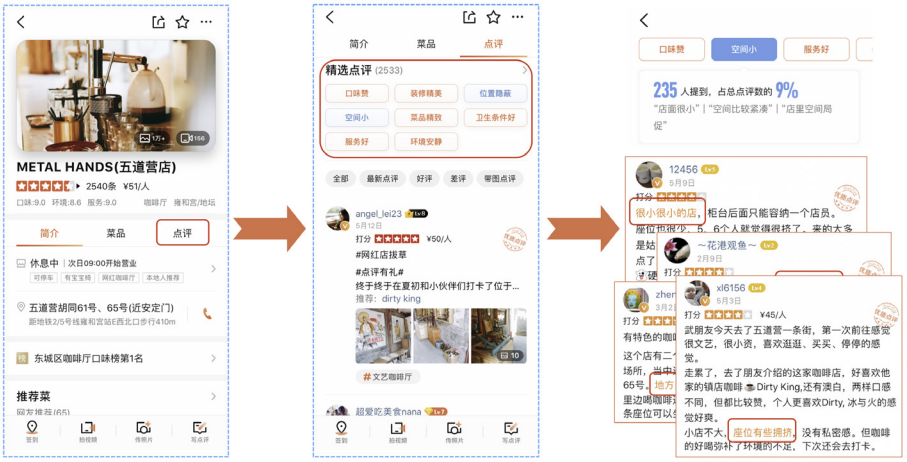
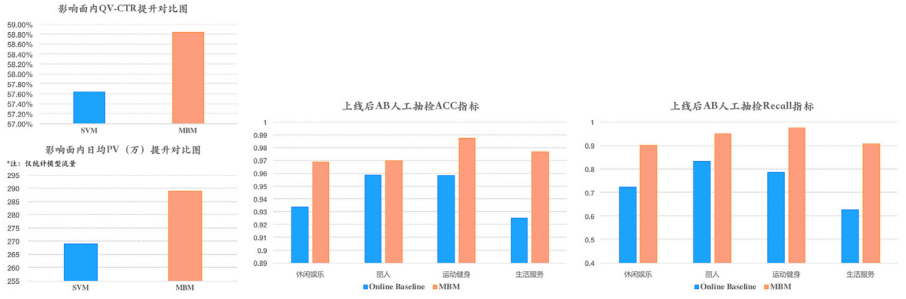


图 10　大众点评精选点评模块产品形态

Query 意图分类

在美团点评的搜索架构中，Deep Query Understanding（DQU）都是重要的前置模块之一。对于用户 Query，需要首先对用户搜索意图进行识别，如美食、酒店、演出等等。我们跟内部的团队合作，尝试了直接使用 MT-BERT 作为 Query 意图分类模型。为了保证模型在线 Inference 时间，我们使用裁剪后的 4 层 MT-BERT 模型（MT-BERT-MINI，MBM 模型）上线进行 Query 意图的在线意图识别，取得的业务效果如图 11 所示：



(a)线上收益指标（以美食频道为例）　　(b) 线下准召指标（以部分频道为例）

图 11　MBM 模型的业务效果

同时对于搜索日志中的高频 Query，我们将预测结果以词典方式上传到缓存，进一步减少模型在线预测的 QPS 压力。MBM 累计支持了美团点评搜索 17 个业务频道的 Query 意图识别模型，相比原有模型，均有显著的提升，每个频道的识别精确度都达到 95% 以上。MBM 模型上线后，提升了搜索针对 Query 文本的意图识别能力，为下游的搜索的召回、排序及展示、频道流量报表、用户认知报表、Bad Case 归因等系统提供了更好的支持。

推荐理由场景化分类

推荐理由是点评搜索智能中心数据挖掘团队基于大众点评 UGC 为每个 POI 生产的自然语言可解释性理由。对于搜索以及推荐列表展示出来的每一个商家，我们会用一句自然语言文本来突出商家的特色和卖点，从而让用户能够对展示结果有所感知，“知其然，更知其所以然”。近年来，可解释的搜索系统越来越受到关注，给用户展示商品或内容的同时透出解释性理由，正在成为业界通行做法，这样不仅能提升系统的透明度，还能提高用户对平台的信任和接受程度，进而提升用户体验效果。在美团点评的搜索推荐场景中，推荐理由有着广泛的应用场景，起到解释展示、亮点推荐、场景化承载和个性化体现的重要作用，目前已经有 46 个业务方接入了推荐理由服务。

对于不同的业务场景，对推荐理由会有不同的要求。在外卖搜索场景下，用户可能更为关注菜品和配送速度，不太关注餐馆的就餐环境和空间，这种情况下只保留符合外卖场景的推荐理由进行展示。同样地，在酒店搜索场景下，用户可能更为关注酒店特色相关的推荐理由（如交通是否方便，酒店是否近海近景区等）。

我们通过内部合作，为业务方提供符合不同场景需求的推荐理由服务。推荐理由场景化分类，即给定不同业务场景定义，为每个场景标注少量数据，我们可以基于 MT-BERT 进行单句分类微调，微调方式如图 8(b) 所示。

图 12　外卖和酒店场景下推荐理由

句间关系

句间关系任务是对两个短语或者句子之间的关系进行分类，常见句间关系任务如自然语言推理（Natural Language Inference, NLI）、语义相似度判断（Semantic Textual Similarity，STS）等。

Query 改写是在搜索引擎中对用户搜索 Query 进行同义改写，改善搜索召回结果的一种方法。在美团和点评搜索场景中，通常一个商户或者菜品会有不同的表达方式，例如“火锅”也称为“涮锅”。有时不同的词语表述相同的用户意图，例如“婚纱摄影”和“婚纱照”，“配眼镜”和“眼镜店”。Query 改写可以在不改变用户意图的情况下，尽可能多的召回满足用户意图的搜索结果，提升用户的搜索体验。为了

减少误改写，增加准确率，需要对改写后 Query 和原 Query 做语义一致性判断，只有语义一致的 Query 改写对才能上线生效。Query 语义一致性检测属于 STS 任务。我们通过 MT-BERT 微调任务来判断改写后 Query 语义是否发生漂移，微调方式如图 8(a) 所示，把原始 Query 和改写 Query 构成句子对，即“[CLS] text_a [SEP] text_b [SEP]”的形式，送入到 MT-BERT 中，通过“[CLS]”判断两个 Query 之间关系。实验证明，基于 MT-BERT 微调的方案在 Benchmark 上准确率和召回率都超过原先的 XGBoost 分类模型。

序列标注

序列标注是 NLP 基础任务之一，给定一个序列，对序列中的每个元素做一个标记，或者说给每一个元素打一个标签，如中文命名实体识别、中文分词和词性标注等任务都属于序列标注的范畴。命名实体识别（Named Entity Recognition，NER），是指识别文本中具有特定意义的实体，主要包括人名、地名、机构名、专有名词等，以及时间、数量、货币、比例数值等文字。

在美团点评业务场景下，NER 主要需求包括搜索 Query 成分分析，UGC 文本中的特定实体（标签）识别 / 抽取，以及客服对话中的槽位识别等。NLP 中心和酒店搜索算法团队合作，基于 MT-BERT 微调来优化酒店搜索 Query 成分分析任务。酒店 Query 成分分析任务中，需要识别出 Query 中城市、地标、商圈、品牌等不同成分，用于确定后续的召回策略。

在酒店搜索 Query 成分分析中，我们对标签采用“BME”编码格式，即对一个实体，第一个字需要预测成实体的开始 B，最后一个字需要预测成实体的结束 E，中间部分则为 M。以图 13 中酒店搜索 Query 成分分析为例，对于 Query“北京昆泰酒店”，成分分析模型需要将“北京”识别成地点，而“昆泰酒店”识别成 POI。MT-BERT 预测高频酒店 Query 成分后通过缓存提供线上服务，结合后续召回策略，显著提升了酒店搜索的订单转化率。

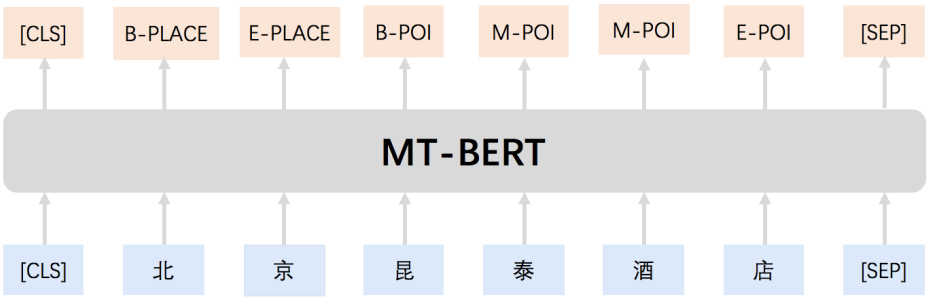


图 13　酒店 Query 的成分分析

一站式 MT-BERT 训练和推理平台建设

为了降低业务方算法同学使用 MT-BERT 门槛，我们开发了 MT-BERT 一站式训练和推理平台，一期支持短文本分类和句间关系分类两种任务，目前已在美团内部开放试用。

基于一站式平台，业务方算法同学上传业务训练数据和选择初始 MT-BERT 模型之后，可以提交微调任务，微调任务会自动分配到 AFO 集群空闲 GPU 卡上自动运行和进行效果验证，训练好的模型可以导出进行部署上线。

融入知识图谱的 MT-BERT 预训练

正如前文所述，尽管在海量无监督语料上进行预训练语言模型取得了很大的成功，但其也存在着一定的不足。BERT 模型通过在大量语料的训练可以判断一句话是否通顺，但是却不理解这句话的语义，通过将美团大脑等知识图谱中的一些结构化先验知识融入到 MT-BERT 中，使其更好地对生活服务场景进行语义建模，是需要进一步探索的方向。

MT-BERT 模型的轻量化和小型化

MT-BERT 模型在各个 NLU 任务上取得了惊人的效果，由于其复杂的网络结构和庞大的参数量，在真实工业场景下上线面临很大的挑战。如何在保持模型效果的前提下，精简模型结构和参数已经成为当前热门研究方向。我们团队在低精度量化、模型裁剪和知识蒸馏上已经做了初步尝试，但是如何针对不同的任务类型选择最合适的

模型轻量化方案，还需要进一步的研究和探索。

参考文献

[1] Peters, Matthew E., et al. “Deep contextualized word representations.” arXiv preprint arXiv:1802.05365 (2018).

[2] Howard, Jeremy, and Sebastian Ruder. “Universal language model fine-tuning for text classification.” arXiv preprint arXiv:1801.06146 (2018).

[3] Vaswani, Ashish, et al. “Attention is all you need.” Advances in neural information processing systems. 2017.

[4] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving Language Understanding by Generative Pre-Training. Technical report, OpenAI.

[5] Devlin, Jacob, et al. “Bert: Pre-training of deep bidirectional transformers for language understanding.” arXiv preprint arXiv:1810.04805 (2018).

[6] Ming Zhou. “The Bright Future of ACL/NLP.” Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. (2019).

[7] Deng, Jia, et al. “Imagenet: A large-scale hierarchical image database.” 2009 IEEE conference on computer vision and pattern recognition. Ieee, (2009).

[8] Girshick, Ross, et al. “Rich feature hierarchies for accurate object detection and semantic segmentation.” Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

[9] Mikolov, Tomas, et al. “Distributed representations of words and phrases and their compositionality.” Advances in neural information processing systems. 2013.

[10] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In EMNLP.

[11] Oren Melamud, Jacob Goldberger, and Ido Dagan.2016. context2vec: Learning generic context embedding with bidirectional lstm. In CoNLL.

[12] Hochreiter, Sepp, and J ü rgen Schmidhuber. “Long short-term memory.” Neural computation 9.8 (1997): 1735-1780.

[13] 张俊林 . 从 Word Embedding 到 BERT 模型—自然语言处理中的预训练技术发展史 . https://zhuanlan.zhihu.com/p/49271699

[14] Sebastion Ruder. “NLP’s ImageNet moment has arrived.” http://ruder.io/nlp-imagenet/. (2019)

[15] Liu, Yinhan, et al. “Roberta: A robustly optimized BERT pretraining approach.” arXiv preprint arXiv:1907.11692 (2019).

[16] 郑坤 . 使用 TensorFlow 训练 WDL 模型性能问题定位与调优 . https://tech.meituan.com/2018/04/08/tensorflow-performance-bottleneck-analysis-on-hadoop.html

[17] Uber. “Meet Horovod: Uber’s Open Source Distributed Deep Learning Framework for TensorFlow”. https://eng.uber.com/horovod/

[18] Goyal, Priya, et al. “Accurate, large minibatch sgd: Training imagenet in 1 hour.” arXiv preprint arXiv:1706.02677 (2017).

[19] Baidu. https://github.com/baidu-research/baidu-allreduce
[20] Micikevicius, Paulius, et al. “Mixed precision training.” arXiv preprint arXiv:1710.03740 (2017).
[21] 仲远，富峥等 . 美团餐饮娱乐知识图谱——美团大脑揭秘 . https://tech.meituan.com/2018/11/22/meituan-brain-nlp-01.html
[22] Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean. “Distilling the knowledge in a neural network.” arXiv preprint arXiv:1503.02531 (2015).
[23] Google. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. https://openreview.net/pdf?id=H1eA7AEtvS. (2019)

作者简介

杨扬，佳昊，礼斌，任磊，峻辰，玉昆，张欢，金刚，王超，王珺，富峥，仲远，都来自美团搜索与 NLP 部。

招聘信息

搜索与 NLP 部是美团人工智能技术研发的核心团队，致力于打造高性能、高扩展的搜索引擎和领先的自然语言处理核心技术和服务能力，依托搜索排序、NLP（自然语言处理）、Deep Learning（深度学习）、Knowledge Graph（知识图谱）等技术，处理美团海量文本数据，打通餐饮、旅行、休闲娱乐等本地生活服务各个场景数据，不断加深对用户、场景、查询和服务的理解，高效地支撑形态各样的生活服务搜索，解决搜索场景下的多意图、个性化、时效性问题，给用户极致的搜索体验，构建美团知识图谱，搭建通用 NLP Service，为美团各项业务提供智能的文本语义理解服务。我们的团队既注重 AI 技术的落地，也开展中长期的搜索、NLP 及知识图谱基础研究。目前项目及业务包括搜索引擎研发、知识图谱、智能客服、语音语义搜索、文章评论语义理解、智能助理等。

美团搜索与 NLP 部诚招智能对话算法专家、推荐算法专家、知识图谱算法专家、NLP 算法专家、数据挖掘专家，以及搜索引擎架构师、高级后台研发工程师、前端技术开发资深工程师、测试工程师、数据工程师等众多岗位，欢迎有兴趣的同学，投递简历至：tech@meituan.com（邮件标题注明：岗位名称 + 美团搜索与 NLP 部）。

深度学习在搜索业务中的探索与实践

艺涛

本文根据美团高级技术专家翟艺涛在 2018 QCon 全球软件开发大会上的演讲内容整理而成，内容有修改。

引言

2018 年 12 月 31 日，美团酒店单日入住间夜突破 200 万，再次创下行业的新纪录，而酒店搜索在其中起到了非常重要的作用。本文会首先介绍一下酒店搜索的业务特点，作为 O2O 搜索的一种，酒店搜索和传统的搜索排序相比存在很大的不同。第二部分介绍深度学习在酒店搜索 NLP 中的应用。第三部分会介绍深度排序模型在酒店搜索的演进路线，因为酒店业务的特点和历史原因，美团酒店搜索的模型演进路线可能跟大部分公司都不太一样。最后一部分是总结。

酒店搜索的业务特点

最大的连接器

美团的使命是帮大家“Eat Better，Live Better”，所做的事情就是连接人与服务。用户在美团平台可以找到他们所需要的服务，商家在美团可以售卖自己提供

的服务，而搜索在其中扮演的角色就是“连接器”。大部分用户通过美团 App 找酒店是从搜索开始的，搜索贡献了大部分的订单，是最大的流量入口。在美团首页点击“酒店住宿”图标，就会进入上图右侧的搜索入口，用户可以选择城市和入住时间并发起搜索。

查询引导

推荐

搜索&排序

酒店搜索技术团队的工作不仅有搜索排序，还有查询引导、推荐等工作，查询引导如搜索智能提示、查询纠错等。之所以还有推荐的工作，是因为很多用户在发起搜索时不带查询词，本质上属于推荐，此外还有特定场景下针对少无结果的推荐等。本文主要介绍搜索排序这方面的工作。

不同搜索对比

现在，大家对搜索都很熟悉，常见的有网页搜索，比如 Google、百度、搜狗等；商品搜索，像天猫、淘宝、京东等；还有就是 O2O（Online To Offline）的搜索，典型的就是酒店的搜索。虽然都是搜索，但是用户使用搜索的目的并不相同，包括找信息、找商品、找服务等等，不同搜索之间也存在很大的差别。

	酒店搜索（O2O）	网页搜索	商品搜索
目标	交易	相关性	交易
个性化	高	低	高
结构化数据	是	否	是
位置约束	高	低	低
供给约束	区域	无	全国

上图对不同搜索进行了简单对比，可以从 5 个维度展开。首先是目标维度。因为用户是来找信息，网页搜索重点是保证查询结果和用户意图的相关性，而在商品搜索和酒店搜索中，用户的主要目的是查找商品或服务，最终达成交易，目标上有较大区别。用户使用不同搜索的目的不同，从而导致不同搜索对个性化程度的要求不同。交易属性的搜索，包括商品搜索和酒店搜索，对个性化程度的要求都比较高，因为不同用户的消费水平不同，偏好也不一样。

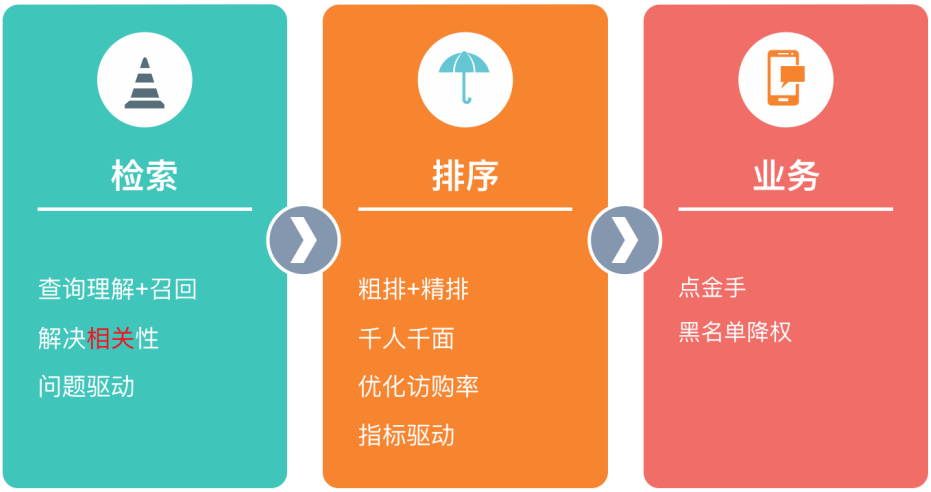
在技术层面上，也存在很多不同点。网页搜索会索引全网的数据，这些数据不是它自己生产，数据来源非常多样，包括新闻、下载页、视频页、音乐页等各种不同的形态，所以整个数据是非结构化的，差异也很大。这意味着网页搜索需要拥有两种技术能力，数据抓取能力和数据解析能力，它们需要抓取网页并解析形成结构化数据。在这个层面上，酒店搜索和商品搜索相对就“幸福”一些，因为数据都是商家提交的结构化数据，相对来说更加规范。

此外，酒店作为一种 O2O 的服务，用户在线上（Online）下单，最终需要到线下（Offline）去消费，所以就有一个位置上的约束，而位置的约束也就导致出现供给侧的约束，供给只能在某个特定位置附近。比如北京大学方圆几公里之内的酒店。这两点约束在网页搜索和商品搜索中就不用考虑，网页可以无限次的进行阅读。商品搜索得益于快递业的快速发展，在北京也可以买到来自浙江的商品，供给侧的约束比较小。

介绍完不同搜索产品的特点，接下来看不同搜索产品的优化目标。通用搜索的优化目标是相关性，评价指标是 DCG、NDCG、MAP 等这些指标，要求查询结果和用户意图相关。对商品搜索来说，不同电商平台的优化目标不太一样，有的目标是最大化 GMV，有的目标是最大化点击率，这些在技术上都可以实现。

而对酒店搜索而言，因为它属于 O2O 的业务形态，线上下单，线下消费，这就要求搜索结果必须和用户的查询意图“强相关”。这个“强相关”包括两层含义，显性相关和隐性相关。举个例子，用户搜索“北京大学”，那么他的诉求很明确，就是要找“北京大学”附近的酒店，这种属于用户明确告诉平台自己的位置诉求。但是，如果用户在本地搜索“七天”，即使用户没有明确说明酒店的具体位置，我们也知道，用户可能想找的是距离自己比较近的“七天酒店”，这时候就需要建模用户的隐性位置诉求。

美团是一个交易平台，大部分用户使用美团是为了达成交易，所以要优化用户的购买体验。刻画用户购买体验的核心业务指标是访购率，用来描述用户在美团是否顺畅的完成了购买，需要优化访购率这个指标。总结一下，酒店搜索不仅要解决相关性，尽量优化用户购买体验、优化访购率等指标，同时还要照顾到业务诉求。



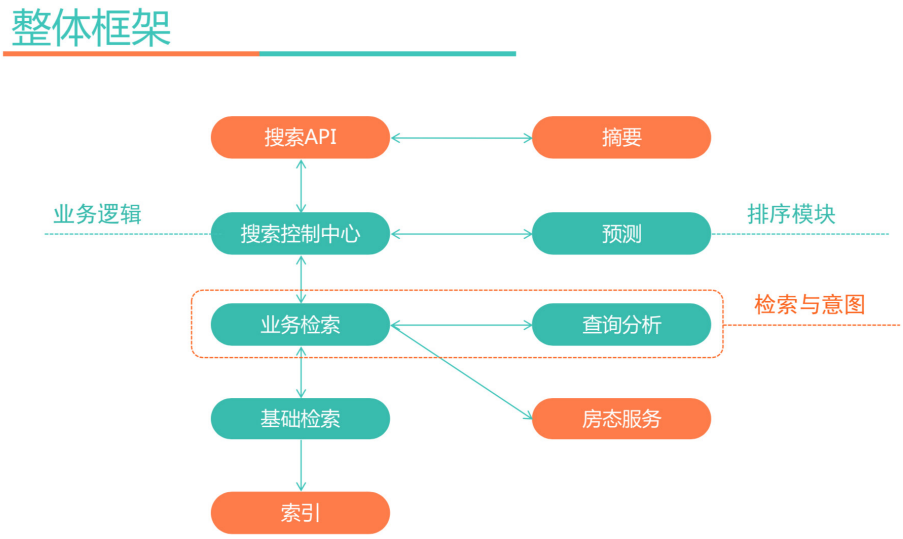
根据上面的分析，酒店搜索的整个搜索框架就可以拆分成三大模块：检索、排序以及业务规则。检索层包括查询理解和召回两部分，主要解决相关性问题。查询理解做的事情就是理解用户意图，召回根据用户意图来召回相关的酒店，两者强耦合，需要放在一起。检索的核心是语义理解，比如用户搜索“北京大学”，平台就知道用户想找的是“北京大学附近的酒店”，所以这个模块的优化方式是问题驱动，不断地发现问题、解决问题来进行迭代。

接下来，从检索模块检索出来的酒店都已经是满足用户需求的酒店了。还是上面“北京大学”的那个例子，检索模块已经检索出来几百家“北京大学”附近的酒店，这些都是和用户的查询词“北京大学”相关的，怎么把用户最有可能购买的酒店排到前面呢？这就是排序模块要做的事情。

排序模块使用机器学习和深度学习的技术提供“千人千面”的排序结果，如果是经常预定经济连锁型酒店的用户，排序模块就把经济连锁型酒店排到前面。针对消费水平比较高，对酒店要求比较高的用户，排序模块就把高档酒店排到前面，对每个用户都可以做到个性化定制。排序属于典型的技术驱动模块，优化目标是访购率，用这个技术指标驱动技术团队不断进行迭代和优化。

最后是业务层面，比如有些商家会在美团上刷单作弊，针对这些商家需要做降权处理。

整体框架

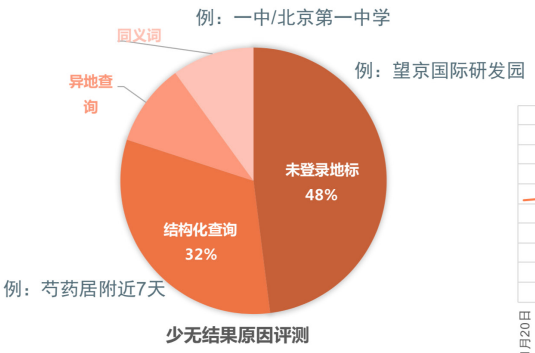


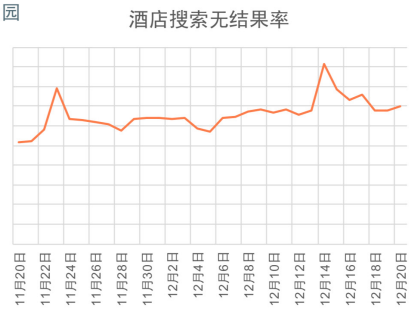
上图是搜索的整体框架，这里详细描述下调用过程：

- 搜索 API 负责接收用户的查询词并发送给搜索控制中心。
- 控制中心把接收到的查询请求发送到检索与意图模块，搜索词会先经过查询分析模块做用户的查询意图分析，分析完之后，会把用户的查询意图分析结果传回去给业务检索模块，业务检索模块根据意图识别结果形成查询条件，然后去基础检索端查询结果。
- 基础检索访问索引得到查询结果后，再把结果返回给上层。
- 业务检索模块获取基础的检索结果后，会调用一些外部服务如房态服务过滤一些满房的酒店，再把结果返回给控制中心。
- 此时，控制中心得到的都是和用户查询意图强相关的结果，这时就需要利用机器学习技术做排序。通过预测模块对每个酒店做访购率预测，控制中心获取预测模块的排序结果后，再根据业务逻辑做一些调整，最终返回结果给搜索 API。

可以看到，模块划分和前文描述的思想一致，检索模块主要解决用户意图识别和召回问题，也就是解决相关性。预测模块做访购率预测，业务逻辑放在搜索控制中心实现。接下来会介绍一下意图理解和排序模块中涉及的一些深度学习技术。

查询理解：问题

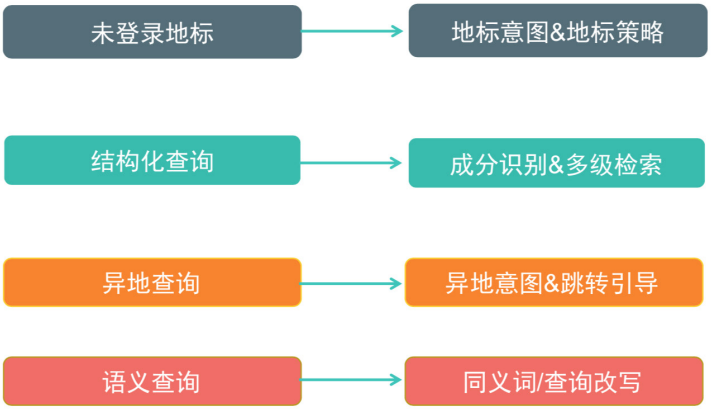




先来看下查询理解的问题，这个模块通过数据分析和 Case 分析，不断的发现问题、解决问题来迭代优化。之前的评测发现少无结果的原因，主要包括以下几种：

- 地标词：比如用户搜索“望京国际研发园”，但是后台没有一家酒店包含“望京国际研发园”这几个字，其实用户想找的是望京国际研发园附近的酒店。
- 结构化查询：比如芍药居附近 7 天，酒店描述信息中没有“附近”这个词，搜索体验就比较差。这种需要对查询词做成分识别，丢掉不重要的词，并且对不用类别的 Term 走不同的检索域。
- 异地查询：用户在北京搜索“大雁塔”没有结果，其实用户的真实意图是西安大雁塔附近的酒店，这种需要做异地需求识别并进行异地跳转。
- 同义词：在北京搜索“一中”和搜索“北京第一中学”，其实都是同一个意思，需要挖掘同义词。

解决方案



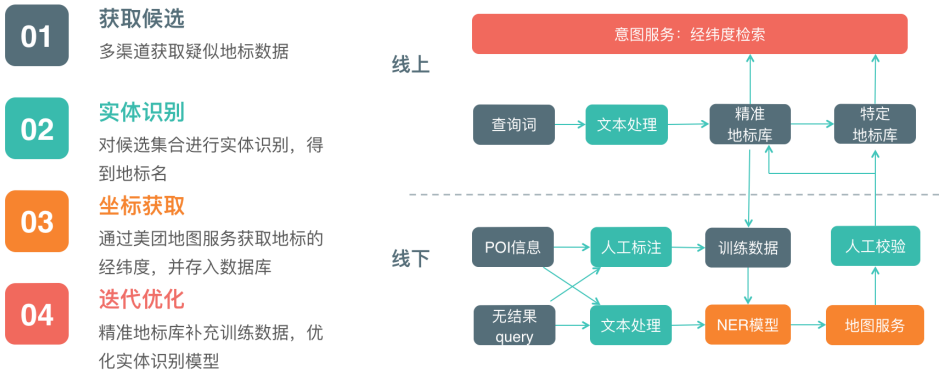
针对这几类问题，我们分别作了以下工作：

- 针对地标词问题，提供地标意图识别和地标策略，把地标类别的查询词改成按经纬度进行画圈检索。
- 针对结构化查询的问题，我们对查询词做了成分识别，设计了少无结果时的多级检索架构。
- 针对异地查询的问题，做异地意图识别和异地的跳转引导。
- 针对语义查询的问题，做同义词和查询改写。

这里的每一个模块都用到了机器学习和深度学习的技术，本文挑选两个酒店搜索中比较特殊的问题进行介绍。

地标问题是 O2O 搜索的一个典型问题，在网页搜索和商品搜索中都较少出现此类问题。当用户搜索类似“望京国际研发园”这种查询词的时候，因为搜索的相关性是根据文本计算的，需要酒店描述中有相关文字，如果酒店的描述信息中没有这个词，那就检索不出来。比如昆泰酒店，虽然就在望京国际研发园旁边，但是它的描述信息中并没有出现“望京国际研发园”，所以就无法检索出来，这会导致用户体验较差。

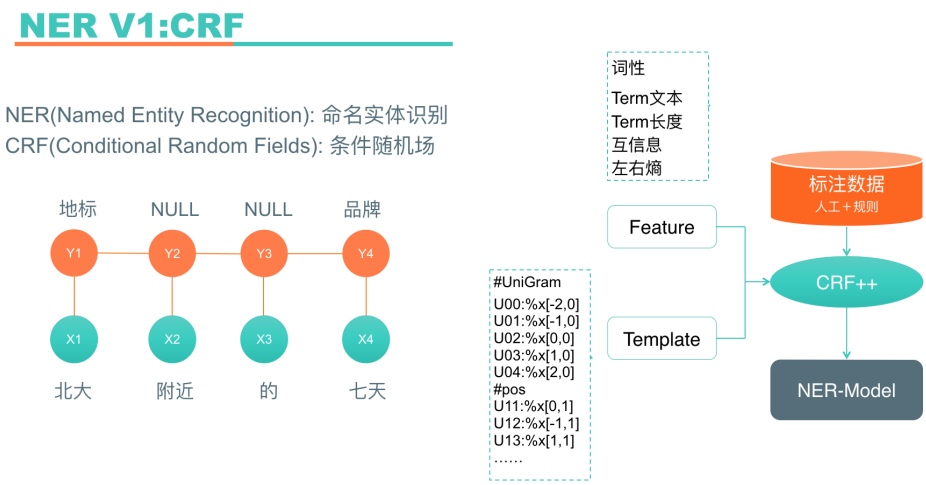
经过分析，我们发现有一类查询词是针对特定地点的搜索，用户的诉求是找特定地点附近的酒店，这种情况下走文本匹配大概率是没有结果的。这个问题的解法是针对这种类型的查询词，从“文本匹配”改成“坐标匹配”，首先分析查询词是不是有地标意图，如果是的话就不走文本匹配了，改走坐标匹配，检索出来这个坐标附近的酒店就可以了。这时就产生了两个问题：第一，怎么确定哪些查询词有地标意图；第二，怎么获取经纬度信息。



针对这个问题，我们做了地标策略，步骤如下：

- 多渠道获取可能包含地标词的候选集，这些候选集包括用户少无结果的查询词，以及一些酒店提供的描述信息。
- 对候选集合进行命名实体识别（NER，Named Entity Recognition），可以得到各个命名实体的类型，标识为“地标”类型的就是疑似地标词。
- 把疑似地标词放到美团地图服务中获取经纬度，经过人工校验无误后，存入线上数据库中；线上来查询请求时，先会去匹配精准地标库，如果匹配成功，说明这个查询词是地标意图，这时就不走文本检索了，直接在意图服务层走经纬度检索。
- 经过人工校验的精准地标库补充到 NER 模型的训练数据中，持续优化 NER 模型。

这里提到了 NER 模型，下面对它做一下详细的介绍。

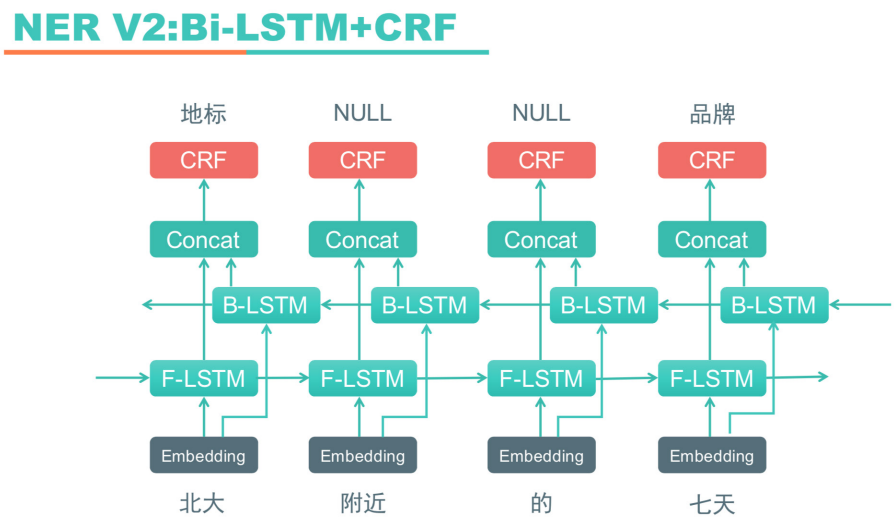


John Lafferty et al. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.ICML2001

NER 是命名实体识别，是机器学习中的序列标注问题，比如输入“北大附近的七天”，就会标注出来每个词的成分，这里“北大”是地标，“七天”是酒店品牌。这里的类别是根据业务特点自己定义的，酒店业务中有地标、品牌、商圈等不同的类别。与分类问题相比，序列标注问题中当前的预测标签不仅与当前的输入特征相关，还与前后的预测标签相关，即预测标签序列之间有强相互依赖关系。

解决序列标注问题的经典模型是 CRF（Conditional Random Field，条件随机场），也是我们刚开始尝试的模型。条件随机场可以看做是逻辑回归的序列化版本，逻辑回归是用于分类的对数线性模型，条件随机场是用于序列化标注的对数线性模型，可以看做是考虑了上下文的分类模型。

机器学习问题的求解就是“数据 + 模型 + 特征”，数据方面先根据业务特点定义了几种实体类别，然后通过“人工 + 规则”的方法标注了一批数据。特征方面提取了包括词性、Term 文本特征等，还定义了一些特征模板，特征模板是 CRF 中人工定义的一些二值函数，通过这些二值函数，可以挖掘命名实体内部以及上下文的构成特点。标注数据、模型、特征都有了，就可以训练 CRF 模型，这是线上 NER 问题的第一版模型。

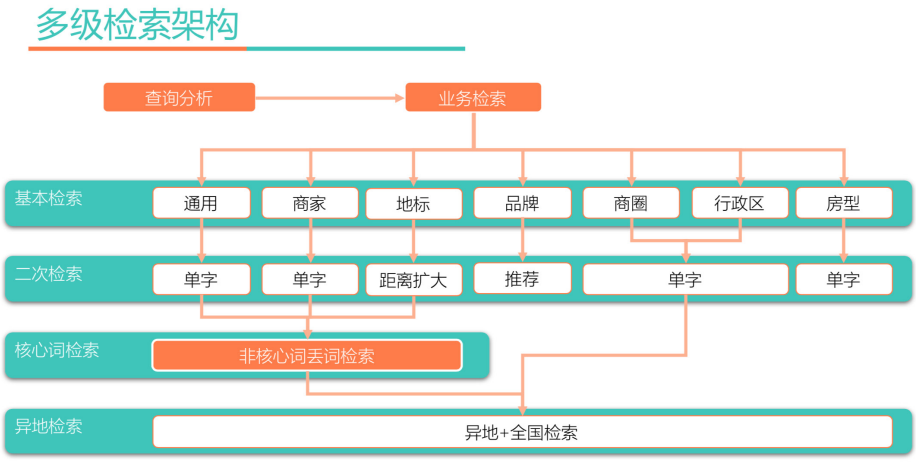


Guillaume Lample et al Neural architectures for named entity recognition. NAACL2016

随着深度学习的发展，用 Word Embedding 词向量作为输入，叠加神经网络单元的方法渐渐成为 NLP 领域新的研究方向。基于双向 LSTM（Long Short-Term Memory）+CRF 的方法成为 NER 的主流方法，这种方法采用双向 LSTM 单元作为特征提取器替代原有的人工特征，不需要专门的领域知识，框架也通用。Embedding 输入也有多种形式，可以是词向量，可以是字向量，也可以是字向量和词向量的拼接。

我们尝试了双向 LSTM+CRF，并在实际应用中做了些改动：由于在 CRF 阶段已经积累了一批人工特征，实验发现把这些特征加上效果更好。加了人工特征的双向 LSTM+CRF 是酒店搜索 NER 问题的主模型。

当然，针对 LSTM+CRF 的方法已经有了很多的改进，比如还有一种 NER 的方法是融合 CNN+LSTM+CRF，主要改进点是多了一个 CNN 模块来提取字级别的特征。CNN 的输入是字级别的 Embedding，通过卷积和池化等操作来提取字级别的特征，然后和词的 Embedding 拼接起来放入 LSTM。这种方法在两个公开数据集上面取得了最好的结果，也是未来尝试的方向之一。



为了解决少无结果的问题，我们设计了多级检索架构，如上图所示，主要分 4 个层次：基本检索、二次检索、核心词检索和异地检索。

- 基本检索会根据查询词的意图选择特定的检索策略，比如地标意图走经纬度检索，品牌意图只检索品牌域和商家名。
- 基本检索少无结果会进行二次检索，二次检索也是分意图的，不同意图类型会有不同的检索策略，地标意图是经纬度检索的，二次检索的时候就需要扩大检索半径；品牌意图的查询词，因为很多品牌在一些城市没有开店，比如香格里拉在很多小城市并没有开店，这时比较好的做法，是推荐给用户该城市最好的酒店。
- 如果还是少无结果，会走核心词检索，只保留核心词检索一遍。丢掉非核心词有多种方式，一种是删除一些运营定义的无意义词，一种是保留 NER 模型识别出来的主要实体类型。此外还有一个 TermWeight 的模型，对每个词都有一个重要性的权重，可以把一些不重要的词丢掉。
- 在还没有结果的情况下，会选择”异地 + 全国“检索，即更换城市或者在全国范围内进行检索。

多级检索架构上线后，线上的无结果率就大幅度降低了。

排序

排序其实是一个典型的技术问题，业界应用比较广泛的有广告排序和推荐排序，广告排序比如 Google 和百度的关键字广告排序，今日头条、腾讯的展示广告排序。推荐排序比如快手、抖音这些短视频平台，以及各大 App、浏览器的信息流。广告排序和推荐排序优化的目标都是点击率，技术栈也比较相似，包括 LR/FTRL、FM/FFM、GBDT、DNN 等模型。

跟以上两种排序应用相比，酒店排序有自己的业务特点，因为美团酒店具有 LBS 属性和交易属性，天生自带很多连续特征，如酒店价格、酒店评分、酒店离用户的距离等，这些连续特征是决定用户购买行为的最重要因素。优化目标也不一样，大部分场景下酒店搜索的优化目标是访购率，部分场景下优化目标是点击率。在技术层面，酒店排序整体的技术栈和广告、推荐比较相似，都可以使用 LR/FTRL、FM/FFM、GBDT、DNN 等模型。

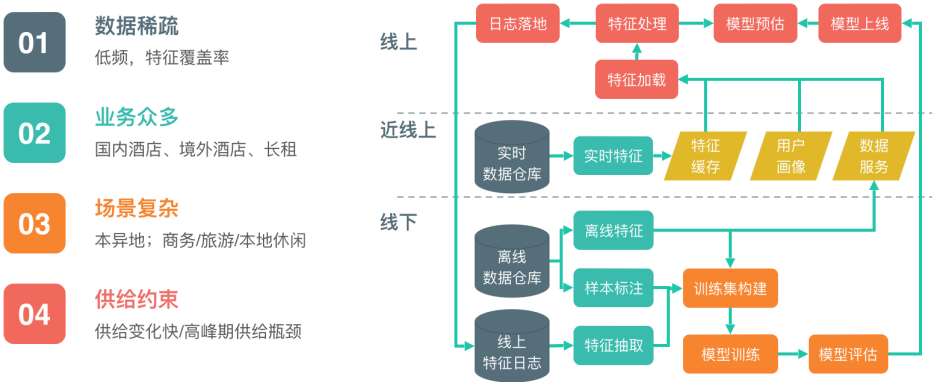
面临的挑战

具体到酒店排序工作，我们面临一些不一样的挑战，主要包括以下 4 点：

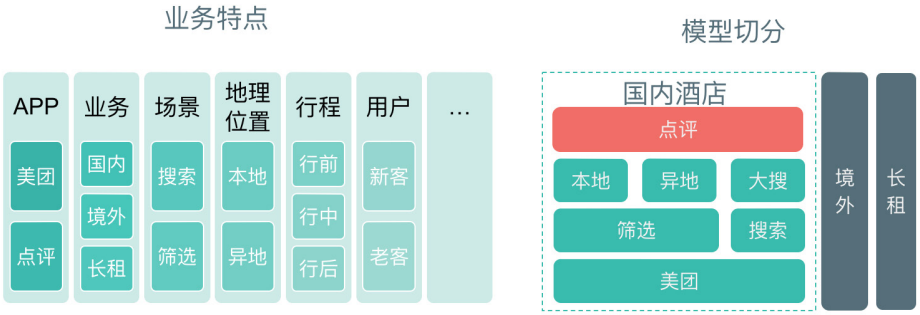
1. 数据稀疏。住酒店本身是一种低频行为，大部分用户一年也就住一两次，导致很多特征的覆盖率比较低。
2. 业务众多。美团酒店包括国内酒店业务、境外酒店业务，以及长租、钟点房

等业务，同时有美团和点评两个不同的 App。

3. 场景复杂。按照用户的位置可以分成本地和异地，按照用户的诉求可以分成商务、旅游、本地休闲等几大类，这些用户之间差异很明显。比如商务用户会有大量复购行为，典型例子是美团员工的出差场景，美团在上海和北京各有一个总部，如果美团的同学去上海出差，大概率会在公司差旅标准内选一家离公司近的酒店，从而会在同一家酒店产生大量的复购行为；但是如果是一个旅游用户，他就很少反复去同一个地方。
4. 供给约束。酒店行业供给的变化很快，一个酒店只有那么多房间，一天能提供的间夜量是固定的，全部订出的话，用户提价也不会提供新的房间，这种情况在劳动节、国庆这种节假日特别明显。



上图右侧是排序的整体架构图，分为线下、线上和近线上三个部分。在线下部分，主要做离线的模型调优和评估，线上部分做预测。这里比较特别的是近线上部分，我们在实时层面做了大量的工作，包括用户的实时行为、酒店实时价格、实时库存等等，以应对供给变化快的特点。



这里介绍一个业务特点导致的比较独特的问题：**模型切分**。美团酒店有很多业务场景，包括国内酒店、境外酒店、长租、钟点房等；还有两个 App，美团 App 和大众点评 App；还有搜索和筛选两种场景，搜索带查询词，筛选没有查询词，两种场景差异较大；从地理位置维度，还可以分成本地和异地两种场景。

面对这么多的业务场景，第一个问题就是模型怎么设计，是用统一的大模型，还是分成很多不同的小模型？我们可以用一个大模型 Cover 所有的场景，用特征来区分不同场景的差异，好处是统一模型维护和优化成本低。也可以划分很多小模型，这里有一个比较好的比喻，多个专科专家会诊，胜过一个全科医生。切分模型后，可以避免差异较大的业务之间互相影响，也方便对特殊场景进行专门的优化。

在模型切分上，主要考虑三个因素：

- 第一，业务之间的差异性。比如长租和境外差异很大，国内酒店和境外业务差异也很大，这种需要拆分。
- 第二，细分后的数据量。场景分的越细，数据量就越小，会导致两个问题，一是特征的覆盖率进一步降低；二是数据量变小后，不利于后续的模型迭代，一些复杂模型对数据量有很高的要求。我们做过尝试，国内酒店场景下，美团和大众点评两个 App 数据量都很大，而且用户也很不一样，所以做了模型拆分；但是境外酒店，因为本身是新业务数据量较小，就没有再进行细分。
- 第三，一切以线上指标为准。我们会做大量的实验，看当前数据量下怎么拆分效果更好，比如美团 App 的国内酒店，我们发现把搜索和筛选拆开后，效果

更好；筛选因为数据量特别大，拆分成本、异地效果也更好，但是如果搜索场景拆分成本地、异地模型就没有额外收益了。最终，一切都要以线上的实际表现为准。

模型演进

接下来介绍一下排序模型的演进过程，因为业务特点及历史原因，酒店搜索的排序模型走了一条不一样的演进路线。大家可以看业界其他公司点击率模型的演进，很多都是从 LR/FTRL 开始，然后进化到 FM/FFM，或者用 GBDT+LR 搞定特征组合，然后开始 Wide&Deep。

酒店搜索的演进就不太一样。酒店业务天生自带大量连续特征，如酒店价格、酒店和用户的距离、酒店评分等，因此初始阶段使用了对连续特征比较友好的树模型。在探索深度排序模型的时候，因为已经有了大量优化过的连续特征，导致我们的整个思路也不太一样，主要是借鉴一些模型的思想，结合业务特点做尝试，下面逐一进行介绍。

非线性

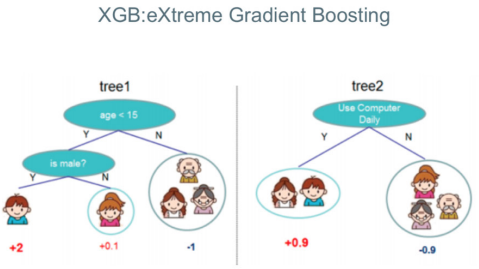
GBDT的改进，通过树节点的分裂实现非线性

特征组合

树的层次结构实现不同特征的自动组合

适合酒店业务特点

连续特征多，如酒店评分、价格、距离等



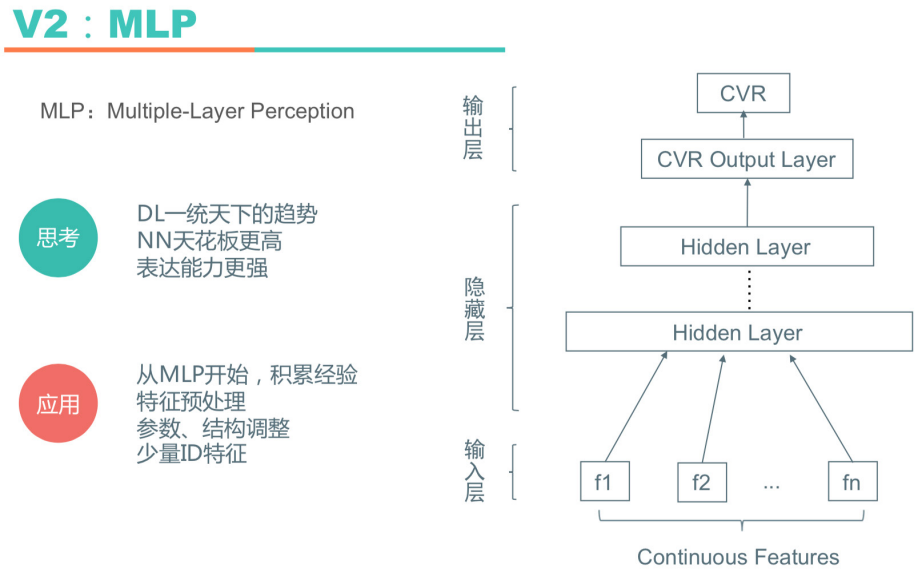
T Chen, C Guestrin. Xgboost: A scalable tree boosting system. KDD2016

初始阶段线上使用的模型是 XGB(XGBoost, eXtreme Gradient Boosting)。作为 GBDT 的改进，XGB 实现了非线性和自动的特征组合。树节点的分裂其实就实现了非线性，树的层次结构实现了不同特征的自动组合，而且树模型对特征的包容性非常好，树的分裂通过判断相对大小来实现，不需要对特征做特殊处理，适合连续特征。

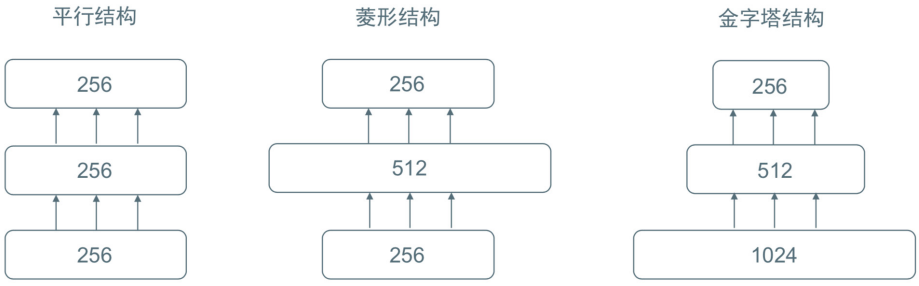
树模型的这些特点确实很适合酒店这种连续特征多的场景，至今为止，XGB 都是数据量较小场景下的主模型。但是树模型优化到后期遇到了瓶颈，比如特征工程收益变小、增大数据量没有额外收益等，此外树模型不适合做在线学习的问题愈发严重。酒店用户在劳动节、国庆节等节假日行为有较大不同，这时需要快速更新模型，我们尝试过只更新最后几棵树的做法，效果不佳。考虑到未来进一步的业务发展，有必要做模型升级。

模型探索的原则是从简单到复杂，逐步积累经验，所以首先尝试了结构比较简单的 MLP(Multiple-Layer Perception) 多层感知机，也就是全连接神经网络。神经网络是一种比树模型“天花板”更高的模型，“天花板”更高两层意思：第一层意思，可以优化提升的空间更大，比如可以进行在线学习，可以做多目标学习；第二层意思，模型的容量更大，“胃口”更大，可以“吃下”更多数据。此外它的表达能力也更强，可以拟合任何函数，网络结构和参数可以调整的空间也更大。但是它的优点同时也是它的缺点，因为它的网络结构、参数等可以调整的空间更大，神经网需要做很

多的参数和网络结构层面的调整。



上图是 MLP 的网络结构图，包含输入层、若干个隐藏层、输出层。在很长一段时间内，在特征相同的情况下，MLP 效果不如 XGB，所以有段时间线上使用的是 XGB 和 MLP 的融合模型。后来经过大量的网络结构调整和参数调整，调参经验越来越丰富，MLP 才逐步超越 XGB。这里额外说明一下，酒店搜索中有少量的 ID 类特征，在第一版 MLP 里 ID 类特征是直接当做连续特征处理的。比如城市 ID，ID 的序关系有一定的物理意义，大城市 ID 普遍较小，小城市开城晚一些，ID 较大。

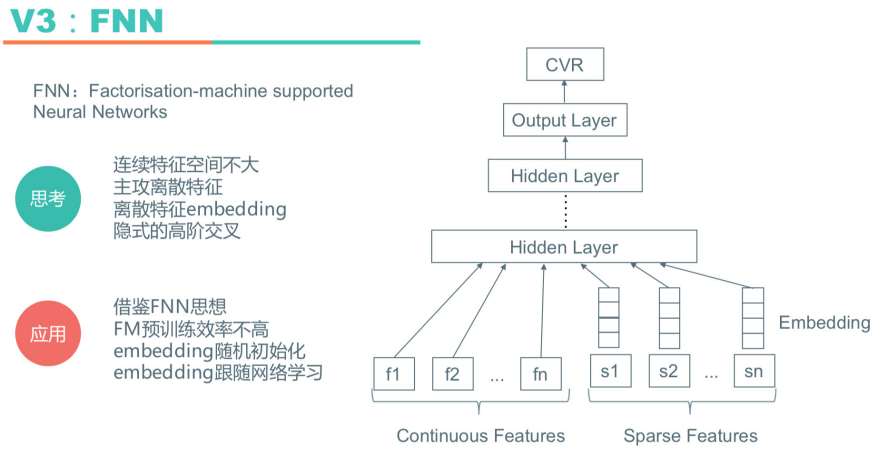


在 MLP 阶段我们对网络结构做了大量实验，尝试过三种网络结构：平行结构、

菱形结构、金字塔结构。在很多论文中提到三者相比平行结构效果最好，但是因为酒店搜索的数据不太一样，实验发现金字塔结构效果最好，即上图最右边的“1024-512-256”的网络结构。同时还实验了不同网络层数对效果的影响，实验发现 3-6 层的网络效果较好，更深的网络没有额外收益而且线上响应时间会变慢，后面各种模型探索都是基于 3 到 6 层的金字塔网络结构进行尝试。

MLP 上线之后，我们开始思考接下来的探索方向。在树模型阶段，酒店搜索组就在连续特征上做了很多探索，连续特征方面很难有比较大的提升空间；同时业界的研究重点也放在离散特征方面，所以离散特征应该是下一步的重点方向。

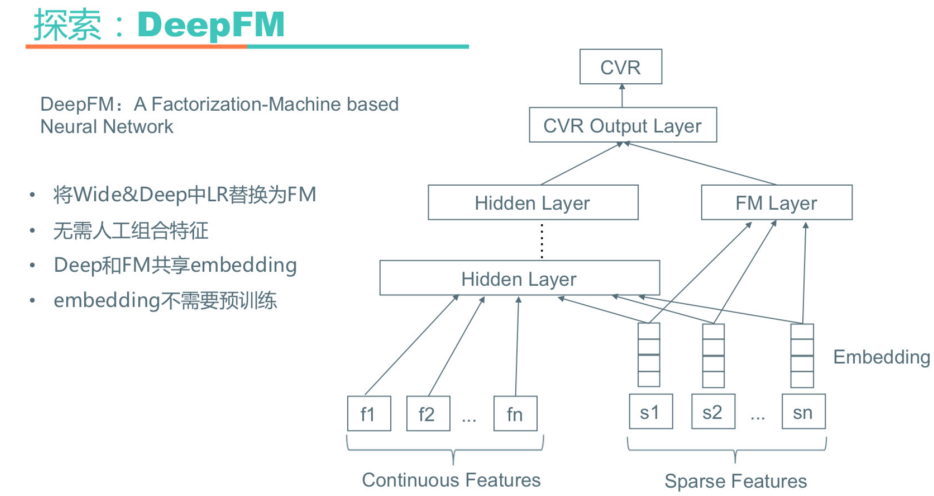
深度排序模型对离散特征的处理有两大类方法，一类是对离散特征做 Embedding，这样离散特征就可以表示成连续的向量放到神经网络中去，另一类是 Wide&Deep，把离散特征直接加到 Wide 侧。我们先尝试了第一种，即对离散特征做 Embedding 的方法，借鉴的是 FNN 的思想。其实离散特征做 Embedding 的想法很早就出现了，FM 就是把离散特征表示成 K 维向量，通过把高维离散特征表示成低维向量增加模型泛化能力。



Weinan Zhang et al. Deep Learning over Multi-Field Categorical Data: A Case Study on User Response Prediction. ECIR 2016

实际使用中，我们稍微做了一些改动，实验中发现使用 FM 预训练的效率不高，所以尝试了不做预训练直接把 Embedding 随机初始化，然后让 Embedding 跟随网络一起学习，实验结果发现比 FM 预训练效果还要好一点。最后的做法是没有用 FM

做预训练，让 Embedding 随机初始化并随网络学习，上图是线上的 V3 模型。



Huifeng Guo et al. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. IJCAI2017

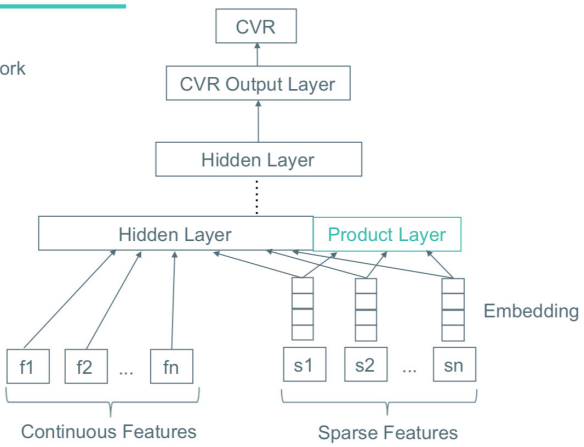
FNN 的成功上线证明离散特征 Embedding 这个方向值得深挖，所以我们接着实验了 DeepFM。DeepFM 相对于 Wide&Deep 的改进，非常类似于 FM 相对 LR 的改进，都认为 LR 部分的人工组合特征是个耗时耗力的事情，而 FM 模块可以通过向量内积的方式直接求出二阶组合特征。DeepFM 使用 FM 替换了 Wide&Deep 中的 LR，离散特征的 Embedding 同时“喂”给神经网和 FM，这部分 Embedding 是共享的，Embedding 在网络的优化过程中自动学习，不需要做预训练，同时 FM Layer 包含了一阶特征和二阶的组合特征，表达能力更强。我们尝试了 DeepFM，线下有提升线上波动提升，并没有达到上线的标准，最终没有全量。

尽管 DeepFM 没有成功上线，但这并没有动摇我们对 Embedding 的信心，接下来尝试了 PNN。PNN 的网络重点在 Product 上面，在点击率预估中，认为特征之间的关系更多是一种 And“且”的关系，而非 Add“加”的关系，例如性别为男且用华为手机的人，他定酒店时属于商务出行场景的概率更高。

探索：PNN

PNN：Product-based Neural Network

- 特征之间‘and’关系
- 显式的2阶特征组合
- 隐式的高阶特征组合
- 内积和外积两种方式



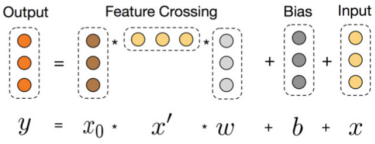
Yanru Qu et al. Product-based neural networks for user response prediction. ICDM2016

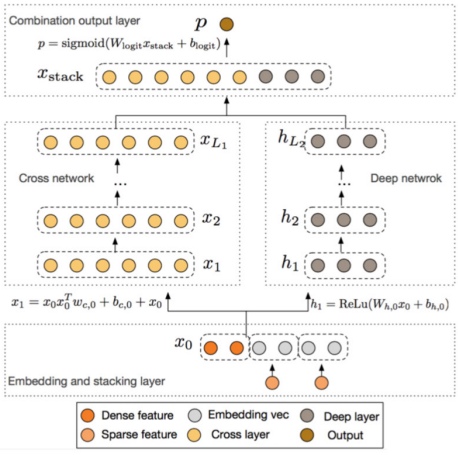
PNN 使用了 Product Layer 进行显式的二阶特征组合。上图右边是 PNN 的网络结构图，依然对离散特征做 Embedding，Embedding 向量同时送往隐层和 Product 层，Product 通过内积或者外积的方式，对特征做显式的二阶交叉，之后再送入神经网的隐层，这样可以做到显式的二阶组合和隐式的高阶特征组合。特征交叉基于乘法的运算实现，有两种方式：内积和外积。我们尝试了内积的方式，线下略有提升线上也是波动提升，没有达到上线标准，所以最终也没有全量上线。

探索：DCN

DCN：Deep & Cross network

- 引入Cross network
- 显式高阶特征交叉
- Cross网络和Deep网络并行
- 显示交叉&隐式交叉
- “元素”级交叉





Ruoxi Wang, et al. Deep & Cross Network for Ad Click Predictions. ADKDD2017

PNN 之后我们认为 Embedding 还可以再尝试一下，于是又尝试了 DCN（Deep&Cross Network）。DCN 引入了一个 Cross Network 进行显式的高阶特征交叉。上图右边是论文中的图，可以看到 Deep&Cross 中用了两种网络，Deep 网络和 Cross 网络，两种网络并行，输入都一样，在最后一层再 Stack 到一起。

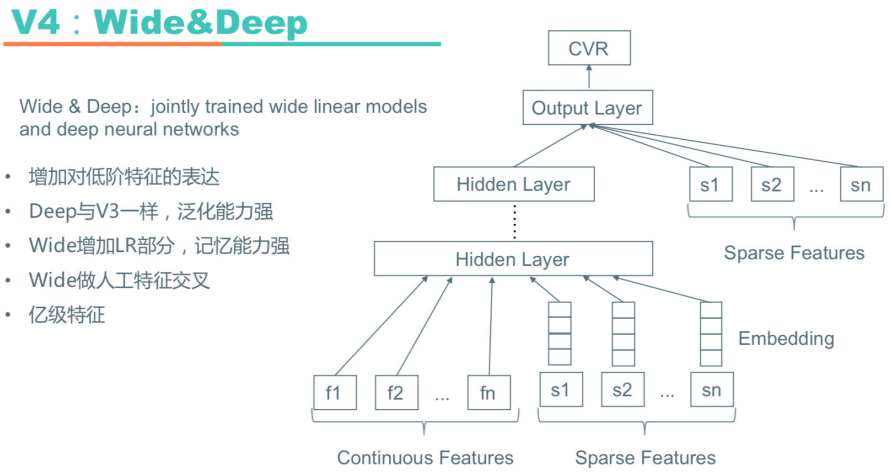
Deep 网络和前面几种网络一样，包括连续特征和离散特征的 Embedding，Cross 网络是 DCN 的特色，在 Cross 网络里面，通过巧妙的设计实现了特征之间的显式高阶交叉。看上图左下角的 Cross 结构示意，这里的 x 是每一层的输入，也就是上一层的输出。Feature Crossing 部分包括了原始输入 x0、本层输入 x 的转置、权重 w 三项，三项相乘其实就做了本层输入和原始输入的特征交叉，x1 就包含了二阶的交叉信息，x2 就包含了三阶的交叉信息，就可以通过控制 Cross 的层数显式控制交叉的阶数。

不得不说，DCN 在理论上很漂亮，我们也尝试了一下。但是很可惜，线下有提升线上波动提升，依然未能达到上线的标准，最终未能全量上线。

经过 DeepFM、PNN、DCN 的洗礼，促使我们开始反思，为什么在学术上特别有效的模型，反而在酒店搜索场景下不能全量上线呢？它们在线下都有提升，在线上也有提升，但是线上提升较小且有波动。

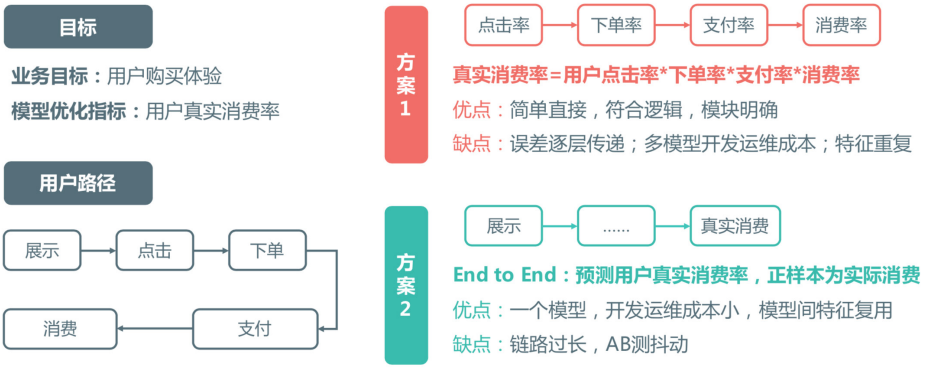
经过认真分析我们发现可能有两个原因：第一，连续特征的影响，XGB 时代尝试了 600 多种连续特征，实际线上使用的连续特征接近 400 种，这部分特征太强了；第二，离散特征太少，离散特征只有百万级别，但是 Embedding 特别适合离散特征多的情况。接下来方向就很明确了：补离散特征的课。

最终，我们还是把目光转回 Wide&Deep。Wide&Deep 同时训练一个 Wide 侧的线性模型和一个 Deep 侧的神经网络，Wide 部分提供了记忆能力，关注用户有过的历史行为，Deep 部分提供了泛化能力，关注一些没有历史行为的 Item。之前的工作主要集中在 Deep 测，对低阶特征的表达存在缺失，所以我们添加了 LR 模块以增加对低阶特征的表达，Deep 部分和之前的 V3 一样。刚开始只用了少量的 ID 类特征，效果一般，后来加了大量人工的交叉特征，特征维度达到了亿级别后效果才得到很好的提升。下图是我们的 V4 模型：



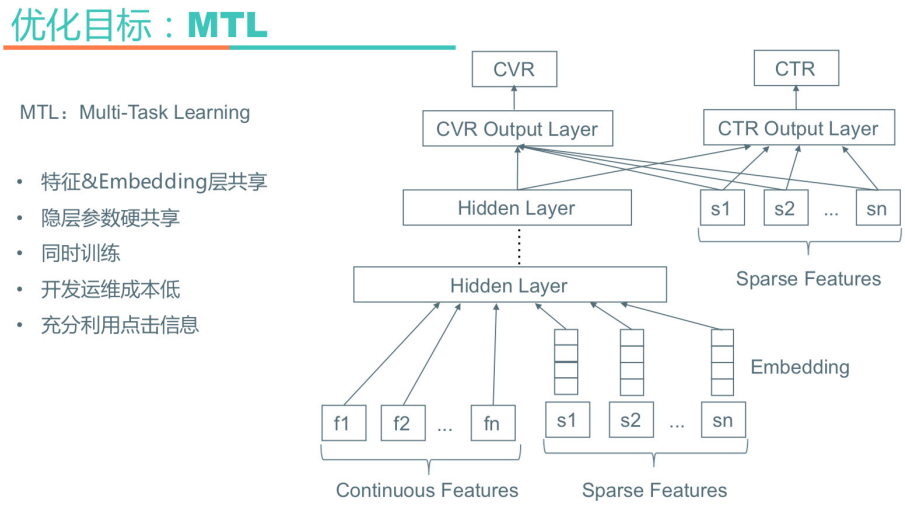
Heng-Tze Cheng et al. 2016. Wide & deep learning for recommender systems. 2016

接下来介绍一下优化目标的迭代过程（后面讲 MTL 会涉及这部分内容）。酒店搜索的业务目标是优化用户的购买体验，模型的优化指标是用户的真实消费率，怎么优化这个目标呢？通过分析用户的行为路径可以把用户的行为拆解成“展示 -> 点击 -> 下单 -> 支付 -> 消费”等 5 个环节，这其中每个环节都可能存在用户流失，比如有些用户支付完成后，因为部分商家确认比较慢，用户等不及就取消了。



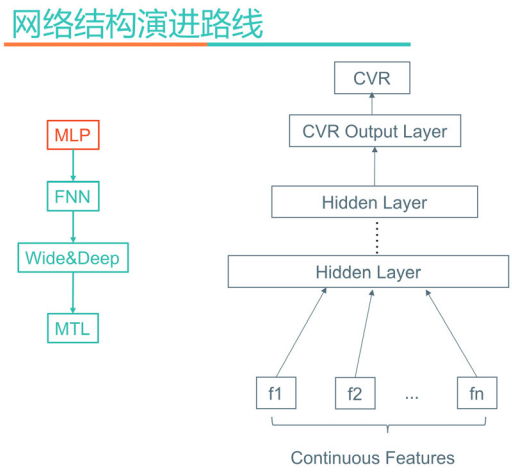
刚开始我们采用了方案 1，对每一个环节建模（真实消费率 = 用户点击率 × 下单率 × 支付率 × 消费率）。优点是非常简单直接且符合逻辑，每个模块分工明确，容易确认问题出在哪里。缺点也很明显，首先是特征重复，4 个模型在用户维度和商

家维度的特征全部一样，其次模型之间是相乘关系且层数过多，容易导致误差逐层传递，此外 4 个模型也增加了运维成本。后来慢慢进化到了方案 2 的“End to End”方式，直接预测用户的真实消费率，这时只需要把正样本设定为实际消费的样本，一个模型就够了，开发和运维成本较小，模型间特征也可以复用，缺点就是链路比较长，上线时经常遇到 AB 测抖动问题。



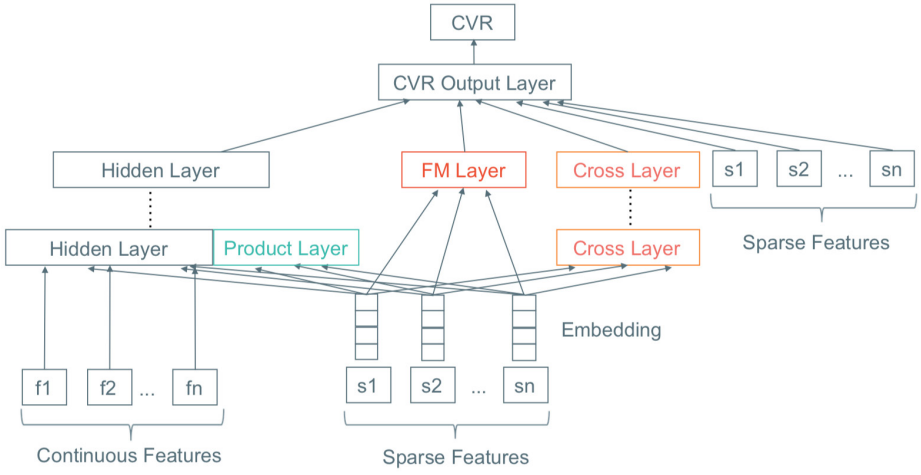
模型切换到神经网络后就可以做多任务学习了，之前树模型时代只预测“End to End”真实访购率，神经网络则可以通过多任务学习同时预测 CTR 展示点击率和 CVR 点击消费率。多任务学习通过硬共享的方式同时训练两个网络，特征、Embedding 层、隐层参数都是共享的，只在输出层区分不同的任务。上图是酒店搜索当前线上的模型，基于 Wide&Deep 做的多任务学习。

网络结构演进路线



上图是酒店搜索排序的深度排序模型演进路线，从 MLP 开始，通过对离散特征做 Embedding 进化到 FNN，中间尝试过 DeepFM、PNN、DCN 等模型，后来加入了 Wide 层进化到 Wide&Deep，现在的版本是一个 MTL 版的 Wide&Deep，每个模块都是累加上去的。

除了上面提到的模型，我们还探索过这个：



这是我们自己设计的混合网络，它融合了 FNN、DeepFM、PNN、DCN、Wide&Deep 等不同网络的优点，同时实现了一阶特征、显式二阶特征组合、显式高阶特征组合、隐式高阶特征组合等，有兴趣的同学可以尝试一下。

不同模型实验结果

不同模型实验结果

模型	离线AUC	线上效果（访购率）
XGBoost	Baseline	Baseline
MLP	+18BP	+0.34%
MLP+XGBoost	+32BP	+0.79%
FNN	+33BP	+0.73%
DeepFM	+39BP	波动提升，未全量
Deep&Cross	+24BP	波动提升，未全量
Wide&Deep	+29BP	无线上实验
Wide&Deep（组合特征）	+54BP	+1.26%

上图是不同模型的实验结果，这里的 BP 是基点（Basis Point），1BP=0.01%。XGB 是 Baseline，MLP 经过很长时间的调试才超过 XGB，MLP 和 XGB 融合模型的效果也很好，不过为了方便维护，最终还是用 FNN 替换了融合模型。Wide&Deep 在开始阶段，提升并没有特别多，后来加了组合特征后效果才好起来。我们 Embedding 上面的尝试，包括 DeepFM、Deep&Cross 等，线下都有提升，线上波动有提升，但是未能达到上线的标准，最终未能全量。

连续特征

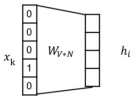
累积分布归一化： $x' = \int_{-\infty}^{x} f(x)dx$

标准化： $x' = \frac{x-\mu}{\sigma}$

根号、对号变换： $x' = \sqrt{x}$ $x' = log_2^x$

离散特征

ID-Embedding：



ID-Combine： $\emptyset_k(x) = \Pi x_i^{c_{ki}}, c_{ki} \in (0,1)$

缺失值

缺失值参数化： $x' = w_{miss} \cdot g(x) + w_{hit} \cdot f(x)$

$g(x) = \begin{cases} 0, x_{正常} \\ 1, x_{缺失} \end{cases}$ $f(x) = \begin{cases} x, x_{正常} \\ 0, x_{缺失} \end{cases}$

处理方法	AUC
标准化	Baseline
累积分布归一化	+15BP
标准化+累积分布归一化	+26BP
根号+对数等手工变化	+7BP
ID-COMBINE	+25BP
ID-Embedding	+23BP
缺失值处理	+27BP

在特征预处理方面对连续特征尝试了累计分布归一化、标准化，以及手工变换如根号变换、对数变换等；累积分布归一化其实就是做特征分桶，因为连续特征多且分布范围很广，累积分布归一化对酒店搜索的场景比较有效。

离散特征方面尝试了特征 Embedding 及离散特征交叉组合，分别对应 FNN 和 Wide&Deep。这里特别提一下缺失值参数化，因为酒店业务是一种低频业务，特征覆盖率低，大量样本存在特征缺失的情况，如果对缺失特征学一个权重，非缺失值学一个权重效果较好。

激活函数

ReLU $f(x) = \max(0, x)$

Sigmoid $f(x) = \frac{1}{1+e^{-x}}$

Leaky_ReLU $f(x) = \begin{cases} x & , x \geq 0 \\ 0.01x & , x < 0 \end{cases}$

ELU $f(x) = \begin{cases} x & , x > 0 \\ \alpha(e^x - 1) & , x \leq 0 \end{cases}$

优化器

Adagrad

RMSProp

Adam

$g_t = \nabla_{\theta_{t-1}} f(\theta_{t-1})$

$\Delta\theta_t = -\eta * g_t$

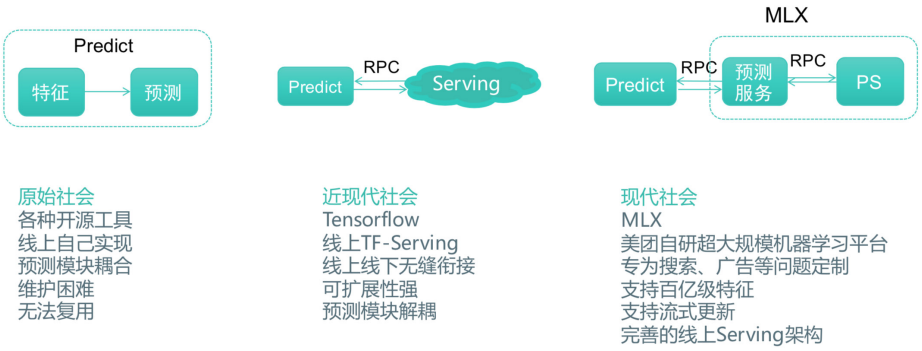
SGD

其他 BN/Dropout/学习率/隐层

激活函数	优化器	AUC
Sigmoid	Adagrad	-300BP
ReLU	Adagrad	Baseline
ELU	Adagrad	-2BP
LeakyReLU	Adagrad	+5BP
ReLU	RMSProp	+10BP
ReLU	Adam	+15BP

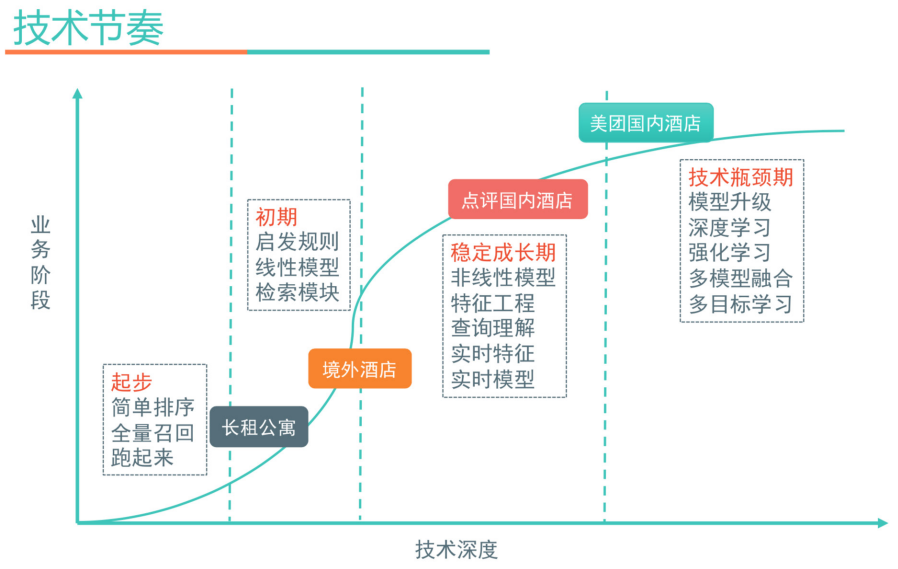
参数调优方面分别尝试了激活函数、优化器等。激活函数尝试过 Sigmoid、

ReLU、Leaky_ReLU、ELU 等；优化器也实验过 Adagrad、Rmsprop、Adam 等；从实验效果看，激活函数 ReLU+Adam 效果最好。刚开始时，加了 Batch Normalization 层和 Dropout 层，后来发现去掉后效果更好，可能和酒店搜索的数据量及数据特点有关。网络结构和隐层数方面用的是 3 到 6 层的金字塔网络。学习率方面的经验是学习率小点比较好，但是会导致训练变慢，需要找到一个平衡点。



下面介绍深度排序模型线上 Serving 架构的演化过程，初始阶段组内同学各自探索，用过各种开源工具如 Keras、TensorFlow 等，线上分别自己实现，预测代码和其他代码都放一起，维护困难且无法复用。

后来组内决定一起探索，大家统一使用 TensorFlow，线上用 TF-Serving，线上线下可以做到无缝衔接，预测代码和特征模块也解耦了。现在则全面转向 MLX 平台，MLX 是美团自研的超大规模机器学习平台，专为搜索、推荐、广告等排序问题定制，支持百亿级特征和流式更新，有完善的线上 Serving 架构，极大地解放了算法同学的生产力。



最后介绍一下我们对搜索排序技术节奏的一些理解，简单来说就是在不同阶段做不同的事情。

在上图中，横轴表示技术深度，越往右技术难度越大，人力投入越大，对人的要求也越高。纵轴是业务阶段。业务阶段对技术的影响包括两方面，数据量和业务价值。数据量的大小，可以决定该做什么事情，因为有些技术在数据量小的时候意义不大；业务价值就更不用说了，业务价值越大越值得“重兵投入”。

- 起步阶段：起步阶段，还没有数据，这时候做简单排序就好，比如纯按价格排序或者距离排序，目的是让整个流程快速地跑起来，能提供最基本的服务。比如 2017 年，美团的长租业务当时就处于起步阶段。
- 业务初期：随着业务的发展，就进入了业务发展初期，订单数慢慢增长，也有了一些数据，这时候可以增加一些启发式规则或者简单的线性模型，检索模型也可以加上。但是由于数据量还比较小，没必要部署很复杂的模型。
- 稳定成长期：业务进一步发展后，就进入了稳定成长期，这时候订单量已经很大了，数据量也非常大了，这段时间是“补课”的时候，可以把意图理解的模块加上，排序模型也会进化到非线性模型比如 XGB，会做大量的特征工程，

实时特征以及实时模型，在这个阶段特征工程收益巨大。

- 技术瓶颈期：这个阶段的特点是基本的东西都已经做完了，在原有的技术框架下效果提升变的困难。这时需要做升级，比如将传统语义模型升级成深度语义模型，开始尝试深度排序模型，并且开始探索强化学习、多模型融合、多目标学习等。

中国有句俗话叫“杀鸡焉用牛刀”，比喻办小事情，何必花费大力气，也就是不要小题大做。其实做技术也一样，不同业务阶段不同数据量适合用不同的技术方案，没有必要过度追求先进的技术和高大上的模型，根据业务特点和业务阶段选择最匹配的技术方案才是最好的。我们认为，**没有最好的模型，只有合适的场景**。

总结

酒店搜索作为 O2O 搜索的一种，和传统的搜索排序相比有很多不同之处，既要解决搜索的相关性问题，又要提供“千人千面”的排序结果，优化用户购买体验，还要满足业务需求。通过合理的模块划分可以把这三大类问题解耦，检索、排序、业务三个技术模块各司其职。在检索和意图理解层面，我们做了地标策略、NER 模型和多级检索架构来保证查询结果的相关性；排序模型上结合酒店搜索的业务特点，借鉴业界先进思想，尝试了多种不同的深度排序模型，走出了一条不一样的模型演进路线。同时通过控制技术节奏，整体把握不同业务的技术选型和迭代节奏，对不同阶段的业务匹配不同的技术方案，只选对的，不选贵的。

参考文献

[1] John Lafferty et al. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.ICML2001.
[2] Guillaume Lample et al Neural architectures for named entity recognition. NAACL2016.
[3] Zhiheng Huang, Wei Xu, and Kai Yu. 2015.
[4] Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint arXiv:1508.01991.
[5] Xuezhe Ma et al.End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF.ACL2016.

[6] T Chen, C Guestrin. XGBoost: A scalable tree boosting system. KDD2016.
[7] Weinan Zhang et al. Deep Learning over Multi-Field Categorical Data: A Case Study on User Response Prediction. ECIR 2016.
[8] Huifeng Guo et al. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. IJCAI2017.
[9] Yanru Qu et al. Product-based neural networks for user response prediction. ICDM2016.
[10] Heng-Tze Cheng et al. 2016. Wide & deep learning for recommender systems. 2016.In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems.
[11] Ruoxi Wang et al. Deep & Cross Network for Ad Click Predictions. ADKDD2017.

作者简介

艺涛，美团高级技术专家，2016 年加入美团，现负责美团酒店业务搜索排序技术。2010 年毕业于中科院计算所，曾在网易有道等公司工作，先后从事网页搜索、购物搜索、计算广告等方向的研发工作。曾荣获“Kaggle 卫星图像分类大赛”亚军，QCon 明星讲师。

大众点评搜索基于知识图谱的深度学习排序实践

非易　祝升　汤彪　张弓　仲远

1. 引言

挑战与思路

搜索是大众点评 App 上用户进行信息查找的最大入口，是连接用户和信息的重要纽带。而用户搜索的方式和场景非常多样，并且由于对接业务种类多，流量差异大，为大众点评搜索（下文简称点评搜索）带来了巨大的挑战，具体体现在如下几个方面：

1. **意图多样**：用户查找的信息类型和方式多样。信息类型包括 POI、榜单、UGC、攻略、达人等。以找店为例，查找方式包括按距离、按热度、按菜品和按地理位置等多种方式。例如用户按照品牌进行搜索时，大概率是需要寻找距离最近或者常去的某家分店；但用户搜索菜品时，会对菜品推荐人数更加敏感，而距离因素会弱化。
2. **业务多样**：不同业务之间，用户的使用频率、选择难度以及业务诉求均不一样。例如家装场景用户使用频次很低，行为非常稀疏，距离因素弱，并且选择周期可能会很长；而美食多为即时消费场景，用户行为数据多，距离敏感。
3. **用户类型多样**：不同的用户对价格、距离、口味以及偏好的类目之间差异很大；搜索需要能深度挖掘到用户的各种偏好，实现定制化的“千人千面”的搜索。
4. **LBS 的搜索**：相比电商和通用搜索，LBS 的升维效应极大地增加了搜索场景的复杂性。例如对于旅游用户和常驻地用户来说，前者在搜索美食的时候可能会更加关心当地的知名特色商户，而对于距离相对不敏感。

上述的各项特性，叠加上时间、空间、场景等维度，使得点评搜索面临比通

用搜索引擎更加独特的挑战。而解决这些挑战的方法，就需要升级 NLP（Natural Language Processing，自然语言处理）技术，进行深度查询理解以及深度评价分析，并依赖知识图谱技术和深度学习技术对搜索架构进行整体升级。在美团 NLP 中心以及大众点评搜索智能中心两个团队的紧密合作之下，经过短短半年时间，点评搜索核心 KPI 在高位基础上仍然大幅提升，是过去一年半涨幅的六倍之多，提前半年完成全年目标。

基于知识图谱的搜索架构重塑

美团 NLP 中心正在构建全世界最大的餐饮娱乐知识图谱——美团大脑（相关信息请参见《美团大脑：知识图谱的建模方法及其应用》）。它充分挖掘关联各个场景数据，用 NLP 技术让机器“阅读”用户公开评论，理解用户在菜品、价格、服务、环境等方面的喜好，构建人、店、商品、场景之间的知识关联，从而形成一个“知识大脑”[1]。通过将知识图谱信息加入到搜索各个流程中，我们对点评搜索的整体架构进行了升级重塑，图 1 为点评搜索基于知识图谱搭建的 5 层搜索架构。本篇文章是“美团大脑”系列文章第二篇（系列首篇文章请参见《美团餐饮娱乐知识图谱——美团大脑揭秘》），主要介绍点评搜索 5 层架构中核心排序层的演变过程，文章主要分为如下 3 个部分：

1. 核心排序从传统机器学习模型到大规模深度学习模型的演进。
2. 搜索场景深度学习排序模型的特征工程实践。
3. 适用于搜索场景的深度学习 Listwise 排序算法——LambdaDNN。

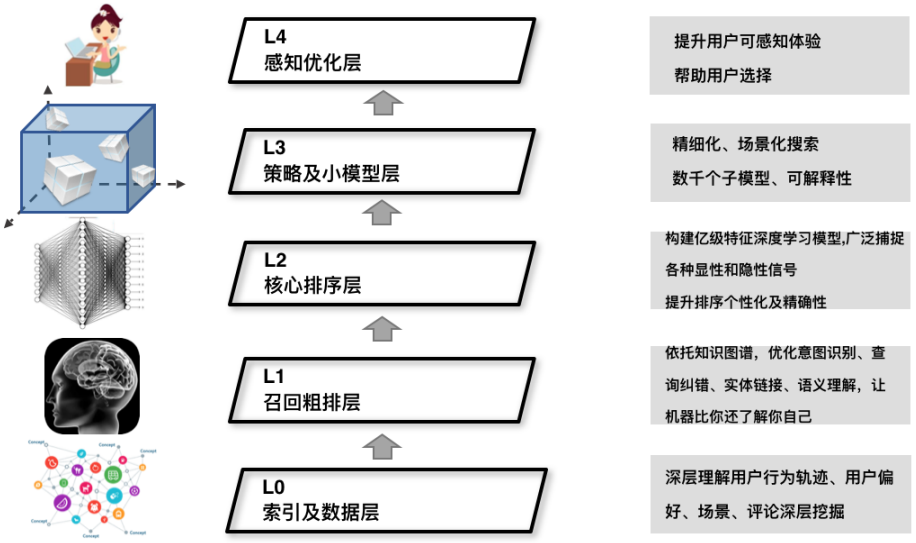


图 1　基于知识图谱的点评搜索 5 层架构

2. 排序模型探索与实践

搜索排序问题在机器学习领域有一个单独的分支，Learning to Rank（L2R）。主要分类如下：

1. 根据样本生成方法和 Loss Function 的不同，L2R 可以分为 Pointwise、Pairwise、Listwise。
2. 按照模型结构划分，可以分为线性排序模型、树模型、深度学习模型，他们之间的组合（GBDT+LR，Deep&Wide 等）。

在排序模型方面，点评搜索也经历了业界比较普遍的迭代过程：从早期的线性模型 LR，到引入自动二阶交叉特征的 FM 和 FFM，到非线性树模型 GBDT 和 GBDT+LR，到最近全面迁移至大规模深度学习排序模型。下面先简单介绍下传统机器学习模型（LR、FM、GBDT）的应用和优缺点，然后详细介绍深度模型的探索实践过程。

传统机器学习模型

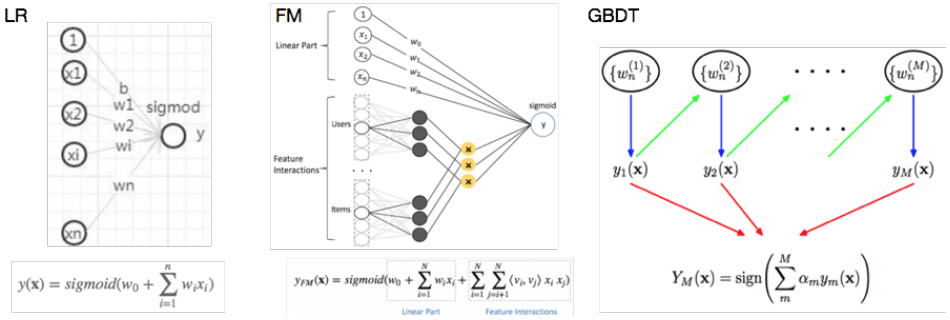


图 2　几种传统机器学习模型结构

1. LR 可以视作单层单节点的线性网络结构。模型优点是可解释性强。通常而言，良好的解释性是工业界应用实践比较注重的一个指标，它意味着更好的可控性，同时也能指导工程师去分析问题优化模型。但是 LR 需要依赖大量的人工特征挖掘投入，有限的特征组合自然无法提供较强的表达能力。
2. FM 可以看做是在 LR 的基础上增加了一部分二阶交叉项。引入自动的交叉特征有助于减少人工挖掘的投入，同时增加模型的非线性，捕捉更多信息。FM 能够自动学习两两特征间的关系，但更高量级的特征交叉仍然无法满足。
3. GBDT 是一个 Boosting 的模型，通过组合多个弱模型逐步拟合残差得到一个强模型。树模型具有天然的优势，能够很好的挖掘组合高阶统计特征，兼具较优的可解释性。GBDT 的主要缺陷是依赖连续型的统计特征，对于高维度稀疏特征、时间序列特征不能很好的处理。

深度神经网络模型

随着业务的发展，在传统模型上取得指标收益变得愈发困难。同时业务的复杂性要求我们引入海量用户历史数据，超大规模知识图谱特征等多维度信息源，以实现精准个性化的排序。因此我们从 2018 年下半年开始，全力推进 L2 核心排序层的主模型迁移至深度学习排序模型。深度模型优势体现在如下几个方面：

1. **强大的模型拟合能力**：深度学习网络包含多个隐藏层和隐藏结点，配合上非线

性的激活函数，理论上可以拟合任何函数，因此十分适用于点评搜索这种复杂的场景。

2. **强大的特征表征和泛化能力**：深度学习模型可以处理很多传统模型无法处理的特征。例如深度网络可以直接中从海量训练样本中学习到高维稀疏 ID 的隐含信息，并通过 Embedding 的方式去表征；另外对于文本、序列特征以及图像特征，深度网络均有对应的结构或者单元去处理。
3. **自动组合和发现特征的能力**：华为提出的 DeepFM，以及 Google 提出的 DeepCrossNetwork 可以自动进行特征组合，代替大量人工组合特征的工作。

下图是我们基于 Google 提出的 Wide&Deep 模型搭建的网络结构 [2]。其中 Wide 部分输入的是 LR、GBDT 阶段常用的一些细粒度统计特征。通过较长周期统计的高频行为特征，能够提供很好的记忆能力。Deep 部分通过深层的神经网络学习 Low-Order、高纬度稀疏的 Categorical 型特征，拟合样本中的长尾部分，发现新的特征组合，提高模型的泛化能力。同时对于文本、头图等传统机器学习模型难以刻画的特征，我们可以通过 End-to-End 的方式，利用相应的子网络模型进行预处理表示，然后进行融合学习。

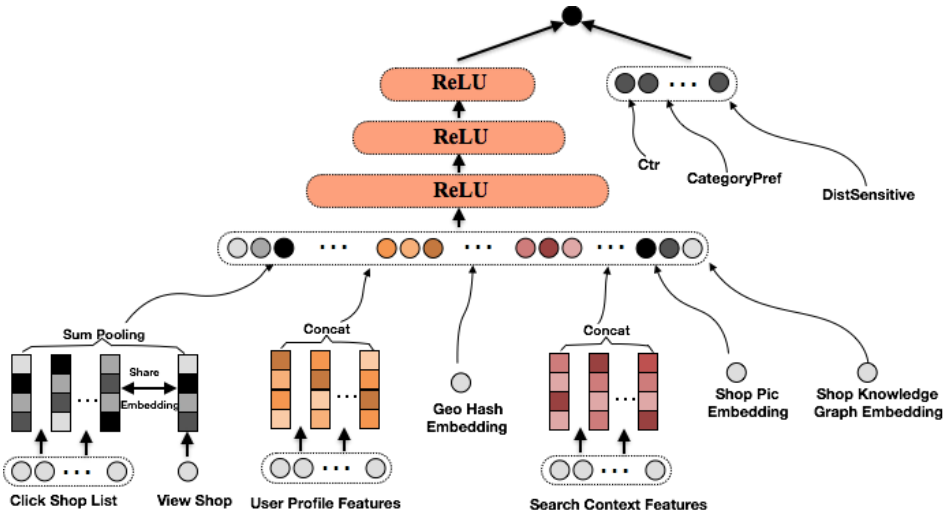


图 3　Deep&Wide 模型结构图

3. 搜索深度排序模型的特征工程实践

深度学习的横空出世，将算法工程师从很多人工挖掘和组合特征的事情中解放出来。甚至有一种论调，专做特征工程的算法工程师可能面临着失业的风险。但是深度学习的自动特征学习目前主要集中体现在 CV 领域，CV 领域的特征数据是图片的像素点——稠密的低阶特征，深度学习通过卷积层这个强力工具，可以自动对低阶特征进行组合和变换，相比之前人工定义的图像特征从效果上来说确实更加显著。在 NLP 领域因为 Transformer 的出现，在自动特征挖掘上也有了长足的进步，BERT 利用 Transformer 在多个 NLP Task 中取得了 State-of-The-Art 的效果。

但是对于 CTR 预估和排序学习的领域，目前深度学习尚未在自动特征挖掘上对人工特征工程形成碾压之势，因此人工特征工程依然很重要。当然，深度学习在特征工程上与传统模型的特征工程也存在着一些区别，我们的工作主要集中在如下几个方面。

3.1 特征预处理

- **特征归一化**：深度网络的学习几乎都是基于反向传播，而此类梯度优化的方法对于特征的尺度非常敏感。因此，需要对特征进行归一化或者标准化以促使模型更好的收敛。
- **特征离散化**：工业界一般很少直接使用连续值作为特征，而是将特征离散化后再输入到模型中。一方面因为离散化特征对于异常值具有更好的鲁棒性，其次可以为特征引入非线性的能力。并且，离散化可以更好的进行 Embedding，我们主要使用如下两种离散化方法：
  - 等频分桶：按样本频率进行等频切分，缺失值可以选择给一个默认桶值或者单独设置分桶。
  - 树模型分桶：等频离散化的方式在特征分布特别不均匀的时候效果往往不好。此时可以利用单特征结合 Label 训练树模型，以树的分叉点做为切分值，相应的叶子节点作为桶号。
- **特征组合**：基于业务场景对基础特征进行组合，形成更丰富的行为表征，为模型提供先验信息，可加速模型的收敛速度。典型示例如下：

- 用户性别与类目之间的交叉特征，能够刻画出不同性别的用户在类目上的偏好差异，比如男性用户可能会较少关注“丽人”相关的商户。
- 时间与类目之间的交叉特征，能够刻画出不同类目商户在时间上的差异，例如，酒吧在夜间会更容易被点击。

3.2 万物皆可 Embedding

深度学习最大的魅力在于其强大的特征表征能力，在点评搜索场景下，我们有海量的用户行为数据，有丰富的商户 UGC 信息以及美团大脑提供的多维度细粒度标签数据。我们利用深度学习将这些信息 Embedding 到多个向量空间中，通过 Embedding 去表征用户的个性化偏好和商户的精准画像。同时向量化的 Embedding 也便于深度模型进一步的泛化、组合以及进行相似度的计算。

3.2.1 用户行为序列的 Embedding

用户行为序列（搜索词序列、点击商户序列、筛选行为序列）包含了用户丰富的偏好信息。例如用户筛选了“距离优先”时，我们能够知道当前用户很有可能是一个即时消费的场景，并且对距离较为敏感。行为序列特征一般有如下图所示的三种接入方式：

- **Pooling**：序列 Embedding 后接入 Sum/Average Pooling 层。此方式接入成本低，但忽略了行为的时序关系。
- **RNN**：LSTM/GRU 接入，利用循环网络进行聚合。此方式能够考虑行为序列的时序关系；代价是增大了模型复杂度，影响线上预测性能。
- **Attention**：序列 Embedding 后引入 Attention 机制，表现为加权的 Sum Pooling；相比 LSTM/GRU 计算开销更低 [4]。

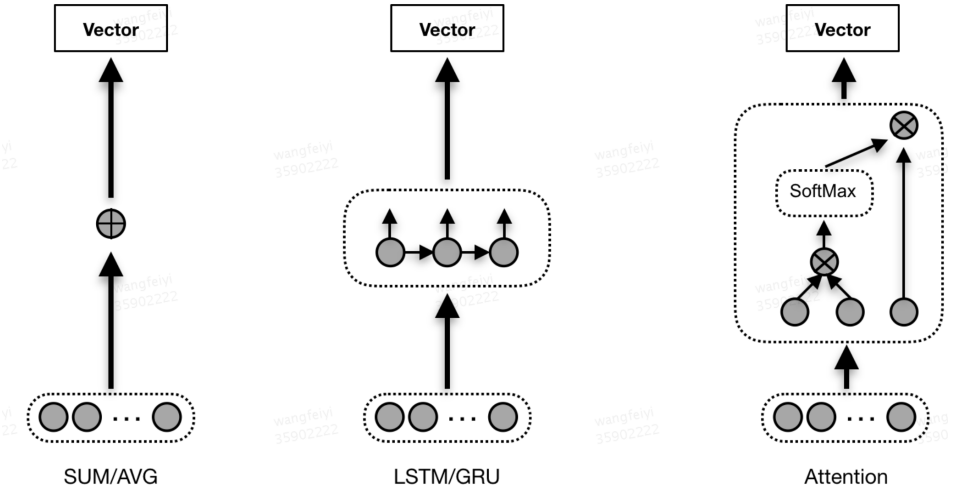


图 4　行为序列特征接入的几种方法

同时，为了突显用户长期偏好和短期偏好对于排序的不同影响，我们按照时间维度对行为序列进行了划分：Session、半小时、一天、一周等粒度，也在线上取得了收益。

3.2.2　用户 ID 的 Embedding

一种更常见的刻画用户偏好的方式，是直接将用户 ID 经过 Embedding 后作为特征接入到模型中，但是最后上线的效果却不尽如人意。通过分析用户的行为数据，我们发现相当一部分用户 ID 的行为数据较为稀疏，导致用户 ID 的 Embedding 没有充分收敛，未能充分刻画用户的偏好信息。

Airbnb 发表在 KDD 2018 上的文章为这种问题提供了一种解决思路 [9]——利用用户基础画像和行为数据对用户 ID 进行聚类。Airbnb 的主要场景是为旅游用户提供民宿短租服务，一般用户一年旅游的次数在 1–2 次之间，因此 Airbnb 的用户行为数据相比点评搜索会更为稀疏一些。

Buckets	1	2	3	4	5	6	7	8
Market	SF	NYC	LA	HK	PHL	AUS	LV	...
Language	en	es	fr	jp	ru	ko	de	...
Device Type	Mac	Msft	Andr	Ipad	Tablet	Iphone	...	
Full Profile	Yes	No						
Profile Photo	Yes	No						
Num Bookings	0	1	2-7	8+				
$ per Night	<40	40-55	56-69	70-83	84-100	101-129	130-189	190+
$ per Guest	<21	21-27	28-34	35-42	43-52	53-75	76+	
Capacity	<2	2-2.6	2.7-3	3.1-4	4.1-6	6.1+		
Num Reviews	<1	1-3.5	3.6-10	> 10				
Listing 5 Star %	0-40	41-60	61-90	90+				
Guest 5 Star %	0-40	41-60	61-90	90+				

图 5　按照用户画像和行为信息聚类

如上图所示，将用户画像特征和行为特征进行离散分桶，拼接特征名和所属桶号，得到的聚类 ID 为：US_lt1_pn3_pg3_r3_5s4_c2_b1_bd2_bt2_nu3。

我们也采取了类似 Airbnb 的方案，稀疏性的问题得到了很好的解决，并且这样做还获得了一些额外的收益。大众点评作为一个本地化的生活信息服务平台，大部分用户的行为都集中自己的常驻地，导致用户到达一个新地方时，排序个性化明显不足。通过这种聚类的方式，将异地有相同行为的用户聚集在一起，也能解决一部分跨站的个性化问题。

3.2.3　商户信息 Embedding

商户 Embedding 除了可以直接将商户 ID 加入模型中之外，美团大脑也利用深度学习技术对 UGC 进行大量挖掘，对商家的口味、特色等细粒度情感进行充分刻画，例如下图所示的“好停车”、“菜品精致”、“愿意再次光顾”等标签。

图 6　美团大脑提供的商家细粒度情感标签

这些信息与单纯的商户星级、点评数相比，刻画的角度更多，粒度也更细。我们将这些标签也进行 Embedding 并输入到模型中：

- **直连**：将标签特征做 Pooling 后直接输入模型。这种接入方式适合端到端的学习方式；但受输入层大小限制，只能取 Top 的标签，容易损失抽象实体信息。
- **分组直连**：类似于直连接入的方式，但是先对标签进行分类，如菜品 / 风格 / 口味等类别；每个分类取 Top N 的实体后进行 Pooling 生成不同维度的语义向量。与不分组的直连相比，能够保留更多抽象信息。

- **子模型接入**：可以利用 DSSM 模型，以标签作为商户输入学习商户的 Embedding 表达。此种方式能够最大化保留标签的抽象信息，但是线上实现和计算成本较高。

3.2.4 加速 Embedding 特征的收敛

在我们的深度学习排序模型中，除了 Embedding 特征，也存在大量 Query、Shop 和用户维度的强记忆特征，能够很快收敛。而 Embedding 特征是更为稀疏的弱特征，收敛速度较慢，为了加速 Embedding 特征的收敛，我们尝试了如下几种方案：

- **低频过滤**：针对出现频率较低的特征进行过滤，可以很大程度上减少参数量，避免过拟合。
- **预训练**：利用多类模型对稀疏 Embedding 特征进行预训练，然后进入模型进行微调：
  - 通过无监督模型如 Word2vec、Fasttext 对用户 – 商户点击关系建模，生成共现关系下的商户 Embedding。
  - 利用 DSSM 等监督模型对 Query– 商户点击行为建模得到 Query 和商户的 Embedding。
- **Multi-Task**：针对稀疏的 Embedding 特征，单独设置一个子损失函数，如下图所示。此时 Embedding 特征的更新依赖两个损失函数的梯度，而子损失函数脱离了对强特征的依赖，可以加快 Embedding 特征的收敛。

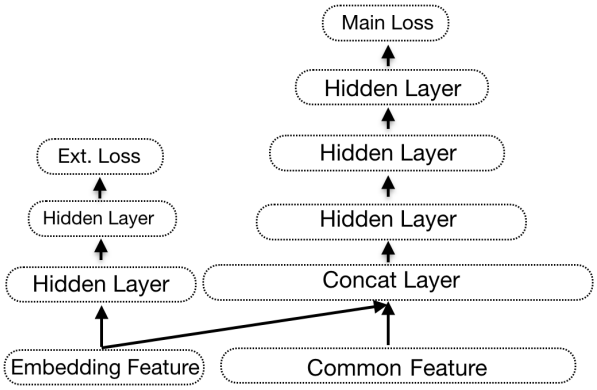


图 7　Multi-Task 加速 Embedding 特征收敛

3.3　图片特征

图片在搜索结果页中占据了很大的展示面积，图片质量的好坏会直接影响用户的体验和点击，而点评商户首图来自于商户和用户上传的图片，质量参差不齐。因此，图片特征也是排序模型中较为重要的一类。目前点评搜索主要用了以下几类图片特征：

- **基础特征**：提取图片的亮度、色度饱和度等基础信息，进行特征离散化后得到图片基础特征。
- **泛化特征**：使用 ResNet50 进行图片特征提取 [3]，通过聚类得到图片的泛化特征。
- **质量特征**：使用自研的图片质量模型，提取中间层输出，作为图片质量的 Embedding 特征。
- **标签特征**：提取图片是否是食物、环境、价目表、Logo 等作为图片分类和标签特征。

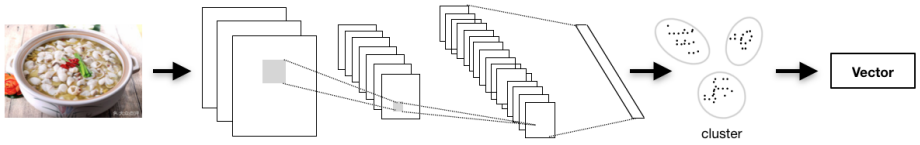


图 8　图片特征接入

4. 适用于搜索场景的深度学习 Listwise 排序算法：LambdaDNN

4.1 搜索业务指标与模型优化目标的 Gap

通常模型的预测目标与业务指标总会存在一些 Gap。如果模型的预测目标越贴近业务目标，越能保证模型优化的同时业务指标也能够有相应的提升；反之则会出现模型离线指标提升，但线上关键业务指标提升不明显，甚至出现负向的问题。工业届大部分深度学习排序采用 Pointwise 的 Log Loss 作为损失函数，与搜索业务指标有较大的 Gap。体现在如下两个方面：

1. 搜索业务常用的指标有 QV_CTR 或者 SSR(Session Success Rate)，更关心的是用户搜索的成功率（有没有发生点击行为）；而 Pointwise 的 Log Loss 更多是关注单个 Item 的点击率。
2. 搜索业务更关心排在页面头部结果的好坏，而 Pointwise 的方法则对于所有位置的样本一视同仁。

图 9　Pointwise 和 Listwise 优化目标的区别

基于上述理由，我们对于深度学习模型的损失函数进行了优化。

4.2 优化目标改进：从 Log Loss 到 NDCG

为了让排序模型的优化目标尽量贴近搜索业务指标，需要按照 Query 计算损失，且不同位置的样本具有不同的权重。搜索系统常用的指标 NDCG(Normalized Discounted Cumulative Gain) 相较于 Log Loss 显然更贴近搜索业务的要求，NDCG 计算公式如下：

$$NDCG@k(l)=\frac{1}{Z_k}\sum_{j=1}^{k}G(l_j)\eta(j)$$

累加部分为 DCG(Discounted Cumulative Gain) 表示按照位置折损的收益，对于 Query 下的结果列表 l，函数 G 表示对应 Doc 的相关度分值，通常取指数函数，即 G(lj)=2lj−1（lj 表示的是相关度水平，如 {0，1，2}）；函数 η 即位置折损，一般采用 η(j)=1/log(j+1)，Doc 与 Query 的相关度越高且位置越靠前则 DCG 值会越大。另外，通常我们仅关注排序列表页前 k 位的效果，Zk 表示 DCG@k 的可能最大值，以此进行归一化处理后得到的就是 NDCG@k。

问题在于 NDCG 是一个处处非平滑的函数，直接以它为目标函数进行优化是不可行的。LambdaRank 提供了一种思路：绕过目标函数本身，直接构造一个特殊的梯度，按照梯度的方向修正模型参数，最终能达到拟合 NDCG 的方法 [6]。因此，如果我们能将该梯度通过深度网络进行反向传播，则能训练一个优化 NDCG 的深度网络，该梯度我们称之为 Lambda 梯度，通过该梯度构造出的深度学习网络称之为 LambdaDNN。

要了解 Lambda 梯度需要引入 LambdaRank。LambdaRank 模型是通过 Pairwise 来构造的，通常将同 Query 下有点击样本和无点击样本构造成一个样本 Pair。模型的基本假设如下式所示，令 Pij 为同一个 Query 下 Doci 相比 Docj 更相关的概率，其中 si 和 sj 分别为 Doci 和 Docj 的模型得分：

$$P_{ij} \equiv P(U_i \rhd U_j) \equiv \frac{1}{1+e^{-\sigma(s_i-s_j)}}$$

使用交叉熵为损失函数，令 Sij 表示样本 Pair 的真实标记，当 Doci 比 Docj 更相关时（即 Doci 有被用户点击，而 Docj 没有被点击），有 Sij=1，否则为 −1；则损失函数可以表示为：

$$C = \frac{1}{2}(1-S_{ij})\sigma(s_i-s_j) + \log(1+e^{-\sigma(s_i-s_j)})$$

在构造样本 Pair 时，我们可以始终令 i 为更相关的文档，此时始终有 Sij ≡ 1，代入上式并进行求导，则损失函数的梯度为：

$$\lambda_{ij} = \frac{\partial C(s_i-s_j)}{\partial s_i} = \frac{-\sigma}{1+e^{\sigma(s_i-s_j)}}$$

到目前为止，损失函数的计算过程中并未考虑样本所在的位置信息。因此进一步对梯度进行改造，考虑 Doci 和 Docj 交换位置时的 NDCG 值变化，下式即为前述的 Lambda 梯度。可以证明，通过此种方式构造出来的梯度经过迭代更新，最终可以达到优化 NDCG 的目的。

$$\lambda_{ij} = \frac{\partial C(s_i-s_j)}{\partial s_i} = \frac{-\sigma}{1+e^{\sigma(s_i-s_j)}}|\Delta_{NDCG}|$$

Lambda 梯度的物理意义如下图所示。其中蓝色表示更相关（用户点击过）的文档，则 Lambda 梯度更倾向于位置靠上的 Doc 得到的提升更大（如红色箭头所示）。有了 Lambda 梯度的计算方法，训练中我们利用深度网络预测同 Query 下的 Doc 得分，根据用户实际点击 Doc 的情况计算 Lambda 梯度并反向传播回深度网络，则可以得到一个直接预测 NDCG 的深度网络。

图 10　Lambda 梯度的物理意义

4.3　LambdaDNN 的工程实施

我们利用 TensorFlow 分布式框架训练 LambdaDNN 模型。如前文所述，Lambda 梯度需要对同 Query 下的样本进行计算，但是正常情况下所有的样本是随机 Shuffle 到各个 Worker 的。因此我们需要对样本进行预处理：

1. 通过 QueryId 进行 Shuffle，将同一个 Query 的样本聚合在一起，同一个 Query 的样本打包进一个 TFRecord。
2. 由于每次请求 Query 召回的 Doc 数不一样，对于可变 Size 的 Query 样本在拉取数据进行训练时需要注意，TF 会自动补齐 Mini-Batch 内每个样本大小一致，导致输入数据中存在大量无意义的默认值样本。这里我们提供两点处理方式：
   - MR 过程中对 Key 进行处理，使得多个 Query 的样本聚合在一起，然后在训练的时候进行动态切分。
   - 读取到补齐的样本，根据设定的补齐标记获取索引位，去除补齐数据。

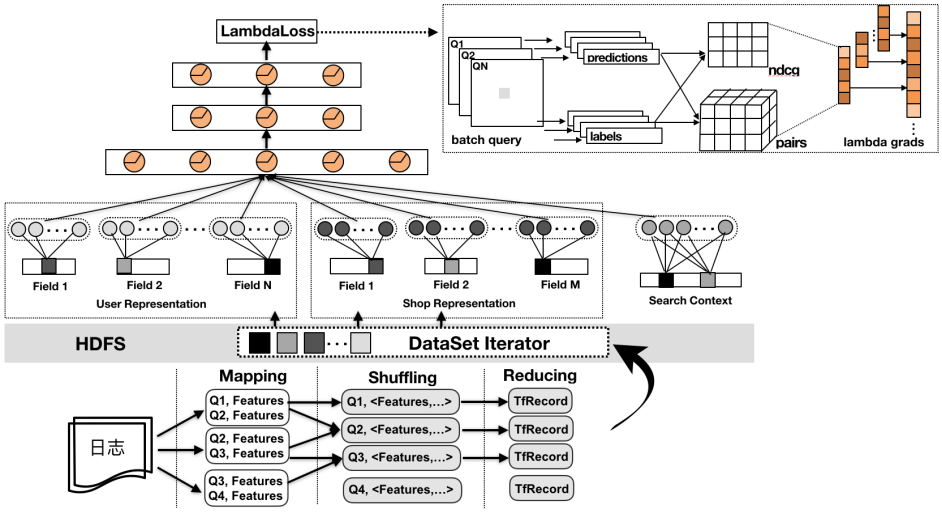


图 11　Lambda 梯度的分布式实现

为了提升训练效率，我们与基础研发平台数据平台中心紧密协同，一起探索并验证了多项优化操作：

1. 将 ID 类特征的映射等操作一并在预处理中完成，减少多轮 Training 过程中的重复计算。
2. 将样本转 TfRecord，利用 RecordDataSet 方式读取数据并计算处理，Worker 的计算性能大概提升了 10 倍。
3. Concat 多个 Categorical 特征，组合成 Multi-Hot 的 Tensor 进行一次 Embedding_Lookup 操作，减少 Map 操作的同时有助于参数做分片存储计算。
4. 稀疏 Tensor 在计算梯度以及正则化处理时保留索引值，仅对有数值的部分进行更新操作。
5. 多个 PS 服务器间进行分片存储大规模 Tensor 变量，减少 Worker 同步更新的通讯压力，减少更新阻塞，达到更平滑的梯度更新效果。

整体下来，对于 30 亿左右的样本量、上亿级别的特征维度，一轮迭代大概在半小时内完成。适当的增加并行计算的资源，可以达到分钟级的训练任务。

4.4 进一步改进优化目标

NDCG 的计算公式中，折损的权重是随着位置呈指数变化的。然而实际曝光点击率随位置变化的曲线与 NDCG 的理论折损值存在着较大的差异。

对于移动端的场景来说，用户在下拉滑动列表进行浏览时，视觉的焦点会随着滑屏、翻页而发生变动。例如用户翻到第二页时，往往会重新聚焦，因此，会发现第二页头部的曝光点击率实际上是高于第一页尾部位置的。我们尝试了两种方案去微调 NDCG 中的指数位置折损：

1. **根据实际曝光点击率拟合折损曲线**：根据实际统计到的曝光点击率数据，拟合公式替代 NDCG 中的指数折损公式，绘制的曲线如图 12 所示。
2. **计算 Position Bias 作为位置折损**：Position Bias 在业界有较多的讨论，其中 [7][8] 将用户点击商户的过程分为观察和点击两个步骤：a. 用户需要首先看到该商户，而看到商户的概率取决于所在的位置；b. 看到商户后点击商户的概率只与商户的相关性有关。步骤 a 计算的概率即为 Position Bias，这块内容可以讨论的东西很多，这里不再详述。

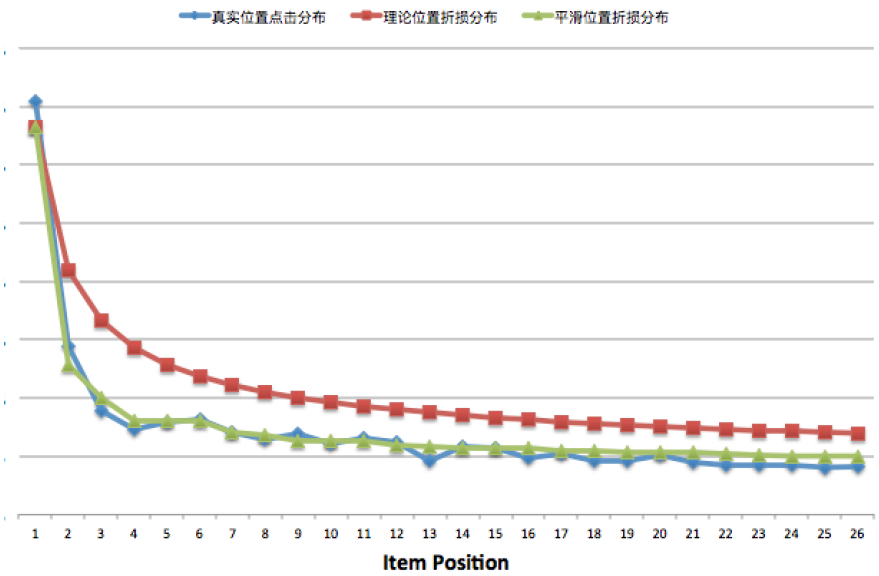


图 12　真实位置折损与理论折损的差别

经过上述对 NDCG 计算改造训练出的 LambdaDNN 模型，相较 Base 树模型和 Pointwise DNN 模型，在业务指标上有了非常显著的提升。

模型	NDCG@10
LambdaMart	0.7141
DNN	0.7378(+3.32%)
LambdaDNN	0.7485(+4.82%)

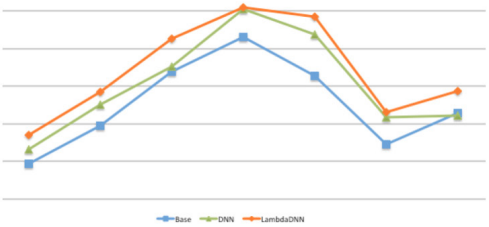


图 13　LambdaDNN 离线 NDCG 指标与线上 PvCtr 效果对比

4.5　Lambda 深度排序框架

Lambda 梯度除了与 DNN 网络相结合外，事实上可以与绝大部分常见的网络结构相结合。为了进一步学习到更多交叉特征，我们在 LambdaDNN 的基础上分别尝试了 LambdaDeepFM 和 LambdaDCN 网络；其中 DCN 网络是一种加入 Cross 的并行网络结构，交叉的网络每一层的输出特征与第一层的原始输入特征进行显性的两两交叉，相当于每一层学习特征交叉的映射去拟合层之间的残差。

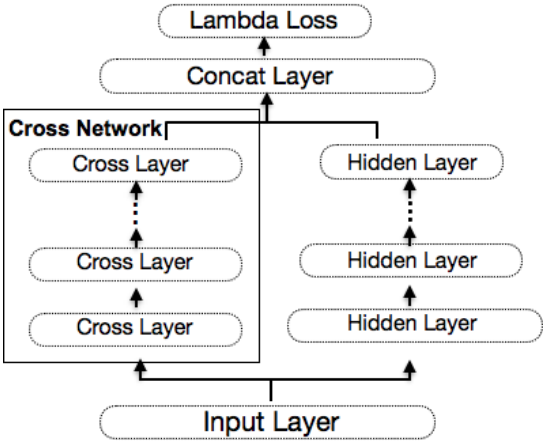


图 14　DCN 模型结构

离线的对比实验表明，Lambda 梯度与 DCN 网络结合之后充分发挥了 DCN 网络的特点，简洁的多项式交叉设计有效地提升模型的训练效果。NDCG 指标对比效

果如下图所示：

模型	NDCG@10
DNN	0.7378
DCN	0.7410(+0.43%)
LambdaDNN	0.7485(+1.45%)
LambdaDCN	0.7496(+1.61%)

图 15　Lambda Loss 与 DCN 网络结果的效果

5. 深度学习排序诊断系统

深度学习排序模型虽然给业务指标带来了大幅度的提升，但由于深度学习模型的“黑盒属性”导致了巨大的解释性成本，也给搜索业务带来了一些问题：

1. **日常搜索 Bad Case 无法快速响应**：搜索业务日常需要应对大量来自于用户、业务和老板们的“灵魂拷问”，“为何这个排序是这样的”，“为什么这家商户质量跟我差不多，但是会排在我的前面”。刚切换到深度学习排序模型的时候，我们对于这样的问题显得手足无措，需要花费大量的时间去定位问题。
2. **无法从 Bad Case 中学习总结规律持续优化**：如果不明白为什么排序模型会得出一个很坏的排序结果，自然也无法定位模型到底出了什么问题，也就无法根据 Bad Case 总结规律，从而确定模型和特征将来的优化方向。
3. **模型和特征是否充分学习无从得知**：新挖掘一些特征之后，通常我们会根据离线评测指标是否有提升决定特征是否上线。但是，即使一个有提升的特征，我们也无法知道这个特征是否性能足够好。例如，模型拟合的距离特征，会不会在特定的距离段出现距离越远反而打分越高的情况。

这些问题都会潜在带来一些用户无法理解的排序结果。我们需要对深度排序模型清晰地诊断并解释。

关于机器学习模型的可解释性研究，业界已经有了一些探索。Lime(Local

Interpretable Model-Agnostic Explanations) 是其中的一种，如下图所示：通过对单个样本的特征生成扰动产生近邻样本，观察模型的预测行为。根据这些扰动的数据点距离原始数据的距离分配权重，基于它们学习得到一个可解释的模型和预测结果[5]。举个例子，如果需要解释一个情感分类模型是如何预测“我讨厌这部电影”为负面情感的，我们通过丢掉部分词或者乱序构造一些样本预测情感，最终会发现，决定“我讨厌这部电影”为负面情感的是因为“讨厌”这个词。

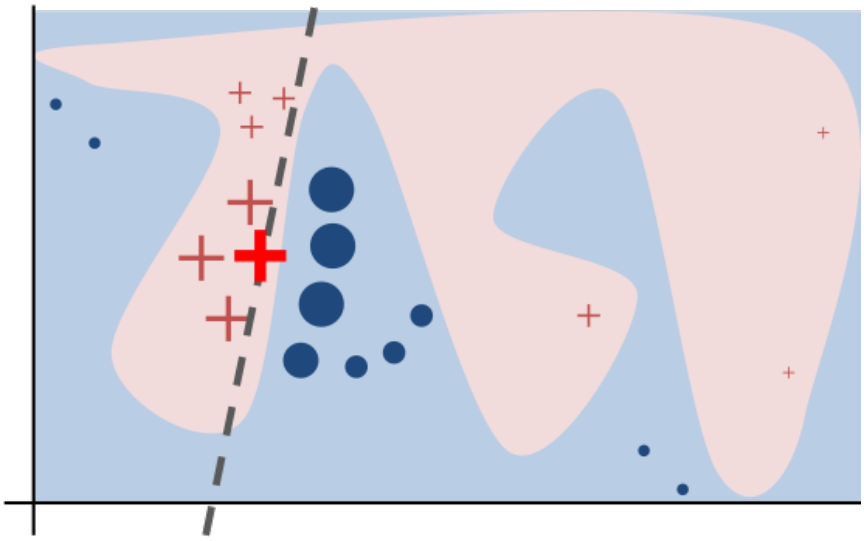


图 16　Lime 解释器的工作原理

基于 Lime 解释器的思想，我们开发了一套深度模型解释器工具——雅典娜系统。目前雅典娜系统支持两种工作模式，Pairwise 和 Listwise 模式：

1. Pairwise 模式用来解释同一个列表中两个结果之间的相对排序。通过对样本的特征进行重新赋值或者替换等操作，观察样本打分和排序位次的变化趋势，诊断出当前样本排序是否符合预期。如下图所示，通过右侧的特征位次面板可以快速诊断出为什么“南京大牌档”的排序比“金时代顺风港湾”要更靠前。第一行的特征位次信息显示，若将“金时代顺风港湾”的 1.3km 的距离特征用“南京大牌档”的 0.2km 的距离特征进行替换，排序位次将上升 10 位；由此得出，“南京大牌档”排在前面的决定性因素是因为距离近。

2. Listwise 模式与 Lime 的工作模式基本类似，通过整个列表的样本生成扰动样本，训练线性分类器模型输出特征重要度，从而达到对模型进行解释的目的。

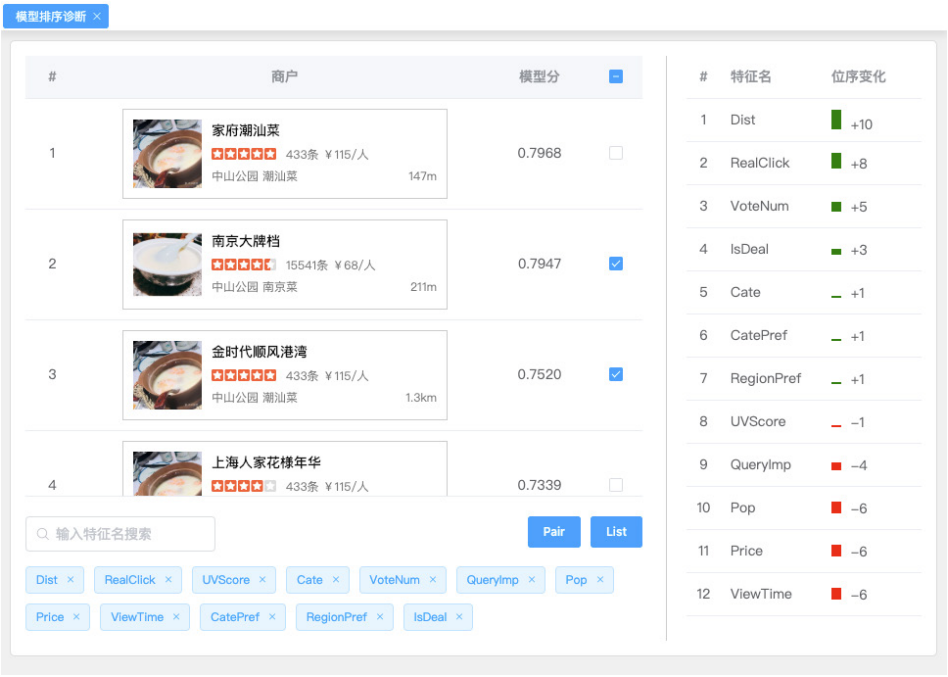


图 17　深度学习排序诊断系统：雅典娜

6. 总结与展望

2018 年下半年，点评搜索完成了从树模型到大规模深度学习排序模型的全面升级。团队在深度学习特征工程、模型结构、优化目标以及工程实践上都进行了一些探索，在核心指标上取得了较为显著的收益。当然，未来依然有不少可以探索的点。

在特征层面，大量知识图谱提供的标签信息尚未充分挖掘。从使用方式上看，简单以文本标签的形式接入，损失了知识图谱的结构信息，因此，Graph Embedding 也是未来需要尝试的方向。同时团队也会利用 BERT 在 Query 和商户文本的深层语义表达上做一些工作。

模型结构层面，目前线上依然以全连接的 DNN 网络结构为主，但 DNN 网络结构在低秩数据的学习上不如 DeepFM 和 DCN。目前 LambdaDeepFM 和

LambdaDCN 在离线上已经取得了收益，未来会在网络结构上做进一步优化。

在模型优化目标上，Lambda Loss 计算损失的时候，只会考虑 Query 内部有点击和无点击的样本对，大量无点击的 Query 被丢弃，同时，同一个用户短时间内在不同 Query 下的行为也包含着一些信息可以利用。因此，目前团队正在探索综合考虑 Log Loss 和 Lambda Loss 的模型，通过 Multi-Task 和按照不同维度 Shuffle 样本让模型充分学习，目前我们已经在线下取得了一些收益。

最后，近期 Google 开源的 TF Ranking 提出的 Groupwise 模型也对我们有一些启发。目前绝大部分的 Listwise 方法只是体现在模型训练阶段，在打分预测阶段依然是 Pointwise 的，即只会考虑当前商户相关的特征，而不会考虑列表上下文的结果，未来我们也会在这个方向上进行一些探索。

参考资料

1. 美团大脑：知识图谱的建模方法及其应用
2. Wide & Deep Learning for Recommender Systems
3. Deep Residual Learning for Image Recognition
4. Attention Is All You Need
5. Local Interpretable Model-Agnostic Explanations: LIME
6. From RankNet to LambdaRank to LambdaMART: An Overview
7. A Novel Algorithm for Unbiased Learning to Rank
8. Unbiased Learning-to-Rank with Biased Feedback
9. Real-time Personalization using Embeddings for Search Ranking at Airbnb

作者简介

非易，2016 年加入美团点评，高级算法工程师，目前主要负责点评搜索核心排序层的研发工作。

祝升，2016 年加入美团点评，高级算法工程师，目前负责点评搜索核心排序层的研发工作。

汤彪，2013 年加入美团点评，高级算法专家，点评平台搜索技术负责人，致力于深层次查询理解和大规模深度学习排序的技术落地。

张弓，2012 年加入美团点评，美团点评研究员。目前主要负责点评搜索业务演进，及集团搜索公共服务平台建设。

仲远，博士，美团 AI 平台部 NLP 中心负责人，点评搜索智能中心负责人。在国际顶级学术会议发表论文 30 余篇，获得 ICDE 2015 最佳论文奖，并是 ACL 2016 Tutorial “Understanding Short Texts” 主讲人，出版学术专著 3 部，获得美国专利 5 项。此前，博士曾担任微软亚洲研究院主管研究员，以及美国 Facebook 公司 Research Scientist。曾负责微软研究院知识图谱、对话机器人项目和 Facebook 产品级 NLP Service。

大众点评信息流基于文本生成的创意优化实践

忆纯　杨肖　明海　众一　扬威　凤阳

1. 引言

信息流是目前大众点评除搜索之外的第二大用户获取信息的入口，以优质内容来辅助用户消费决策并引导发现品质生活。整个大众点评信息流（下文简称点评信息流）围绕个性化推荐去连接用户和信息，把更好的内容推荐给需要的用户。信息流推荐系统涉及内容挖掘、召回、精排、重排、创意等多层机制和排序。本文主要围绕创意部分的工作展开，并选取其中重要的文本创意优化做介绍，分为三个部分：第一部分阐述几个重点问题，包括创意优化是什么，为什么做，以及挑战在哪里；第二部分讲述领域内的应用及技术进展；第三部分介绍我们创意优化的实践，最后做个总结。

什么是创意优化

创意是一个宽泛的概念，它作为一种信息载体对受众展现，可以是文本、图像、视频等任何单一或多类间的组合，如新闻的标题就是经典的创意载体。而创意优化，作为一种方法，指在原有基础上进一步挖掘和激活资源组合方式进而提升资源的价值。在互联网领域产品中，往往表现为通过优化创意载体来提升技术指标、业务目标的过程，在信息流中落地重点包括三个方向：

1. **文本创意**：在文本方面，既包括了面向内容的摘要标题、排版改写等，也包括面向商户的推荐文案及内容化聚合页。它们都广泛地应用了文本表示和文本生成等技术，也是本文的主要方向。
2. **图像创意**：图像方面涉及到首图或首帧的优选、图像的动态裁剪，以及图像的二次生成等。
3. **其他创意**：包括多类展示理由（如社交关系等）、元素创意在内的额外补充信息。

核心目标与推荐问题相似，提升包括点击率、转化率在内的通用指标，同时需要

兼顾考量产品的阅读体验包括内容的导向性等。关于“阅读体验”的部分，这里不作展开。

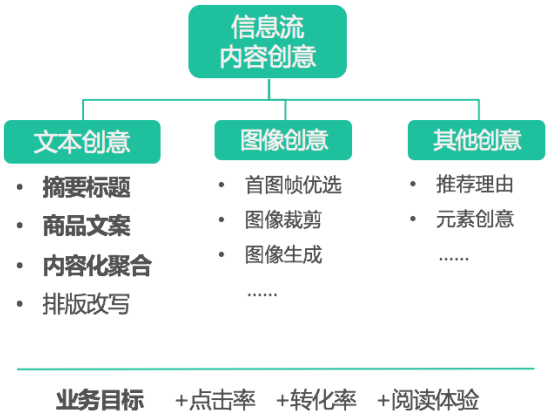


图 1　创意优化的整体应用

为什么要做文本生成

首先文本创意本身为重要的业务发展赋能。在互联网下半场，大众点评平台（下称点评平台）通过内容化去提升用户停留时长，各类分发内容类型在不停地增加，通过优化创意来提升内容的受众价值是必由之路。其次，目前很多内容类型还主要依赖运营维护，运营内容天然存在覆盖少、成本高的问题，无法完全承接需要内容化改造的场景。最后，近几年深度学习在 NLP（Natural Language Processing，自然语言处理）的不同子领域均取得了重大突破。更重要的是，点评平台历经多年，积淀了大量可用的内容数据。从技术层面来说，我们也有能力提供系统化的文本创意生成的解决方案。

对此，我们从文本创意面向对象的角度定义了两类应用形态，分别是面向内容的摘要标题，以及面向商户的推荐文案与内容化聚合页。前者主要应用信息流各主要内容场景，后者则主要应用在信息流广告等内容化场景。这里提前做下产品的简单介绍，帮助大家建立一个立体化的感知。

- **摘要标题**：顾名思义，就是针对某条分发内容生成摘要作标题展示。点评内容

源非常多样，但超过 95% 内容并没有原生标题，同时原生标题质量和多样性等差异也极大。

- **商户文案**：生成有关单个商户核心卖点的描述，一般形式为一句话的短文案。
- **内容聚合**：生成完整的内容页包括标题及多条文案的短篇推荐理由，不同于单商户文案的是，既需要考虑商户的相关性，又要保证理由的多样性。

图 2　文本创意的应用场景

最后需要明确的是，我们做文本创意优化最大的初心，是希望通过创意这个载体显式地连接用户、商户和内容。我们能够知道用户关注什么，知道哪些内容说什么，如何引导用户看，知道哪些商户好、好在哪里，将信息的推荐更进一步。而非为了生成而生成。

面临的挑战

文本创意优化，在业务和技术上分别面临着不同的挑战。首先业务侧，启动创意优化需要两个基础前提：

- 第一，衔接好创意优化与业务目标，因为并不是所有的创意都能优化，也不是所有创意优化都能带来预期的业务价值，方向不对则易蹚坑。
- 第二，创意优化转化为最优化问题，有一定的 Gap。其不同于很多分类排序问

题，本身相对主观，所谓“一千个人眼中有一千个哈姆雷特”，创意优化能不能达到预期的业务目标，这个转化非常关键。

其次，在技术层面，业界不同的应用都面临不一样的挑战，并且尝试和实践对应的解决方案。对文本创意生成来说，我们面临的最大的挑战包括以下三点：

- **带受限的生成** 生成一段流畅的文本并非难事，关键在于根据不同的场景和目标能控制它说什么、怎么说。这是目前挑战相对较大的一类问题，在我们的应用场景中都面临这个挑战。
- **业务导向** 生成能够提升业务指标、贴合业务目标的内容。为此，对内容源、内容表示与建模上提出了更高的要求。
- **高效稳定** 这里有两层含义，第一层是高效，即模型训练预测的效果和效率；第二层是稳定，线上系统应用，需要具备很高的准确率和一套完善的质量提升方案。

2. 文本生成问题综述

我们整体的技术方案演进，可以视作近两年 NLP 领域在深度学习推动下发展的一个缩影。所以在展开之前，先谈一谈整个领域的应用及技术进展。

2.1 相关领域应用

在学界相关领域，文本生成被称为 NLG，其相关任务目标是根据输入数据生成自然语言的文本。而我们在 NLP 领域使用更多的一般是 NLU（Nature Language Understanding 自然语言理解）类任务，如文本分类、命名实体识别等，NLU 的目标则是将自然语言文本转化成结构化数据。NLU 和 NLG 两者表向上是一对相反的过程，但其实是紧密相连的，甚至目前很多 NLU 的任务都受到了生成式模型中表示方法的启发，它们更多只在最终任务上有所区别。

文本生成也是一个较宽泛的概念，如下图所示，广义上只要输出是自然语言文本的各类任务都属于这个范畴。但从不同的输入端可以划分出多种领域应用，从应用相对成熟的连接人和语言的 NMT（神经机器翻译），到 2019 年初，能续写短篇故事的 GPT2 都属于 Text2Text 任务。给定结构化数据比如某些信息事件，来生成

文本比如赛事新闻的属于 Data2Text 类任务，我们的商户文案也属此类。另外还有 Image2Text 等，这块也逐渐在出现一些具有一定可用性又让人眼前一亮的应用，比如各种形式的看图说话。

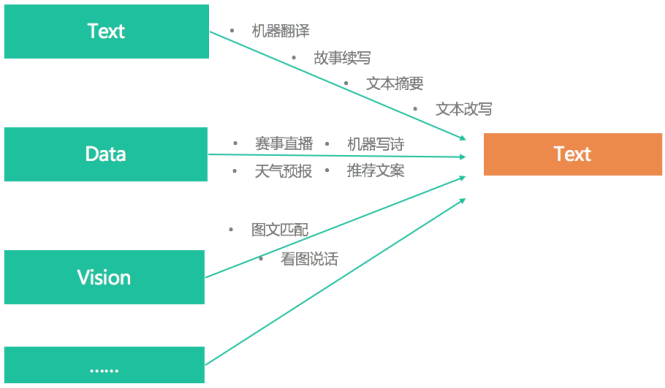


图 3　从输入端划分文本生成的领域应用

2.2　相关技术与进展

文本生成包含文本表示和文本生成两个关键的部分，它们既可以独立建模，也可以通过框架完成端到端的训练。

文本生成

文本生成要解决的一个关键问题，是根据给定的信息如何生成一段文本句子。这是一个简单输入复杂输出的任务，问题的复杂度太大，至今在准确和泛化上都没有兼顾的非常好的方法。2014 年提出的 Seq2Seq Model，是解决这类问题一个非常通用的思路，本质是将输入句子或其中的词 Token 做 Embedding 后，输入循环神经网络中作为源句的表示，这一部分称为 Encoder；另一部分生成端在每一个位置同样通过循环神经网络，循环输出对应的 Token，这一部分称为 Decoder。通过两个循环神经网络连接 Encoder 和 Decoder，可以将两个平行表示连接起来。

另外一个非常重要的，就是 Attention 机制，其本质思想是获取两端的某种权重关系，即在 Decoder 端生成的词和 Encoder 端的某些信息更相关。它也同样可以处理多模态的问题，比如 Image2Text 任务，通过 CNN 等将图片做一个关键特征的向

量表示，将这个表示输出到类似的 Decoder 中去解码输出文本，视频语音等也使用同样的方式（如下图所示）。

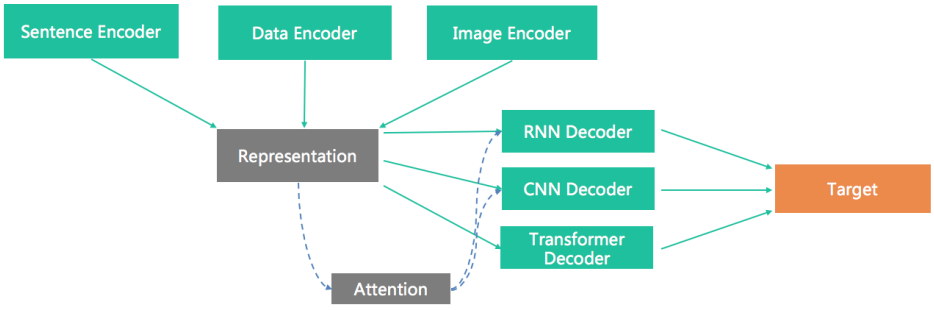


图 4　Seq2Seq 结构图示

可见 Encoder-Decoder 是一个非常通用的框架，它同样深入应用到了文本生成的三种主流方法，分别是规划式、抽取式和生成式，下面看下这几类方法各自的优劣势：

- **规划式**：根据结构化的信息，通过语法规则、树形规则等方式规划生成进文本中，可以抽象为三个阶段。宏观规划解决“说什么内容”，微观规划解决“怎么说”，包括语法句子粒度的规划，以及最后的表层优化对结果进行微调。其优势是控制力极强、准确率较高，特别适合新闻播报等模版化场景。而劣势是很难做到端到端的优化，损失信息上限也不高。
- **抽取式**：顾名思义，在原文信息中抽取一部分作为输出。可以通过编码端的表征在解码端转化为多种不同的分类任务，来实现端到端的优化。其优势在于：能降低复杂度，较好控制与原文的相关性。而劣势在于：容易受原文的束缚，泛化能力不强。
- **生成式**：通过编码端的表征，在解码端完成序列生成的任务，可以实现完全的端到端优化，可以完成多模态的任务。其在泛化能力上具有压倒性优势，但劣势是控制难度极大，建模复杂度也很高。

目前的主流的评估方法主要基于数据和人工评测。基于数据可以从不同角度衡量和训练目标文本的相近程度，如基于 N-Gram 匹配的 BLUE 和 ROUGE 等，基于字符编辑距离（Edit Distance）等，以及基于内容 Coverage 率的 Jarcard 距离等。基于数据的评测，在机器翻译等有明确标注的场景下具有很大的意义，这也是机器翻译领域最先有所突破的重要原因。但对于我们创意优化的场景来说，意义并不大，我们更重要的是优化业务目标，多以线上的实际效果为导向，并辅以人工评测。

另外，值得一提的是，近两年也逐渐涌现了很多利用 GAN（Generative Adversarial Networks，生成对抗网络）的相关方法，来解决文本生成泛化性多样性的问题。有不少思路非常有趣，也值得尝试，只是 GAN 对于 NLP 的文本生成这类离散输出任务在效果评测指标层面，与传统的 Seq2Seq 模型还存在一定的差距，可视为一类具有潜力的技术方向。

文本表示

前文提到，在 Encoder 端包括有些模型在 Decoder 端都需要对句子进行建模，那如何设计一个比较好的模型做表示，既可以让终端任务完成分类、序列生成，也可以做语义推理、相似度匹配等等，就是非常重要的一个部分。那在表示方面，整个 2018 年有两方面非常重要的工作进展：

- **Contextual Embedding**：该方向包括一系列工作，如最佳论文 Elmo(Embeddings from Language Models)，OpenAI 的 GPT(Generative Pre-Training)，以及谷歌大力出奇迹的 BERT(Bidirectional Encoder Representations from Transformers)。解决的核心问题，是如何利用大量的没标注的文本数据学到一个预训练的模型，并通过通过这个模型辅助在不同的有标注任务上更好地完成目标。传统 NLP 任务深度模型，往往并不能通过持续增加深度来获取效果的提升，但是在表示层面增加深度，却往往可以对句子做更好的表征，它的核心思想是利用 Embedding 来表征上下文的的信息。但是这个想法可以通过很多种方式来实现，比如 ELMo，通过双向的 LSTM 拼接后，可以同时得到含上下文信息的 Embedding。而 Transformer 则在

Encoder 和 Decoder 两端，都将 Attention 机制都应用到了极致，通过序列间全位置的直连，可以高效叠加多层（12 层），来完成句子的表征。这类方法可以将不同的终端任务做一个统一的表示，大大简化了建模抽象的复杂度。我们的表示也经历了从 RNN 到拥抱 Attention 的过程。

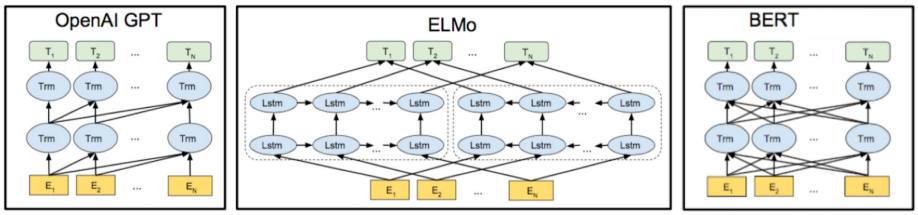


图 5　GPT ELMo BERT 模型结构

- **Tree-Based Embedding**：另外一个流派则是通过树形结构进行建模，包括很多方式如传统的语法树，在语法结构上做 Tree Base 的 RNN，用根结点的 Embedding 即可作为上下文的表征。Tree 本身可以通过构造的方式，也可以通过学习的方式（比如强化学习）来进行构建。最终 Task 效果，既和树的结构（包括深度）有关，也受“表示”学习的能力影响，调优难度比较大。在我们的场景中，人工评测效果并不是很好，仍有很大继续探索的空间。

3. 探索与实践

该部分介绍从 2017 年底至今，我们基于文本生成来进行文本创意优化的一些探索和实践。

3.1　内容源

启动文本生成，首先要了解内容本身，数据的数量和质量对我们的任务重要性无须赘述，这是一切模型的基础。目前我们使用到的数据和大致方法包括：

- **平台渠道**：用户评价、用户笔记、Push、攻略、视频内容、榜单、团单等等。
- **第三方渠道**：合作获取了很多第三方平台的内容来补缺，同时运营侧辅助创意撰写和标注了大量内容，他们同样贡献了可观的数据量。

- **标注数据**：最稀缺的永远是标注数据，尤其是符合业务目标的标注。为此，我们在冷启动阶段设计了 E&E（Explore and Exploit，探索与利用）机制，有意识地积累线上标注，同时尽量引入更多第三方的标注源。

但这些内容的不同特点，也带来了不同的挑战：

- **内容多样**：前面提到的这些内容的结构化程度各不相同，长短差异也极大，对内容表示提出了很高的要求。
- **质量不一**：源内容非常丰富，但事实上质量、质感远远没有达到理想的标准。尤其是占绝对大头的 UGC 的内容，不做好两端的质控将极大影响业务目标的优化，甚至会造成体验问题。
- **聚焦商户**：平台 99% 以上的内容，都以商户作为核心载体，这个对商户的理解和表示同样提出了很高的要求，尤其是在内容化升级的场景下。
- **场景差异**：不同的场景、不同的应用，对模型能力的侧重和优化目标不一样。比如内容和商户，前者要求要有很高的准确率，同时保证优化线上效果；后者更多的是要求有较强的泛化性，并对质感进行优化。

图 7　双平台内容特点与挑战

3.2　基础能力模块

所以，文本创意优化要在业务侧落地产生效果，还需应用到 NLP 领域诸多方向的技术。下图是抽象的整个文本生成应用的基础能力模块，包括用于源和端质量控制的文本质量层，构建 Context 表示的文本表示层，以及面向业务优化的端到端模型

层，其中很多技术应用了公司其他兄弟团队包括内容挖掘组、NLP 中心、离线计算组的出色成果。如针对负面内容过滤的情感分析，多项针对性的文本分类，针对商户表示的标签挖掘等，在这里特别向他们表示感谢。

图 8　文本生成应用的基础能力模块

3.3　信息流标题实践

双平台的内容需要在信息流分发，在创意上最先优化的就是标题，这是用户仅能看到两个要素之一（另一个为首图），而我们超过 95% 的内容并没有原生标题，同时原生标题也存在诸如多样性差非场景导向等问题，还有二次优化的空间。

但是，有两点比较大的挑战，在不同任务上具象可能不一样。它们的本质并没有改变，部分也是业界难点：

- **1. 两个受限条件**：第一，需要以线上点击率转化率为优化目标，线上没效果，写的再好意义都不大；第二，需要与原文强相关，并且容错空间极小，一出现就是 Case。
- **2. 优化评估困难**：第一，模型目标和业务目标间存在天然 Gap；第二，标注数据极度稀缺，离线训练和线上实际预测样本数量之间，往往差距百倍。

对此，我们通过抽取式和生成式的相结合互补的方式，并在流程和模型结构上着手进行解决。

抽取式标题

抽取式方法在用户内容上有比较明显的优势：首先控制力极强，对源内容相关性好，改变用户行文较少，也不容易造成体验问题，可以直接在句子级别做端到端优化。对此，我们把整个标题建模转变为一个中短文本分类的问题，但也无法规避上文提到两个大挑战，具体表现在：

- 在优化评估上，首先标题创意衡量的主观性很强，线上 Feeds 的标注数据也易受到其他因素的影响，比如推荐排序本身；其次，训练预测数据量差异造成 OOV 问题非常突出，分类任务叠加噪音效果提升非常困难。对此，我们重点在语义 + 词级的方向上来对点击 / 转化率做建模，同时辅以线上 E&E 选优的机制来持续获取标注对，并提升在线自动纠错的能力。
- 在受限上，抽取式虽然能直接在 Seq 级别对业务目标做优化，但有时候也须兼顾阅读体验，否则会形成一些“标题党”，亦或造成与原文相关性差的问题。对此，我们抽象了预处理和质量模型，来通用化处理文本创意内容的质控，独立了一个召回模块负责体验保障。并在模型结构上来对原文做独立表示，后又引入了 Topic Feature Context 来做针对性控制。

整个抽取式的流程，可以抽象为四个环节 + 一个在线机制：

图 9　抽取式生成标题流程

1. 源数据在内容中台完成可分发分析后，针对具体内容，进行系统化插件式的预处理，包括分句拼句、繁简转换、大小写归一等，并进行依存分析。
2. 而后将所有可选内容作质量评估，包括情感过滤、敏感过滤等通用过滤，以

及规则判别等涉及表情、冗余字符处理与语法改写的二次基础优化。

3. 在召回模块中，通过实体识别 +TF-IDF 打分等方式来评估候选内容标题基础信息质量，并通过阈值召回来保证基础阅读体验，从而避免一些极端的 Bad Case。
4. 最后，针对候选标题直接做句子级别的点击 / 转化率预估，负责质感、相关性及最终的业务目标的优化。为此，我们先后尝试了诸多模型结构来解决不同问题，下面重点在这方面做下介绍。

我们第一版 Bi-LSTM + Attention 整个结构并不复杂。我们的输入层是 PreTrain 的 Word Embedding，经过双向 LSTM 给到 Attention 层，Dropout 后全连接，套一个交叉熵的 Sigmod，输出判别，但它的意义非常明显，既可以对整句序列做双向语义的建模，同时可以通过注意力矩阵来对词级进行加权。这个在线上来看，无论是对体感还是点击转化率都较召回打分的原始版本，有了巨大提升。而后，我们还在这个 Base 模型基础上，尝试添加过 ELMo 的 Loss，在模型的第一层双向 LSTM 进行基于 ELMo Loss 的 Pre Train 作为初始化结果，在线上指标也有小幅的提升。

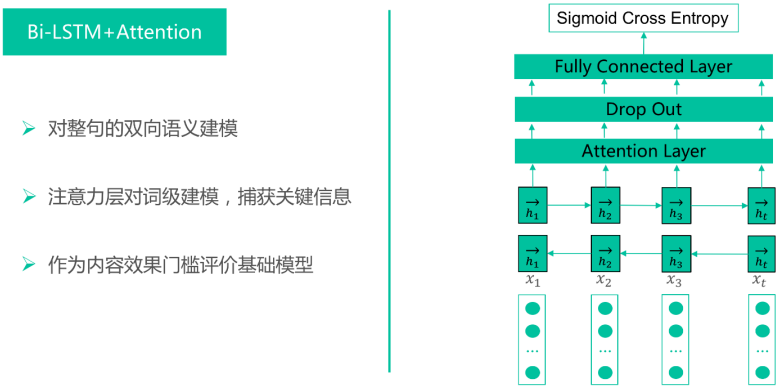


图 10　Bi-LSTM + Attention

但是上述这个结构，将中短文本脱离原文独立建模，显然无法更好地兼顾原文受限这个条件。一个表现，就是容易出现“标题党”、原文不相关等对体验造成影响

的问题。对此，我们在原文与候选标题结合的表示建模方面，做了不少探索，其中以 CNN+Bi-LSTM+Attention 的基模型为代表，但其在相关性建模受原文本身长度的影响较大，而且训练效率也不理想。

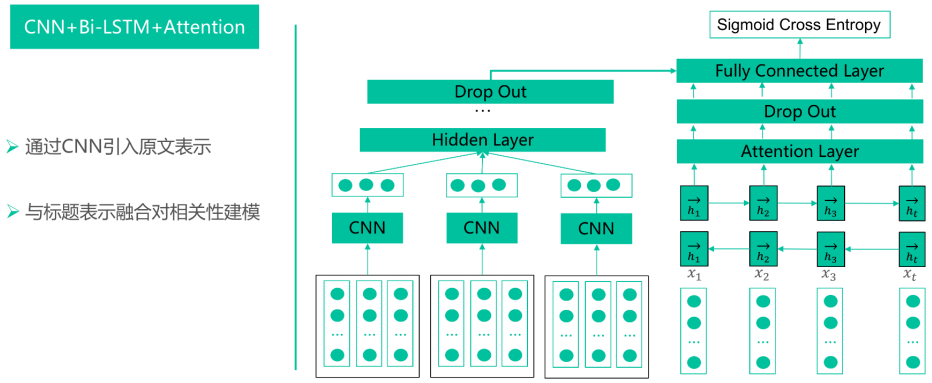


图 11　CNN+Bi-LSTM + Attention

经过一段时间的探索分析，在原文受限问题上，最终既通过深度模型来表征深层的语义，也辅以更多的特征工程，如属性、Topic 等挖掘特征我们统称为 Context，来表征用户能感知到的浅层信息，“两条腿走路”才能被更好的学习，这个在文案生成和标题生成的探索中反过来为抽取式提供了借鉴。

在效率上，我们整体替换了 RNN-LSTM 的循环结构，采用了谷歌那时新提出的自注意力的机制，来解决原文表征训练效率和长依赖问题。采用这个结构在效果和效率上又有了较大的提升。主要问题是，我们的 Context 信息如何更好地建模到 Self-Attention 的结构中。它与生成式模型结构非常类似，在下文生成式部分有所介绍。

另外，需要说明的一点是，除非有两个点以上的巨大提升，一般我们并不会以离线评测指标来评价模型好坏。因为前面提到，我们的标注数据存在不同程度的扰动，而且只是线上预测很小的一个子集，无法避免的与线上存在一定的 Gap，所以我们更关注的是模型影响的基础体验（人工检测通过率即非 Bad Case 率），效率表现（训练预测的时效）最重要的还是线上实际的业务效果。在我们这几个版本的迭代中，这三

个方面都分别获得了不同程度的优化，尤其是包括点击率、总点击量等在内的业务指标，都累计获得了 10% 以上的提升。

	V0.1 TF-IDF	V1.0 Bi-LSTM+Attn	V1.5 CNN+Bi-LSTM+Attn	V2.0 TopicFeature+Self-Attn
业务优化（CTR/CVR/点击曝光）	**+0%**	**+12%**	**+2%**	**+4%**
基础体验（检验通过率）	*+0%*	*+10%*	*+5%*	*+8%*
效率优化（训练预测时效）	*+0%*	*+0%*	*-100%*	*+200%*

图 13　效果数据

受限生成式标题

抽取式标题在包括业务指标和基础体验上都获取了不错的效果，但仍有明显的瓶颈。第一，没有完全脱离原文，尤其在大量质量欠优内容下无法实现创意的二次优化；第二，更好的通过创意这个载体显式的连接用户、商户和内容，这个是生成式标题可以有能力实现的，也是必由之路。

生成式标题，可以抽象描述为：在给定上文并在一定受限条件下，预估下个词的概率的问题。在信息流标题场景，抽取式会面临的问题生成式全部会继承，且在受限优化上面临更大的挑战：

- 原文受限，首先只有表示并学习到原文的语义意图才能更好的控制标题生成，这个本身在 NLU 就是难点，在生成式中就更为突出；其次，标注数据稀缺，原文＋标题对的数据极少，而大部分又存在于长文章。为了保证控制和泛化性，我们初期将标题剥离原文独立建模，通过 Context 衔接，这样能引入更多的非标数据，并在逐步完成积累的情况下，才开始尝试做原文的深度语义表示。
- 优化评估，受限生成式对训练语料的数量和质量要求高很多，首先要保证基础的语义学习也要保证生成端的质量；其次，生成式本质作为语言模型无法在句子层面对业务目标直接做优化，这中间还存在一道 Gap。

在表示上，前面已经提到，我们经历过目标单独建模和结合原文建模的过程，主要原因还是在于仅针对 Target 的理解去构建 Context 衔接，非常容易出现原文相关性问题。所以我们在描述的泛化性方向也做了不少的尝试，比如尽可能地描述广而泛

主题。诸如“魔都是轻易俘获人心的聚餐胜地”，因为只面向上海的商户，内容符合聚餐主题，泛化能力很强，但仍然不能作为一个普适的方案解决问题。

下图为我们一个有初步成效的 RNN-Base 的 Seq2Seq 模型的整体结构。Encoder 端使用的是，包括前面提到的主题（包括商户信息）表示以及原文的双向语义表示，两部分的拼接构成的 Context，输出给注意力层。Decoder 端生成文本时，通过注意力机制学习主题和原文表示的权重关系，这个结构也完整应用到了文案生成，其中控制结构会在文案中展开介绍。

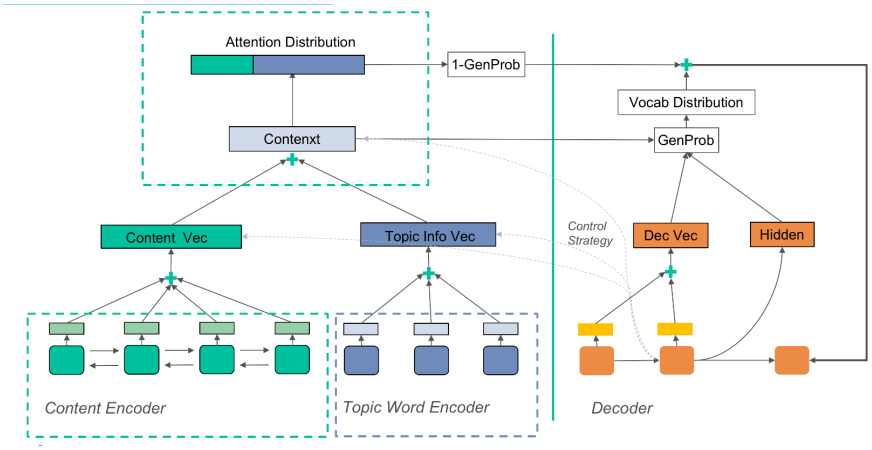


图 14　LSTM Attention Based Seq2Seq 模型结构

在序列建模上，我们经历了一个从 RNN 到自注意力的过程。简单介绍下，序列建模一个核心要点是如何建模序列间的长依赖关系。影响它的重要因素是，信号在网络正向和反向计算中传递的长度（也就是计算次数），较长的依赖关系消失越严重。而在自注意力结构中，每一层都直接与前一层的所有位置直接连接，因此依赖长度均为 O(1)，最大程度保留了序列间的依赖关系。

可以看到，Encoder 包括两部分，一部分是 Source 原文，一部分是基于原文和商户理解的主题 Context，两者共同组成。为此，我们借鉴了 NMT 的一部分研究思想，调整了 Transformer 的结构，在原结构上额外引入了 Context Encoder，并且在 Encoder 和 Decoder 端加入了 Context 的 Attention 层，来强化模型捕捉 Context 信息的能力。

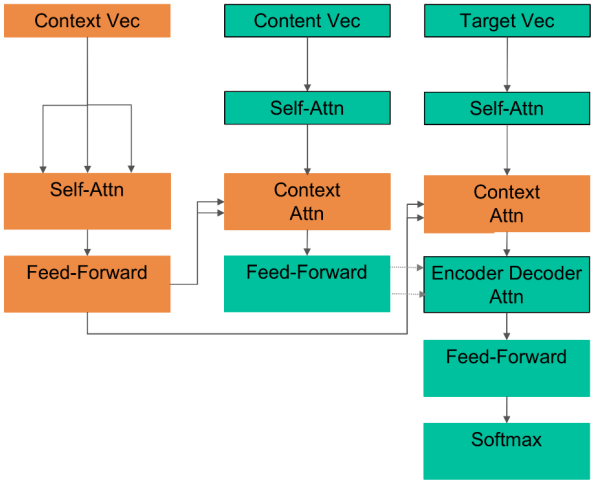


图 15　Transformer Based Seq2Seq Model

我们在生成式方向探索过程中，对低质内容的标题生成，在线上获得了接近 10% 的效果提升，但仍有很多值得进一步的尝试和深挖的空间。

抽取与生成 Combine

在我们的场景中，有两种 Combine 的思路，一个是以业务效果为导向的偏工程化方法，另外一个是我们正在探索的一种 Copy 方法。

工程化的思想非常简洁，在推荐问题上扩充候选，是提升效果的一个可行途径，那生成内容即作为新增的候选集之一，参与整体的预估排序。这个方法能保证最终线上效果不会是负向的，实际上也取得了一定的提升。

另一种方法也是学业界研究的子方向之一，即 Copy 机制，我们也在做重点探索，这里仅作思路的介绍，不再进行展开。

使用 Copy 机制的原始目的，是为了解决生成式的 OOV（超出词表范围）问题。但对于我们的场景来说，大部分的“内容 – 标题”对数据是来自于抽取式，即我们很多标题数据，其实参考了原文。那如何继承这个参考机制，针对业务目标学习何时 Copy 以及 Copy 什么，来更优雅地发挥生成式的优势，就是我们探索 Copy 方法的初衷。我们的方向是对 Copy 和 Generate 概率做独立建模，其中重点解决在受限情况下的“Where To Point”问题。

业务指标与生成式目标的 Gap

我们知道生成式模型其本质是一个 Language Model，它的训练目标是最小化 Word 级别的交叉熵 Loss，而最终我们的需要评价的其实是业务相关的句子级别点击率，这就导致了训练目标和业务指标不一致。

解决这个问题，在我们的场景中有三个可行的方向，第一是在 Context 中显式地标注抽取式模型的 Label，让模型学习到两者的差异；第二是在预测 Decoder 的 Beam Search 计算概率的同时，添加一个打分控制函数；第三则是在训练的 Decoder 中，建立一个全局损失函数参与训练，类似于 NMT 中增加的 Coverage Loss。

考虑到稳定性和实现成本，我们最终尝试了第一和第二种方式，其中第二种方式还是从商户文案迁移过来的，也会在下文进行介绍。在线上，这个尝试并没有在 Combine 的基础上取得更好的效果，但同样值得更加深入的探索。

在线 E&E 机制

最后，介绍一下前面提到过的标题 E&E（Explore and Exploit，探索与利用）机制，用来持续获取标注数据，并提升在线自动纠错的能力。我们采用了一种贪心的 Epsilon Greedy 策略，并做了一点修改，类似经典的 Epsilon 算法，区别是引入创意状态，根据状态将 Epsilon 分成多级。目的是将比较好的创意可以分配给较大概率的流量，而不是均分，差的就淘汰，以此来提升效率。在初期优化阶段，这种方式发挥了很大的作用。

具体我们根据标题和图片的历史表现和默认相比，将状态分成 7 档，从上到下效果表现依次递减，流量分配比例也依次降低，这样可以保证整个系统在样本有噪音的情况下实现线上纠偏。

创意状态	划分原则	Epsilon
Win	曝光充足，且CTR置信大于默认创意	高：70%
Prewin	曝光不充足，且CTR大于默认创意	中：20%
Default	默认创意	
Notsure	曝光不充足	低：10%
Candidate	无曝光	
Prefail	曝光不充足，且CTR低于默认创意	
Fail	曝光充足，且CTR置信小于默认创意	0%

图 17　在线 E&E 选优

3.4　商户文案实践

文案作为一个常见的创意形式，在 O2O 以商户为主要载体的场景下有三点需要：第一，赋予商户以内容调性，丰富创意；第二，通过内容化扩展投放的场景；最后，赋能平台的内容化升级，主要业务目标包括点击率、页面穿透率等等。

文案生成和标题生成能够通用整体的生成模型框架，可以归为 Data2Text 类任务，最大区别是由文案的载体”商户”所决定。不同于内容，准确性的要求低很多，复杂度也大大降低，但同时为泛化能力提出了更高的要求，也带来了与内容生成不同的问题。首先在表示上，对商户的结构化理解变得尤其关键；其次在控制上，有 D2T 任务特有且非常重要的控制要求。前文也提到了生成一段文本从来不是难点，重要的是如何按照不同要求控制 Seq 生成的同时，保证很好的泛化性。下文也会分别介绍卖点控制、风格控制、多样性控制控制等几个控制方法。实现这样的控制，也有很多不同的思路。

商户表示

商户的表示抽象为 Context，如下图中所示，主要分两部分。

第一部分来源于商户的自身理解，一部分则来源于目标文本，两部分有一定交集。其中商户理解的数据为卖点或者 Topic，在初期，为了挖掘商户卖点和 Topic，我们主要使用成本较低、无需标注的 LDA。但是它的准确性相对不可控，同时对产出的卖点主题仍需要进行人工的选择，以便作为新的标注，辅助后续扩展有监督的任

务。我们通过 Key 和 Value 两个 Field，来对卖点和主题进行共同表达（也存在很多只有 Value 的情况），比如下图这个商户“菜品”是个 Key，“雪蟹”是 Value，“约会”则仅是 Value。随着时间的推移，后续我们逐渐利用平台商户标签和图谱信息，来扩展商户卖点的覆盖，以此丰富我们的输入信息。该部分在内容挖掘和 NLP 知识图谱的相关介绍中都有涉及，这里不再进行展开。

第二部分目标文本来源，特意添加这部分进入 Context，主要有三方面原因：

- 第一，仅仅依靠商户理解的 Context，在训练过程中 Loss 下降极慢，并且最终预测生成多样性不理想。本质原因是，目标文本内容与商户卖点、主题间的相关性远远不够。通过不同商户的集合来学习到这个表示关系，非常困难。
- 第二，拓宽可用数据范围，不受商户评论这类有天然标注对的数据限制，从商户衔接扩展到卖点衔接，引入更多的泛化描述数据，比如各类运营文案等等。
- 第三，这也是更为重要的一点，能够间接地实现卖点选择的能力，这个会在下文进行介绍。

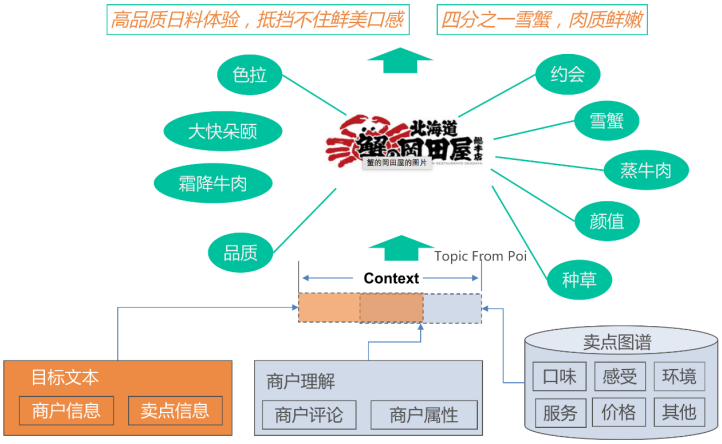


图 18　商户表示

控制端实现

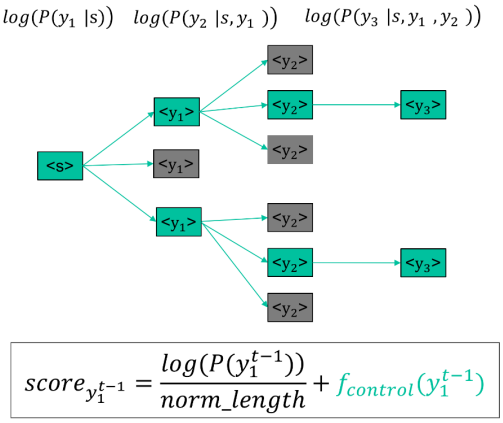
控制，在解码端表现为两类，一类我们称之为 Hard Constrained（强控制），即在数据端给定（或没有给定）的信息，一定要在解码端进行（或不进行）相应描述，这

个适用于地域类目等不能出错的信息。比如这家商户在上海，生成时不能出现除上海以外的地域信息，否则容易造成歧义。另一类称之为 Soft Constrained（弱控制），不同于 NMT 问题，在文案生成上即便是完全相同的输入，不同的输出都是允许的，比如同一商户，最终的文案可以选择不同的卖点去描述不同的内容。

这类同属受限优化的问题，前文提到过有两个思路方向：第一，通过构建机制来让模型自己学习到目标；第二，在 Decoder 的 Beam Search 阶段动态地加入所需的控制目标。我们使用两者相结合的方法，来完成最终的不同控制的实现。

- **两端机制设计**：在具体机制实现上，主要依赖在 Input Context 和 Output Decoder 两端同时生效，让 Context 的 Hard Constrained 来源于 Output，从而使 Model 能够自动学习到强受限关系；而 Soft Constrained 则通过贝叶斯采样的方法，动态添加进 Context，从而帮助 Model 提升泛化能力。
- **Decoder 控制**：简单介绍下 Beam Search，前面提到过，文本生成的预测过程是按 Word 级进行的，每轮预测的候选是整个词汇空间，而往往一般的词表都是十万以上的量级。如果生成序列序列长度为 N，最终候选序列就有十万的 N 次方种可能，这在计算和存储上绝不可行。这时候，就需要使用到 Beam Search 方法，每一步保留最优的前 K（K 一般为 2）个最大概率序列，其他则被剪枝，本质上可以视作一个压缩版的维特比解码。

我们在预测 Beam Search 阶段，除了计算模型概率外，额外增加下图中绿色部分的 Fuction。输入为之前已生成的序列，具体计算逻辑取决于控制目标，可以自由实现。



$$score_{y_1^{t-1}} = \frac{log(P(y_1^{t-1}))}{norm\_length} + f_{control}(y_1^{t-1})$$

图 19　Decoder Beam_Search 控制

下面简单介绍两个重要的控制实现：

- **卖点控制**：这是最重要的一个控制机制，我们整理了涉及到 Hard Constrained 的卖点和实体，重要的如地域、品类等，在目标理解过程中直接加入 Context。对于 Soft Constrained，我们通过卖点的共现计算一个简单的条件概率，并将卖点依此条件概率随机添加进 Context 中，从而让模型通过注意力学习到受限关系。最后在 Decoder fuction 部分，我们新增了一个 Hard&Soft Constrained 的匹配打分项，参与最终的概率计算。最终的实际结果，也非常符合我们的预期。
- **风格控制**：实现方法和卖点控制非常相似，只是这里的风格，其实是通过不同内容之间的差异来间接进行实现。比如大众点评头条、PGC 类的内容与 UGC 类的的写作风格，就存在极大的差异。那么在文案上，比如聚合页标题上可能更需要 PGC 的风格，而聚合页内容上则需要 UGC 的风格。这样的内容属性，即可作为一个 Context 的控制信号，让模型捕获。

3.5　内容聚合

多样性控制

多样性，在文案生成上是一个比较重要和普遍的问题，尤其对于同一个店铺、同

一个卖点或主题同时生成 N 条内容的聚合页来说，更为突出。本质原因是，在解码预测 Beam Search 时永远选择概率最大的序列，并不考虑多样性。但是如果预测时采用 Decoder 概率 Random Search 的方法，则在通顺度上会存在比较大的问题。

对此，我们直接对全局结果进行优化，在预测时把一个聚合页 Context 放到同一个 batch 中，batch_size 即为文案条数，对已经生成序列上进行实体重复检测和 n-gram 重复检测，将检测判重的加一个惩罚性打分，这个简单的思想已经能非常好的解决多样性问题。

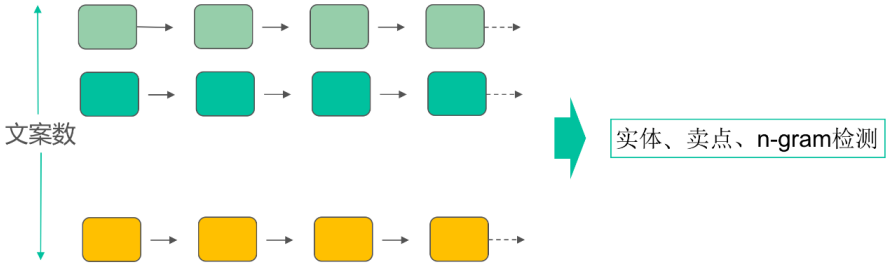


图 20　聚合页多样性控制

4. 动态创意

目前，很多搜索推荐等排序优化场景，都会将创意信息作为特征工程一部分添加进精排或召回模型。那如果把创意优化近似为一个内容级创意排序问题，也可以无缝衔接常用的 Wide&Deep、DNN、FNN 等 CTR 预估模型。但是这之前，需要明确一点非常重要的问题，即它与推荐精排模型的差异，它们之间甚至可能会相互影响，对此，提供下我们的思考。

与精排模型的差异

- 第一，精排模型能否一并完成创意的排序，答案显然是肯定的。但它的复杂度决定了能 Cover 候选集的上限，性能上往往接受不了叉乘创意带来的倍数增长。但此非问题的关键。
- 第二，创意层排序在精排层之前还是之后，直接影响了创意模型的复杂度，也

间接决定了其效果的上限，以及它对精排模型可能的影响程度，从而可能带来全局的影响。此没有最佳实践，视场景权衡。

- 第三，精排模型与创意排序业务目标一致，但实现方式不同。精排模型通过全局排序的最优化来提升业务指标，而创意优化则是通过动态提升内容受众价值来提升业务指标。

最后，我们回到用户视角，当用户在浏览信息流时，其实看到的只有创意本身（标题、图片、作者等信息），但用户却能从中感知到背后的诸多隐含信息，也就是 CTR 预估中的重要内容 / 商户类特征，诸如类目、场景、商户属性等。这个现象背后的本质在于，创意可以表征很多高阶的结构化信息。

基于这一点，在创意优化的特征工程上，方向就很明确了：强化 User/Context，弱化 Item/POI，通过创意表征，来间接学习到弱化的信息从而实现创意层面的最优排序。该部分工作不仅仅涉及到文本，在本文中不再展开。

用户兴趣与文本生成结合的可能性

动态创意为文本生成提供了全新的空间，也提出了更高的要求。动态创意提升受众价值，不仅仅只能通过排序来实现，在正篇介绍的最后部分，我们抛出一个可能性的问题，供各位同行和同学一起思考。也希望能看到更多业界的方案和实践，共同进步。

5. 总结与展望

整个 2018 年，大众点评信息流在核心指标上取得了显著的突破。创意优化作为其中的一部分，在一些方面进行了很多探索，也在效果指标上取得了较为显著的收益。不过，未来的突破，更加任重而道远。

2018 年至 2019 年初，NLP 的各个子领域涌现了非常多令人惊喜的成果，并且这些成果已经落地到业界实践上。这是一个非常好的趋势，也预示着在应用层面会有越来越多的突破。比如 2019 年初，能够续写短篇小说的 GPT2 问世，虽然它真实的泛化能力还未可知，但让我们真切看到了在内容受限下高质量内容生成的可能性。

最后，回到初心，我们希望通过创意的载体显式地连接用户、商户和内容。我们能了解用户关注什么，知道某些内容表达什么，获知哪些商户好，好在哪里，将信息的推荐更进一步。

参考资料

[1] Context-aware Natural Language Generation with Recurrent Neural Networks. arXiv preprint arXiv:1611.09900.
[2] Attention Is All You Need. arXiv preprint arXiv:1706.03762.
[3] Universal Transformers. arXiv preprint arXiv:1807.03819.
[4] A Convolutional Encoder Model for Neural Machine Translation. arXiv preprint arXiv:1611.02344.
[5] Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. arXiv preprint arXiv:1808.08745.
[6] Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
[7] ELMO：Deep contextualized word representations. arXiv preprint arXiv:1802.05365.
[8] openAI GPT：Improving Language Understanding by Generative Pre-Training.
[9] Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.
[10] Tensor2Tensor for Neural Machine Translation. arXiv preprint arXiv:1803.07416.
[11] A Convolutional Encoder Model for Neural Machine Translation. arXiv preprint arXiv:1611.02344.
[12] Sequence-to-Sequence Learning as Beam-Search Optimization. arXiv preprint arXiv:1606.02960.
[13] A Deep Reinforced Model For Abstractive Summarization. arXiv preprint arXiv:1705.04304.
[14] SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. arXiv preprint arXiv:1609.05473.
[15] Generating sequences with recurrent neural networks. CoRR,abs/1308.0850.

作者简介

忆纯，2015 年加入美团点评，算法专家，目前负责点评信息流内容创意工作。
杨肖，博士，2016 年加入美团点评，高级算法专家，点评推荐智能中心内容团队负责人。
明海，2016 年加入美团点评，美团点评研究员，点评推荐智能中心团队负责人。
众一，2016 年加入美团点评，算法研发工程师，目前主要负责点评信息流创意相关算法研发工作。
扬威，2018 年初加入美团点评，算法研发工程师，目前主要负责点评信息流动态创意相关算法研发工作。
凤阳，2016 年加入美团点评，算法研发工程师，目前主要负责点评信息流内容运营算法优化的工作。

AI Challenger 2018：细粒度用户评论情感分析冠军思路总结

程惠阁

2018 年 8 月 –12 月，由美团点评、创新工场、搜狗、美图联合主办的“AI Challenger 2018 全球 AI 挑战赛”历经三个多月的激烈角逐，冠军团队从来自全球 81 个国家、1000 多所大学和公司的过万支参赛团队中脱颖而出。其中“后厂村静静”团队 – 由毕业于北京大学的程惠阁（现已入职美团点评）单人组队，勇夺“细粒度用户评论情感分类”赛道的冠军。本文系程惠阁对于本次参赛的思路总结和经验分享，希望对大家能够有所帮助和启发。

背景

在 2018 全球 AI 挑战赛中，美团点评主要负责了其中两个颇具挑战的主赛道赛题：细粒度用户评论情感分析和无人驾驶视觉感知。其中 NLP 中心负责的细粒度用户评论情感分析赛道，最受欢迎，参赛队伍报名数量最多，约占整个报名团队的五分之一。

细粒度用户评论情感分析赛道提供了 6 大类、20 个细分类的中文情感评论数据，标注规模难度之大，在 NLP 语料特别是文本分类相关语料中都属于相当罕见，这份数据有着极其重要的科研学术以及工业应用价值。

赛题简介

在线评论的细粒度情感分析对于深刻理解商家和用户、挖掘用户情感等方面有至关重要的价值，并且在互联网行业有极其广泛的应用，主要用于个性化推荐、智能搜索、产品反馈、业务安全等。本次比赛我们提供了一个高质量的海量数据集，共包含 6 大类 20 个细粒度要素的情感倾向。参赛人员需根据标注的细粒度要素的情感倾向建立算法，对用户评论进行情感挖掘，组委将通过计算参赛者提交预测值和场景真实值之间的误差确定预测正确率，评估所提交的预测算法。

1. 工具介绍

在本次比赛中，采用了自己开发的一个训练框架，来统一处理 TensorFlow 和 PyTorch 的模型。在模型代码应用方面，主要基于香港科技大学开源的 RNet 和 MnemonicReader 做了相应修改。在比赛后期，还加入了一个基于 BERT 的模型，从而提升了一些集成的效果。

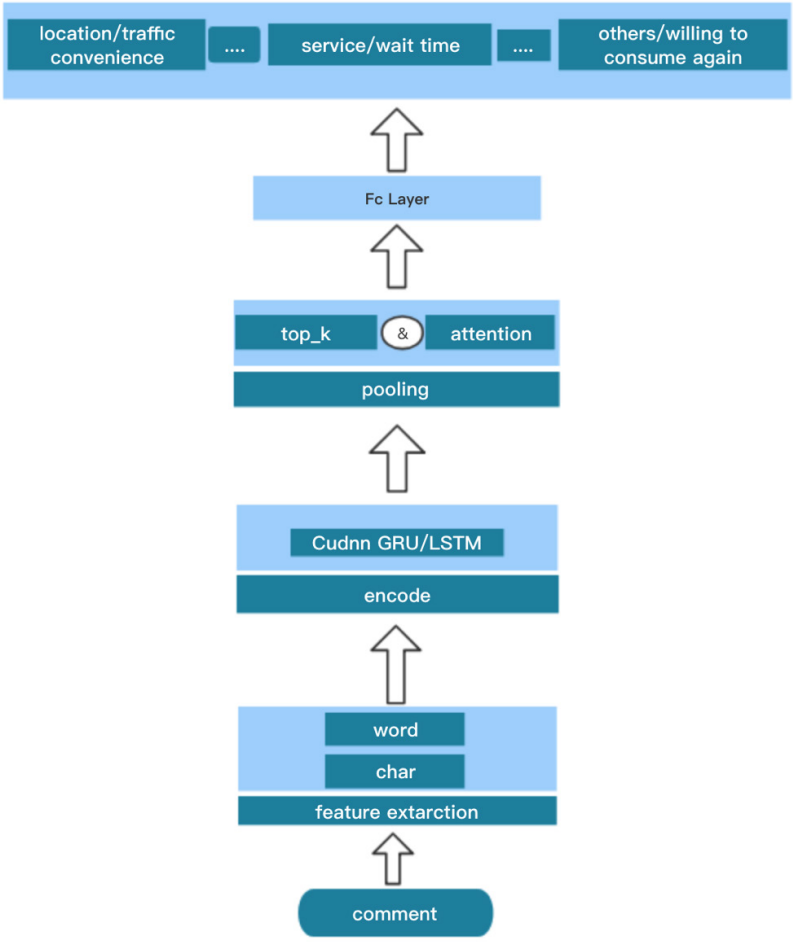
2. 整体思路

整体将该问题看作 20 个 Aspect 的情感多分类问题，采用了传统的文本分类方法，基于 LSTM 建模文本，End2End 多 Aspect 统一训练。

文本分类是业界一个较为成熟的问题，在 2018 年 2 月份，我参加了 Kaggle 的“作弊文本分类”比赛，当时的冠军团队主要依靠基于翻译的数据增强方法获得了成功。2018 年反作弊工作中的一些实践经验，让我意识到，数据是提升文本分类效果的第一关键。因此，我第一时间在网络上寻找到了较大规模的大众点评评论语料，在 Kaggle 比赛的时候，NLP 的语言模型预训练还没有出现，而随着 ELMo 之类模型的成功，也很期待尝试一下预训练语言模型在这个数据集合上的整体效果。

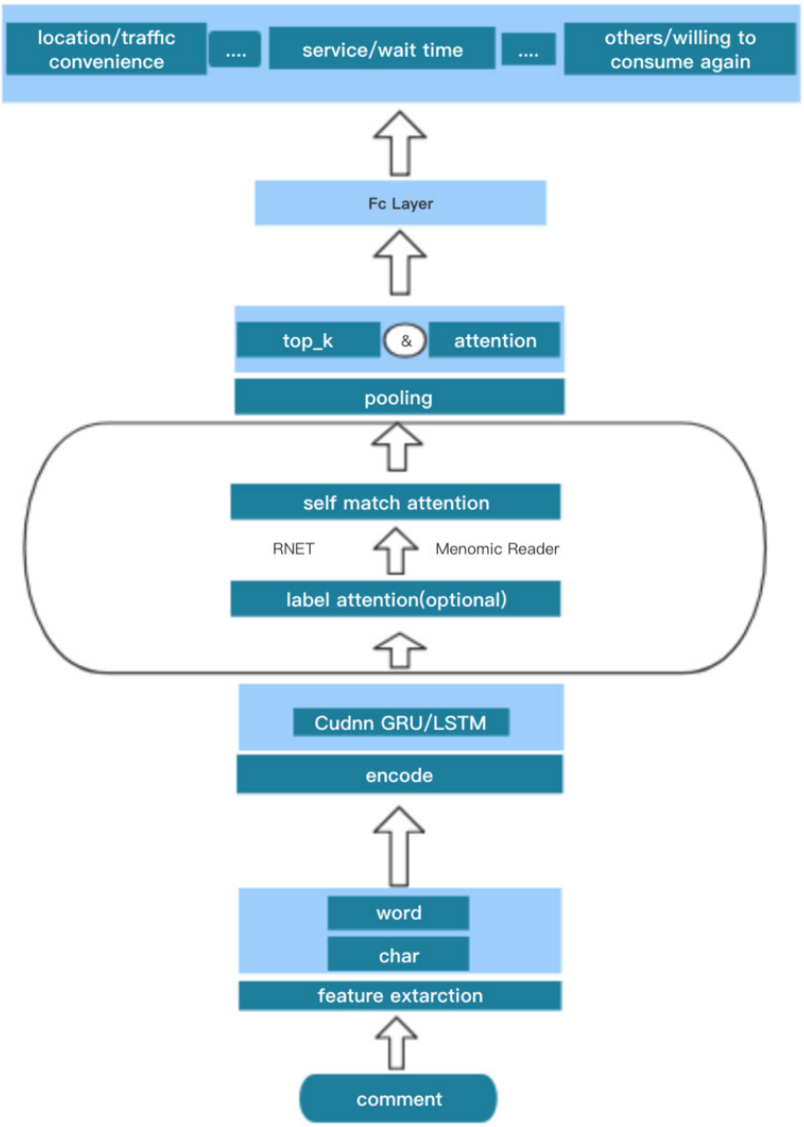
3. 基础模型思路

首先，尝试了不使用预训练语言模型的基础模型，基于 Kaggle Toxic 比赛的经验，直接使用了当时表现最好的 LSTM Encode + Pooling 作为基线模型。在 Kaggle 的比赛中，大家实验的普遍结果是针对中长文本的分类任务的最佳单模型，都是基于 RNN(LSTM/GRU) 或者部分基于 RNN 的模型，比如 RCNN、Capsule + RNN 这样的模型，而其他的模型，比如单纯的 CNN 结构相对表现较差，主要可能是因为 RNN 模型能更好地捕获相对较长距离的顺序信息。



4. 模型层面优化

在基线模型的基础上，效仿阅读理解常见的做法，增加了 Self Attention 层（计算文本到文本自身的 Attention 权重），并将 Attention 之后的输出和原始 LSTM 输出，采用 Gate(RNet) 或者 Semantic Fusion(MnemonicReader) 的方式进行融合。



5. 模型细节处理

更宽的参数更多的模型效果更好

- LSTM 效果好于 GRU。
- Hidden size 400 > 200 > 100。
- Topk Pooling + Attention Pooling 的效果好于单独的 Max 或者 Attention Pooling。
- 共享层前置，Pooling 层 和最后 Fc 层不同 aspect 参数独占效果更好（来自赛后实验，以及其他团队经验）。

这里推测主要原因：是这个数据集有 20 个 Aspect，每个 Aspect 分 4 个不同的类别，所需要的参数相对较多。

三角学习率调节效果最佳

- 参考 BERT 开源代码的学习率设置带来较大效果提升。

采用 Word + Char 的词建模方式

- 这种建模方式能结合分词和字符粒度切分的好处，最大限度避免词汇 UNK 带来的损失。
- 注意对比 Kaggle Toxic 比赛那次比赛是英文语料，对应英文，当时的实验结果是 Word + Ngram 的建模效果更好，收敛更快，所以针对不同 NLP 任务，我们需要具体进行分析。

采用尽可能大的词表

和其他团队相比，我采用了更大的词表 14.4W（Jieba 分词），19.8W（Sentence Piece Unigram 分词），依靠外部大众点评评论数据基于 fastText 预训练词向量，能够支持更大的词表。同时为了避免训练过拟合，采用了只 Finetune 训练中高频的词对低频词固定词向量的处理方式。

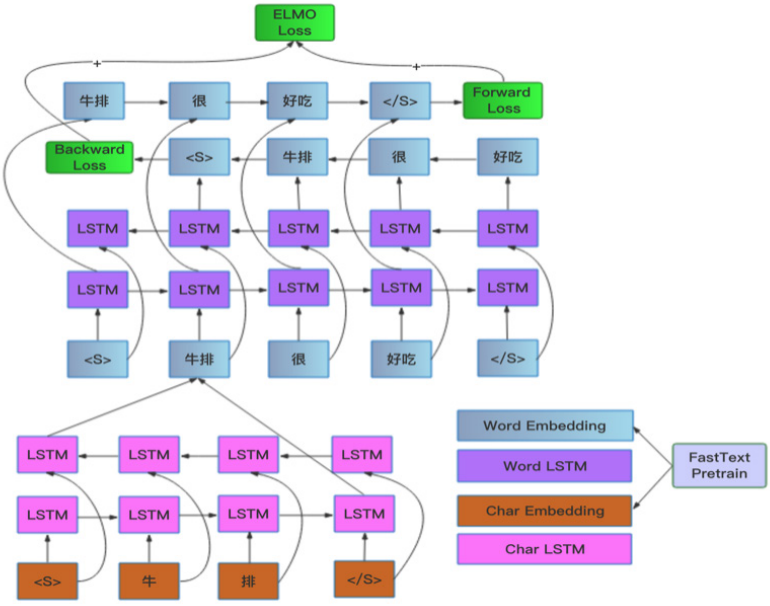
最开始，预计情感相关的词汇相对较少，不需要较大的词表，但是实验过程中发现更大的词表相对地能够提升性能，前提是利用较多的外部数据去比较好的刻画训练数据中低频词的向量。在理论上，我们可以采用一个尽可能大的词表在预测过程中去

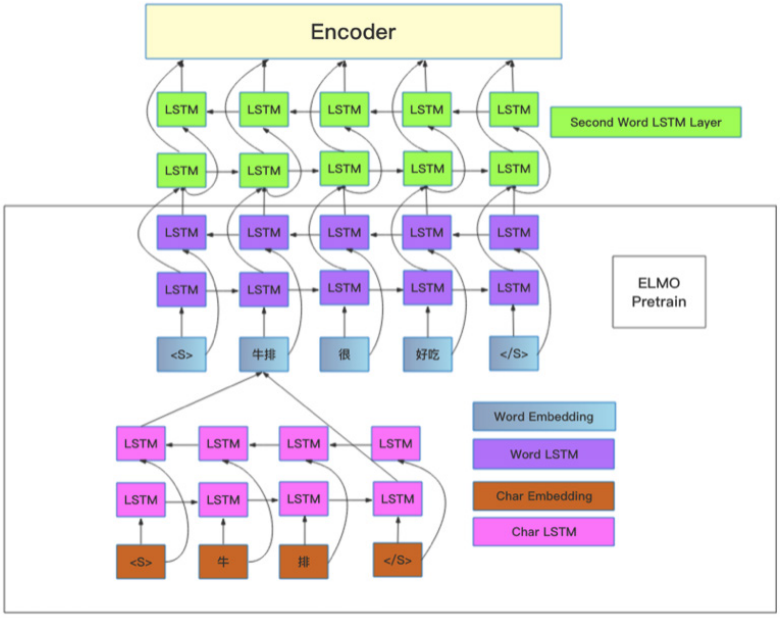
尽可能的减少 UNK 的存在（有论文的结论是对应 UNK 不同的词赋于不同随机向量效果，好于一个固定的 UNK 向量。这里类似，如果我们赋予一个基于无监督外部数据，通过语言模型训练得到的向量则效果更好）。

6. 预训练语言模型

这部分是模型效果提升的关键，这里采用了 ELMo Loss。在简单尝试了官方的 ELMo 版本之后，感觉速度相对比较慢，为此，采用了自己实现的一个简化版的 ELMo，实质上只使用了 ELMo 的 Loss 部分。

在当前双层 LSTM Encoder 的基础上，采用了最小代价的 ELMo 引入，也就是对当前模型的第一层 LSTM 进行基于 ELMo Loss 的预训练，而 Finetune 的时候，模型结构和之前完全不变，只是第一层 LSTM 以及词向量部分采用的 ELMo 预训练的初始化结果，另外在 ELMo 的训练过程中，也采用了基于 fastText 的词向量参数初始化。这个设计使得 ELMo 训练以及 Finetune 训练的收敛，都加快了很多，只需要大概 1 小时的 ELMo 训练，就能在下游任务产生明显受益。值得一提的是，ELMo 和 Self Attention 的搭配在这个数据集合效果非常好。





7. 模型集成

为了取得更好的模型多样性，采用了多种粒度的分词方式，在 Jieba 分词的主要模型基础上，同时引入了基于 SentencePiece 的多种粒度分词。SentencePiece 分词能带来更短的句子长度，但是分词错误相对 Jieba 略多，容易过拟合，因此采用了只 Finetune Char 向量，固定词向量的策略来避免过拟合。多种粒度的分词配合 Word + Char 的建模方式带来了很好的模型多样性。

此外，模型维度的多样性来源自 RNet 结构和 MnemonicReader 结构，以及 BERT 模型的结构的不同。

在模型选择的时候选取了平均 F1 值最优的轮次模型，集成的时候采用了按 Aspect 效果分开加权集成的方式（权重来自 Valid 数据的 F1 分值排序）。基于以上的多样性策略，只需要 7 个单模型集成就能取得较好的效果。

8. 关于 BERT

在实验中基于 Char 的 BERT 单模型，在本次比赛中并没有取得比 ELMo 更好的效果，受限于 512 的长度和只基于 Char 的限制，目前看起来 BERT 模型在这个

数据集合更容易过拟合，Train Loss 下降较快，对应 Valid Loss 效果变差。相信通过适当的优化 BERT 模型能取得更好的效果。

9. 后续优化

F1 的优化是一个有意思的方向。本次比赛中，没有对此做特殊处理，考虑到 F1 是一个全局优化值，如果基于 Batch 强化学习，每个 Batch 可能很难拟合稀有样本分布。

BERT 的进一步优化。因为 BERT 出现之前，基于 Transformer 的模型在长文本分类效果大都是差于基于 LSTM 的模型的，所以如果我们按照 BERT 的 Loss 去预训练基于 LSTM 而不是 Transformer 的模型，在分类问题层面的效果如何？另外，在这个数据集合基于 Transformer 的 BERT，能否取得比 ELMo 更好的分类效果？

对话 AI Challenger 2018 冠军：程惠阁

Q：谈谈对本次参赛的感受？

程惠阁：作为一个多年的算法从业者，我真实的感受到在 AI 时代，技术更新非常之快，比如席卷而来的 ELMo、BERT 等预训练语言模型在工业界影响力之大。包括美团在内的很多公司都快速跟进并上线，而且取得了很好收益，因此技术人员时刻保持学习的心态是非常重要的。

而比赛和工作存在很大的不同，比赛相对更加单纯明确，比赛可以使我在最短时间去学习实验验证一些新的技术，而在标准数据集合验证有效的模型策略，往往在工作中也有实际的价值。对于比赛以及工作中的模型开发，我觉得比较重要的一点首先要做好细致的模型验证部分，在此基础上逐步开发迭代模型才有意义。比如在这次比赛中，我从一开始就监控了包括整体以及各个 Aspect 的包括 F1、AUC、Loss 等等各项指标。

Q：对学习算法的新同学有哪些建议？

程惠阁：如果有时间，可以系统地学习一些名校的深度学习相关的课程，还有很重要的一点，就是实践，我们可以参加去学校项目或者去大公司实习，当然也可以利

用 AI Challenger、Kaggle 这样的竞赛平台进行实践。

Q：为什么会选择参加细粒度用户评论情感分类这个赛道？

程惠阁：因为我之前参加过类似的比赛，并且做过文本分类相关的工作，对这个赛道的赛题也比较感兴趣。

Q：本次比赛最有成就感的事情是什么？

程惠阁：不断迭代提升效果带来的成就感吧，特别是简化版 ELMo 带来的效果提升。

Q：参赛过程中，有哪些收获和成长？

程惠阁：作为一个 TensorFlow 重度用户，我学会了使用 PyTorch 并且体验到 PyTorch 带来的优雅与高效。体验到了预训练语言模型的威力。在比赛中和比赛后，我也收获了很多志同道合的朋友，和他们的交流学习，也帮助我提高了很多。

更重要的是，因为这次比赛，我加入了美团点评这个大家庭，入职这段时间，让我真切地感受到美团点评为了提升用户体验，为了让用户吃的更好，生活更好，在技术方面做了大量的投入。

WSDM Cup 2019 自然语言推理任务获奖解题思路

帅朋

WSDM（Web Search and Data Mining，读音为 Wisdom）是业界公认的高质量学术会议，注重前沿技术在工业界的落地应用，与 SIGIR 一起被称为信息检索领域的 Top2。

刚刚在墨尔本结束的第 12 届 WSDM 大会传来一个好消息，由美团搜索与 NLP 部 NLP 中心的刘帅朋、刘硕和任磊三位同学组成的 Travel 团队，在 WSDM Cup 2019 大赛“真假新闻甄别任务”中获得了第二名的好成绩。队长刘帅朋受邀于 2 月 15 日代表团队在会上作口头技术报告，向全球同行展示了来自美团点评的解决方案。本文将详细介绍他们本次获奖的解决方案。

1. 背景

信息技术的飞速发展，催生了数据量的爆炸式增长。技术的进步也使得了人们获取信息的方式变得更加便捷，然而任何技术都是一把“双刃剑”，信息技术在为人们的学习、工作和生活提供便利的同时，也对人类社会健康持续的发展带来了一些新的威胁。目前亟需解决的一个问题，就是如何有效识别网络中大量存在的“虚假新闻”。

虚假新闻传播了很多不准确甚至虚构的信息，对整个线上资讯的生态造成了很大的破坏，而且虚假新闻会对读者造成误导，干扰正常的社会舆论，严重的危害了整个社会的安定与和谐。因此，本届 WSDM Cup 的一个重要议题就是研究如何实现对虚假新闻的准确甄别，该议题也吸引了全球众多数据科学家的参与。

虽然美团点评的主营业务与在线资讯存在一些差异，但本任务涉及的算法原理是通用的，而且在美团业务场景中也可以有很多可以落地，例如虚假评论识别、智能客服中使用的问答技术、NLP 平台中使用的文本相似度计算技术、广告匹配等。于是，Travel 团队通过对任务进行分析，将该问题转化为 NLP 领域的“自然语言推理”(NLI) 任务，即判断给定的两段文本间的逻辑蕴含关系。因此，基于对任务较为深入理解和平时的技术积累，他们提出了一种解决方案——一种基于多层次深度模型融合框架的虚假新闻甄别技术，该技术以最近 NLP 领域炙手可热的 BERT 为基础模型，并在此基础上提出了一种多层次的模型集成技术。

2. 数据分析

为了客观地衡量算法模型的效果，本届大会组织方提供了一个大型新闻数据集，该数据集包含 32 万多个训练样本和 8 万多个测试样本，这些数据样本均取材于互联网上真实的数据。每个样本包含有两个新闻标题组成的标题对，其中标题对类别标签包括 Agreed、Disagreed、Unrelated 等 3 种。他们的任务就是对测试样本的标签类别进行预测。

“磨刀不误砍柴功”，在一开始，Travel 团队并没有急于搭建模型，而是先对数据进行了全面的统计分析。他们认为，如果能够通过分析发现数据的一些特性，就会有助于后续采取针对性的策略。

首先，他们统计了训练数据中的类别分布情况，如图 1 所示，Unrelated 类别占比最大，接近 70%；而 Disagreed 类占比最小，不到 3%。训练数据存在严重的类别不均衡问题，如果直接用这样的训练数据训练模型，这会导致模型对占比较大类的学习比较充分，而对占比较小的类别学习不充分，从而使模型向类别大的类别进行偏移，存在较严重的过拟合问题。后面也会介绍他们针对该问题提出的对应解决方案。

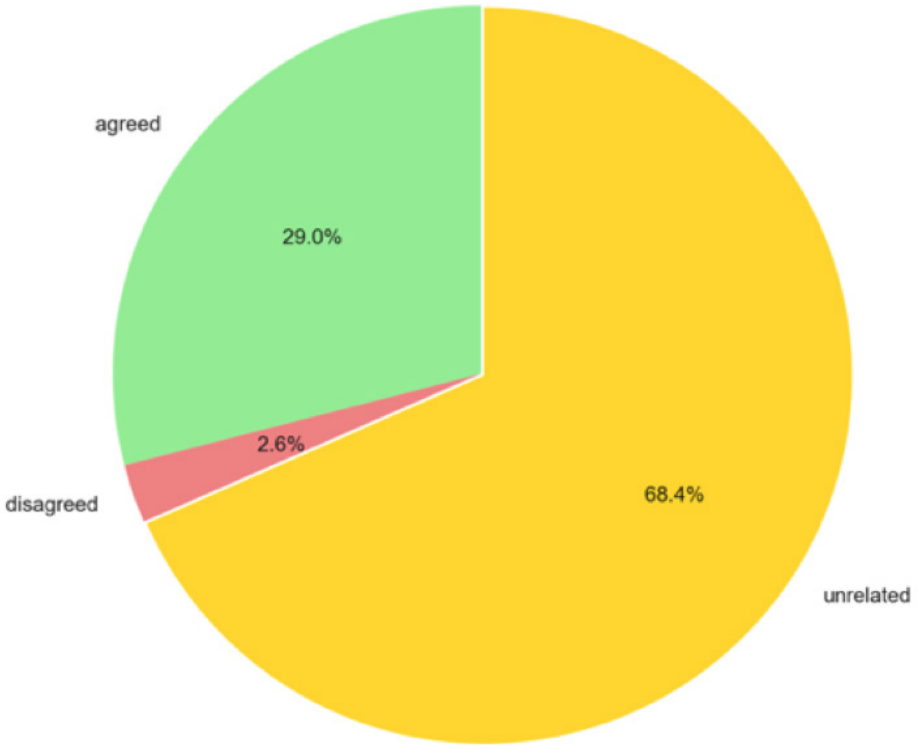


图 1　数据集中类别分布情况

然后，Travel 团队对训练数据的文本长度分布情况进行了统计，如图 2 所示，不同类别的文本长度分布基本保持一致，同时绝大多数文本长度分布在 20 ~ 100 内。这些统计信息对于后面模型调参有着很大的帮助。

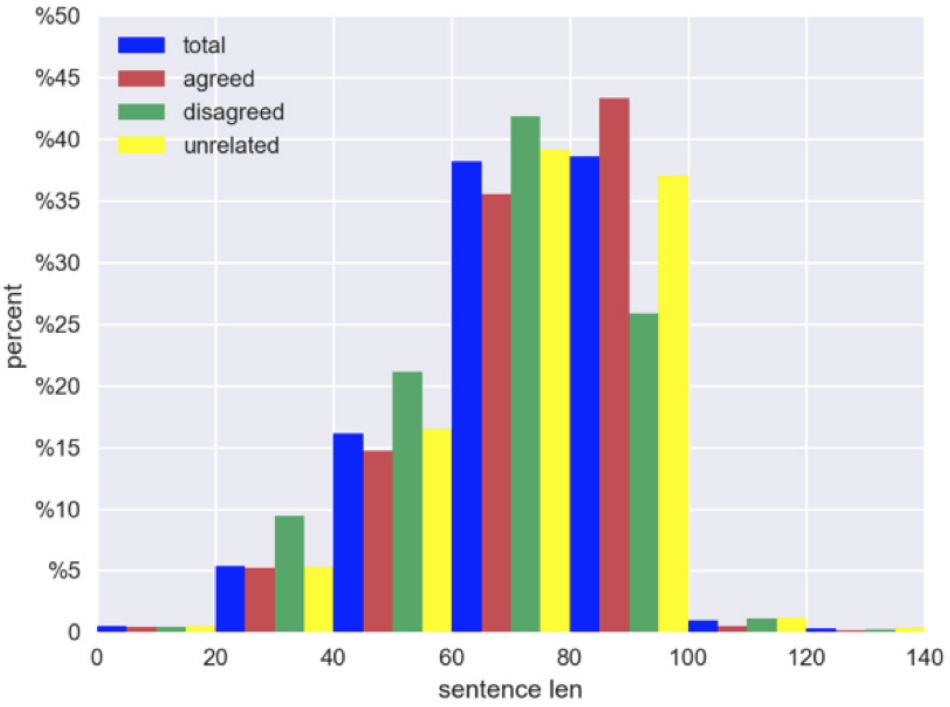


图 2　数据集中文本长度分布情况

3. 数据的预处理与数据增强

本着“数据决定模型的上限，模型优化只是不断地逼近这个上限”的想法，接下来，Travel 团队对数据进行了一系列的处理。

在数据分析时，他们发现训练数据存在一定的噪声，如果不进行人工干预，将会影响模型的学习效果。比如新闻文本语料中简体与繁体共存，这会加大模型的学习难度。因此，他们对数据进行繁体转简体的处理。同时，过滤掉了对分类没有任何作用的停用词，从而降低了噪声。

此外，上文提到训练数据中，存在严重的样本不均衡问题，如果不对该问题做针对性的处理，则会严重制约模型效果指标的提升。通过对数据进行了大量的分析后，他们提出了一个简单有效的缓解样本不均衡问题的方法，**基于标签传播的数据增强方法**。具体方法如图 3 所示：

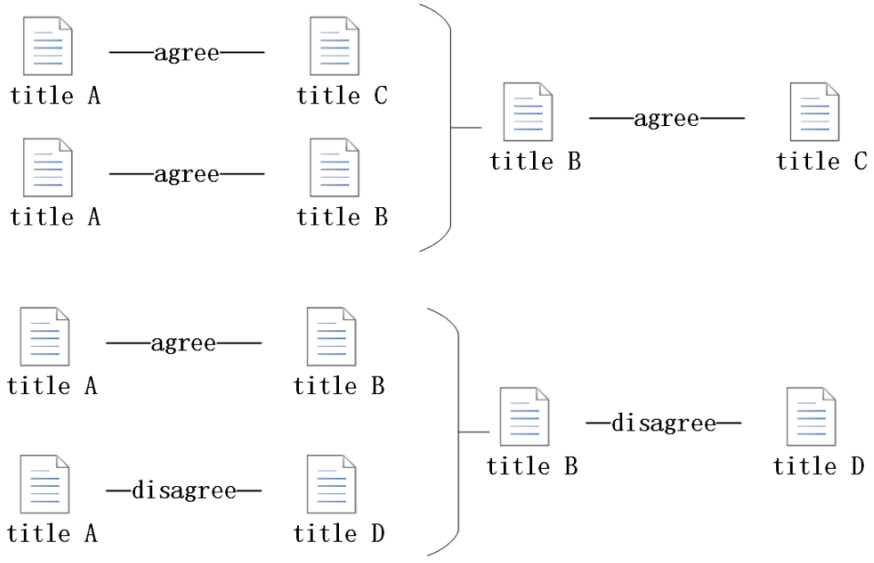


图 3　数据增强策略

如果标题 A 与标题 B 一致，而标题 A 与标题 C 一致，那么可以得出结论，标题 B 与标题 C 一致。同理，如果标题 A 与标题 B 一致，而标题 A 与标题 D 不一致，那么可以得出结论，标题 B 与标题 D 也不一致。此外，Travel 团队还通过将新闻对中的两条文本相互交换位置，来扩充训练数据集。

4. 基础模型

BERT 是 Google 最新推出的基于双向 Transformer 的大规模预训练语言模型，在 11 项 NLP 任务中夺得 SOTA 结果，引爆了整个 NLP 界。BERT 取得成功的一个关键因素是 Transformer 的强大特征提取能力。Transformer 可以利用 Self-Attention 机制实现快速并行训练，改进了 RNN 最被人所诟病的“训练慢”的缺点，可以高效地对海量数据进行快速建模。同时，BERT 拥有多层注意力结构（12 层或 24 层），并且在每个层中都包含有多个“头”（12 头或 16 头）。由于模型的权重不在层与层之间共享，一个 BERT 模型相当于拥有 12 × 12=224 或 24 × 16=384 种不同的注意力机制，不同层能够提取不同层次的文本或语义特征，这可以让 BERT 具有超强的文本表征能力。

本赛题作为典型的自然语言推理（NLI）任务，需要提取新闻标题的高级语义特征，BERT 的超强文本表征能力正好本赛题所需要的。基于上述考虑，Travel 团队的基础模型就采用了 BERT 模型，其中 BERT 网络结构如图 4 所示：

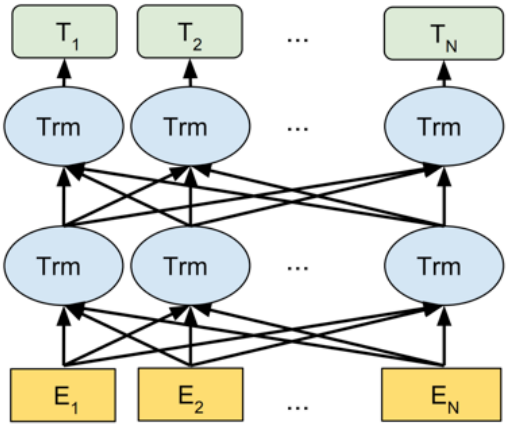


图 4　BERT 网络结构图

在比赛中，Travel 团队在增强后的训练数据上对 Google 预训练 BERT 模型进行了微调（Finetune），使用了如图 5 所示的方式。为了让后面模型融合增加模型的多样性，他们同时 Finetune 了中文版本和英文版本。

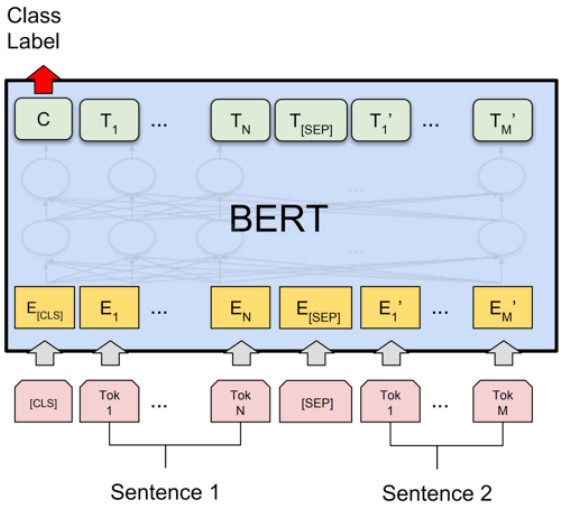


图 5　基于 BERT 的假新闻分类模型结构

5. 多层次深度模型融合框架

模型融合，是指对已有的多个基模型按照一定的策略进行集成以提升模型效果的一种技术，常见的技术包括 Voting、Averaging、Blending、Stacking 等等。这些模型融合技术在前人的许多工作中得到了应用并且取得了不错的效果，然而任何一种技术只有在适用场景下才能发挥出最好的效果，例如 Voting、Averaging 技术的融合策略较为简单，一般来说效果提升不是非常大，但优点是计算逻辑简单、计算复杂度低、算法效率高；而 Stacking 技术融合策略较复杂，一般来说效果提升比较明显，但缺点是算法计算复杂度高，对计算资源的要求较苛刻。

本任务使用的基模型为 BERT，该模型虽然拥有非常强大的表征建模能力，但同时 BERT 的网络结构复杂，包含的参数众多，计算复杂度很高，即使使用了专用的 GPU 计算资源，其训练速度也是比较慢的，因此这就要求在对 BERT 模型融合时不能直接使用 Stacking 这种高计算复杂度的技术，因此我们选择了 Blending 这种计算复杂度相对较低、融合效果相对较好的融合技术对基模型 BERT 做融合。

同时，Travel 团队借鉴了神经网络中网络分层的设计思想来设计模型融合框架，他们想既然神经网络可以通过增加网络深度来提升模型的效果，那么在模型融合中是否也可以通过增加模型融合的层数来提升模型融合的效果呢？基于这一设想，他们提出了一种多层次深度模型融合框架，该框架通过增加模型的层数进而提升了融合的深度，最终取得了更好的融合效果。

具体来说，他们的框架包括三个层次，共进行了两次模型融合。第一层采用 Blending 策略进行模型训练和预测，在具体实践中，他们选定了 25 个不同的 BERT 模型作为基模型；第二层采用 5 折的 Stacking 策略对 25 个基模型进行第一次融合，这里他们选用了支持向量机（SVM）、逻辑回归（LR）、K 近邻（KNN）、朴素贝叶斯（NB），这些传统的机器学习模型，既保留了训练速度快的优点，也保证了模型间的差异性，为后续融合提供了效率和效果的保证；第三层采用了一个线性的 LR 模型，进行第二次模型融合并且生成了最终的结果。模型融合的架构如图 6 所示：

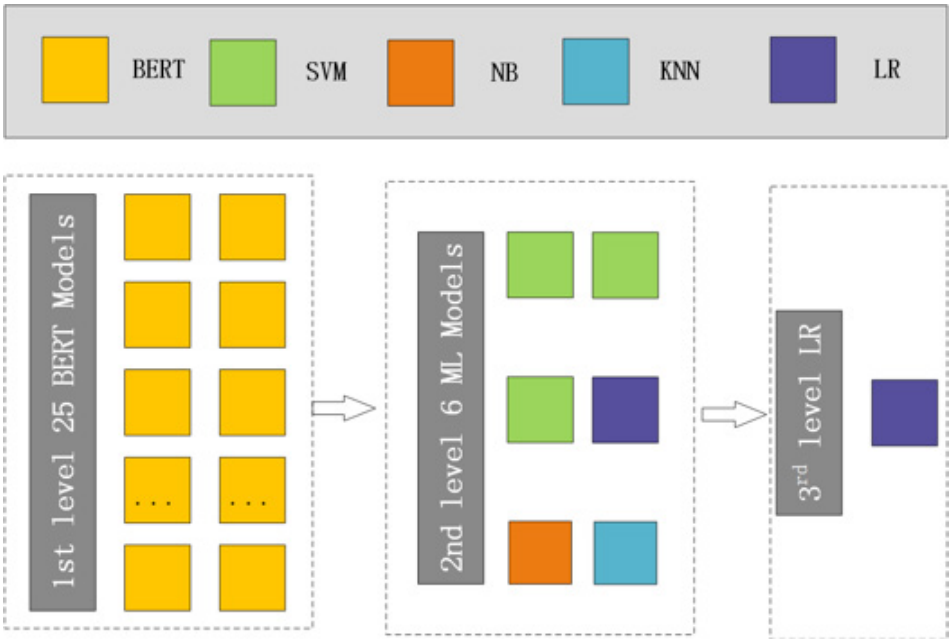


图 6　模型融合架构

整体方案模型训练分为三个阶段，如图 7 所示：

- 第一个阶段，将训练数据划分为两部分，分别为 Train Data 和 Val Data。Train Data 用于训练 BERT 模型，用训练好的 BERT 模型分别预测 Val Data 和 Test Data。将不同 BERT 模型预测的 Val Data 和 Test Data 的结果分别进行合并，可以得到一份新的训练数据 New Train Data 和一份新的测试数据 New Test Data。
- 第二阶段，将上一阶段的 New Train Data 作为训练数据，New Test Data 作为测试数据。本阶段将 New Train Data 均匀的划分为 5 份，使用“留一法”训练 5 个 SVM 模型，用这 5 个模型分别去预测剩下的一份训练数据和测试数据，将 5 份预测的训练数据合并，可以得到一份新的训练数据 NewTrainingData2，将 5 份预测的测试数据采用均值法合并，得到一份新的测试数据 NewTestData2。同样的方法再分别训练 LR、KNN、NB 等模型。
- 第三阶段，将上一阶段的 NewTrainingData2 作为训练数据，NewTestDa-

ta2 作为测试数据，重新训练一个 LR 模型，预测 NewTestData2 的结果作为最终的预测结果。为了防止过拟合，本阶段采用 5 折交叉验证的训练方式。

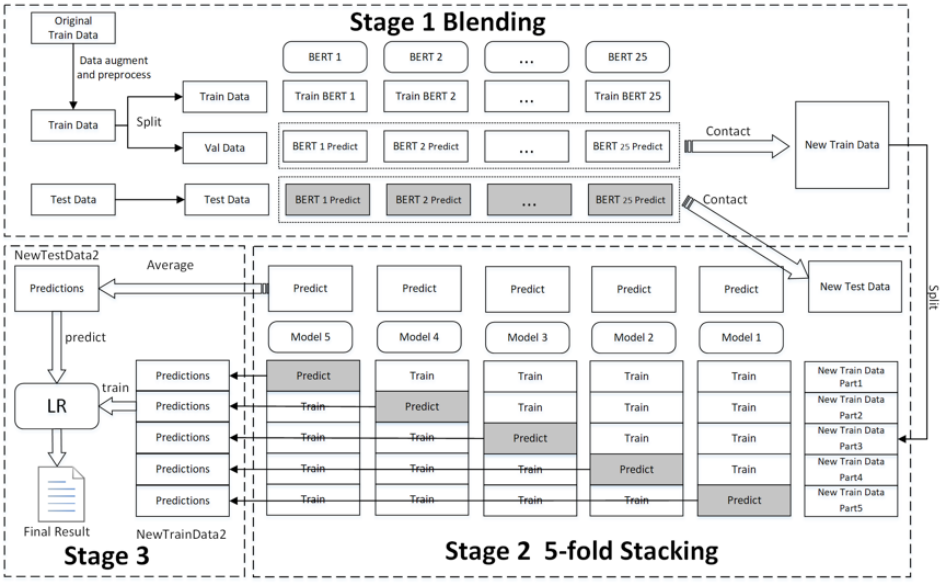


图 7　假新闻分类方案的整体架构和训练流程

6. 实验

6.1　评价指标

为了缓解数据集中存在的类别分布不均衡问题，本任务使用带权重的准确率作为衡量模型效果的评价指标，其定义如下所示：

$$weightedAccuracy(y,\hat{y},\omega)=\frac{1}{n}\sum_{i=1}^{n}\frac{\omega_i(y_i=\hat{y})}{\sum\omega_i}$$

其中，y 为样本的真实类别标签，$\hat{y}$ 为模型的预测结果，ωi 为数据集中第 i 个样本的权重，其权重值与类别相关，其中 Agreed 类别的权重为 1/15，Disagreed 类别的权重为 1/5，Unrelated 类别的权重为 1/16。

6.2 实验结果

在官方测试集上，Travel 团队的最优单模型的准确率达到 0.86750，25 个 BERT 模型简单平均融合后准确率达 0.87700（+0.95PP），25 个 BERT 模型结果以加权平均的形式融合后准确率达 0.87702（+0.952PP），他们提出的多层次模型融合技术准确率达 0.88156（+1.406PP）。实践证明，美团 NLP 中心的经验融合模型在假新闻分类任务上取得了较大的效果提升。

Table 1: Performance of Various Models

Model	Weighted Acc on Private LB
Best Single base model	0.86750
Averaging of 25 BERT	0.87700
Weighted Averaging of 25 BERT	0.87702
Our Empirical Ensemble Model	0.88156

图 8　效果提升

7. 总结与展望

本文主要对解决方案中使用的关键技术进行了介绍，比如数据增强、数据预处理、多层模型融合策略等，这些方法在实践中证明可以有效的提升预测的准确率。由于参赛时间所限，还有很多思路没有来及尝试，例如美团使用的 BERT 预训练模型是基于维基百科数据训练而得到的，而维基百科跟新闻在语言层面也存在较大的差异，所以可以将现有的 BERT 在新闻数据上进行持续地训练，从而使其能够对新闻数据具有更好的表征能。

参考文献

[1] Dagan, Ido, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL recognising textual entailment challenge, Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment. Springer, Berlin, Heidelberg, 177–190.

[2] Bowman S R, Angeli G, Potts C, et al. 2015. A large annotated corpus for learning natural language inference. In proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP).

[3] Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In NAACL.
[4] Rajpurkar P, Zhang J, Lopyrev K, et al. 2016. Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250.
[5] Luisa Bentivogli, Bernardo Magnini, Ido Dagan, Hoa Trang Dang, and Danilo Giampiccolo. 2009. The fifth PASCAL recognizing textual entailment challenge. In TAC. NIST.
[6] Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The winograd schema challenge. In Aaai spring symposium: Logical formalizations of commonsense reasoning, volume 46, page 47.
[7] Bowman, Samuel R., et al. 2015. "A large annotated corpus for learning natural language inference." arXiv preprint arXiv:1508.05326.
[8] Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. R. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. arXiv preprint arXiv:1804.07461.
[9] Chen, Q., Zhu, X., Ling, Z., Wei, S., Jiang, H., & Inkpen, D. 2016. Enhanced lstm for natural language inference. arXiv preprint arXiv:1609.06038.
[10] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, OpenAI.
[11] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
[12] David H. Wolpert. 1992. Stacked generalization. Neural Networks (1992). https: // doi.org/10.1016/S0893- 6080(05)80023- 1.

作者简介

刘帅朋，硕士，美团点评搜索与 NLP 部 NLP 中心高级算法工程师，目前主要从事 NLU 相关工作。曾任中科院自动化研究所研究助理，主持研发的智能法律助理课题获 CCTV-1 频道大型人工智能节目《机智过人第二季》报道。

刘硕，硕士，美团点评搜索与 NLP 部 NLP 中心智能客服算法工程师，目前主要从事智能客服对话平台中离线挖掘相关工作。

任磊，硕士，美团点评搜索与 NLP 部 NLP 中心知识图谱算法工程师，目前主要从事美团大脑情感计算以及 BERT 应用相关工作。

会星，博士，担任美团点评搜索与 NLP 部 NLP 中心的研究员，智能客服团队负责人。目前主要负责美团智能客服业务及智能客服平台的建设。在此之前，会星在阿里达摩院语音实验室作为智能语音对话交互专家，主要负责主导的产品有斑马智行语音交互系统，YunOS 语音助理等，推动了阿里智能对话交互体系建设。

富峥，博士，担任美团点评搜索与 NLP 部 NLP 中心的研究员，带领知识图谱算法团队。目前主要负责美团大脑项目，围绕美团吃喝玩乐场景打造的知识图谱及其应用，能够打通餐饮、旅行、休闲娱乐等各个场景数据，为美团各场景业务提供更加智能的服务。张富峥博士在知识图

谱、个性化推荐、用户画像、时空数据挖掘等领域展开了众多的创新性研究，并在相关领域的顶级会议和期刊上发表 30 余篇论文，如 KDD、WWW、AAAI、IJCAI、TKDE、TIST 等，曾获 ICDM2013 最佳论文大奖，出版学术专著 1 部。

仲远，博士，美团点评搜索与 NLP 部负责人。在国际顶级学术会议发表论文 30 余篇，获得 ICDE 2015 最佳论文奖，并是 ACL 2016 Tutorial“Understanding Short Texts”主讲人，出版学术专著 3 部，获得美国专利 5 项。此前，博士曾担任微软亚洲研究院主管研究员，以及美国 Facebook 公司 Research Scientist。曾负责微软研究院知识图谱、对话机器人项目和 Facebook 产品级 NLP Service。

深度学习在美团配送 ETA 预估中的探索与实践

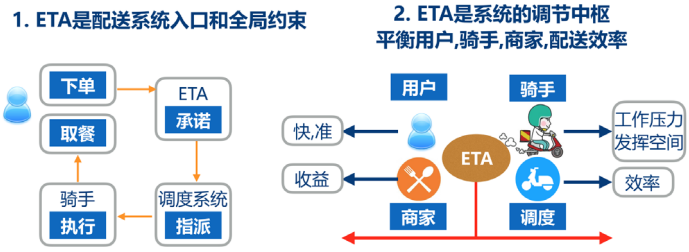
基泽　周越　显杰

1. 背景

ETA（Estimated Time of Arrival，“预计送达时间”），即用户下单后，配送人员在多长时间内将外卖送达到用户手中。送达时间预测的结果，将会以”预计送达时间”的形式，展现在用户的客户端页面上，是配送系统中非常重要的参数，直接影响了用户的下单意愿、运力调度、骑手考核，进而影响配送系统整体成本和用户体验。

对于整个配送系统而言，ETA 既是配送系统的入口和全局约束，又是系统的调节中枢。具体体现在：

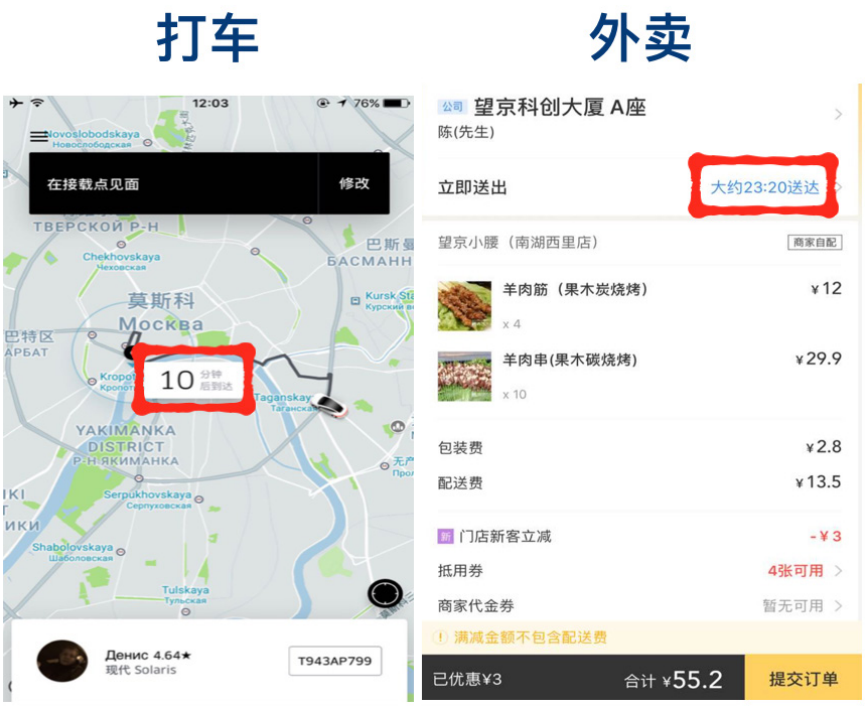
- ETA 在用户下单时刻就需要被展现，这个预估时长继而会贯穿整个订单生命周期，首先在用户侧给予准时性的承诺，接着被调度系统用作订单指派的依据及约束，而骑手则会按照这个 ETA 时间执行订单的配送，配送是否准时还会作为骑手的工作考核结果。
- ETA 作为系统的调节中枢，需要平衡用户 – 骑手 – 商家 – 配送效率。从用户的诉求出发，尽可能快和准时，从骑手的角度出发，太短会给骑手极大压力。从调度角度出发，太长或太短都会影响配送效率。而从商家角度出发，都希望订单被尽可能派发出去，因为这关系到商家的收入。



ETA 在配送系统中作用

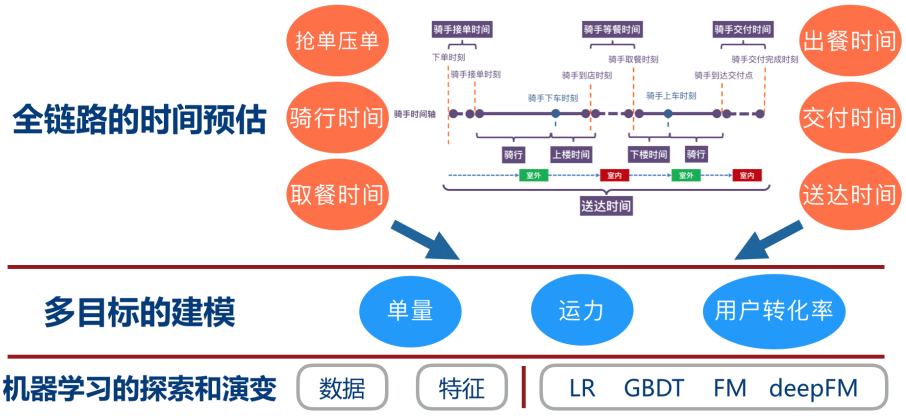
在这样多维度的约束之下，外卖配送的 ETA 的建模和估计会变得更加复杂。与打车场景中的 ETA 做对比，外卖场景的 ETA 面临如下的挑战：

- 外卖场景中 ETA 是对客户履约承诺的重要组成部分，无论是用户还是骑手，对于 ETA 准确性的要求非常高。而在打车场景，用户更加关心是否能打到车，ETA 仅提供一个参考，司机端对其准确性也不是特别在意。
- 由于外卖 ETA 承担着承诺履约的责任，因此是否能够按照 ETA 准时送达，也是外卖骑手考核的指标、配送系统整体的重要指标；承诺一旦给出，系统调度和骑手都要尽力保证准时送达。因此过短的 ETA 会给骑手带来极大的压力，并降低调度合单能力、增加配送成本；过长的 ETA 又会很大程度影响用户体验。
- 外卖场景中 ETA 包含更多环节，骑手全程完成履约过程，其中包括到达商家、商家出餐、等待取餐、路径规划、不同楼宇交付等较多的环节，且较高的合单率使得订单间的流程互相耦合，不确定性很大，做出合理的估计也有更高难度。



外卖及打车中的 ETA

下图是骑手履约全过程的时间轴，过程中涉及各种时长参数，可以看到有十几个节点，其中关键时长达到七个。这些时长涉及多方，比如骑手（接 – 到 – 取 – 送）、商户（出餐）、用户（交付），要经历室内室外的场景转换，因此挑战性非常高。对于 ETA 建模，不光是简单一个时间的预估，更需要的是全链路的时间预估，同时更需要兼顾”单量 – 运力 – 用户转化率”转化率之间的平衡。配送 ETA 的演变包括了数据、特征层面的持续改进，也包括了模型层面一路从 LR–XGB–FM–DeepFM– 自定义结构的演变。



ETA 的探索与演变

具体 ETA 在整个配送业务中的位置及配送业务的整体机器学习实践，请参看《机器学习在美团配送系统的实践：用技术还原真实世界》。

2. 业务流程迭代中的模型改进

2.1 基础模型迭代及选择

与大部分 CTR 模型的迭代路径相似，配送 ETA 模型的业务迭代经历了 LR–> 树模型 –>Embedding–>DeepFM–> 针对性结构修改的路径。特征层面也进行不断迭代和丰富。

- 模型维度从最初考虑特征线性组合，到树模型做稠密特征的融合，到 Embedding 考虑 ID 类特征的融合，以及 FM 机制低秩分解后二阶特征组合，最终通

过业务指标需求，对模型进行针对性调整。

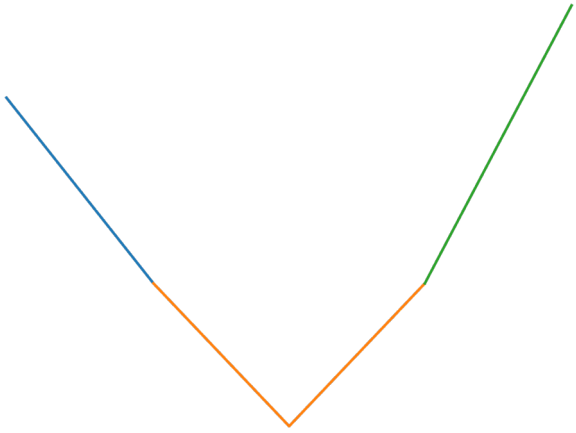
- 特征维度逐步丰富到用户画像 / 骑手画像 / 商家画像 / 地址特征 / 轨迹特征 / 区域特征 / 时间特征 / 时序特征 / 订单特征等维度。

目前版本模型在比较了 Wide&Deep、DeepFM、AFM 等常用模型后，考虑到计算性能及效果，最终选择了 DeepFM 作为初步的 Base 模型。整个 DeepFM 模型特征 Embedding 化后，在 FM（Factorization Machine）基础上，进一步加入 deep 部分，分别针对稀疏及稠密特征做针对性融合。FM 部分通过隐变量内积方式考虑一阶及二阶的特征融合，DNN 部分通过 Feed-Forward 学习高阶特征融合。模型训练过程中采取了 Learning Decay/Clip Gradient/ 求解器选择 /Dropout/ 激活函数选择等，在此不做赘述。

2.2 损失函数

在 ETA 预估场景下，准时率及置信度是比较重要的业务指标。初步尝试将 Square 的损失函数换成 Absolute 的损失函数，从直观上更为切合 MAE 相比 ME 更为严苛的约束。在适当 Learning Decay 下，结果收敛且稳定。

同时，在迭代中考虑到相同的 ETA 承诺时间下，在前后 N 分钟限制下，早到 1min 优于晚到 1min，损失函数的设计希望整体的预估结果能够尽量前倾。对于提前部分，适当降低数值惩罚。对于迟到部分，适当增大数值惩罚。进行多次调试设计后，最终确定以前后 N 分钟以及原点作为 3 个分段点。在原先 absolute 函数优化的基础上，在前段设计 1.2 倍斜率 absolute 函数，后段设计 1.8 倍斜率 absolute 函数，以便让结果整体往中心收敛，且预估结果更倾向于提前送达，对于 ETA 各项指标均有较大幅度提升。



损失函数

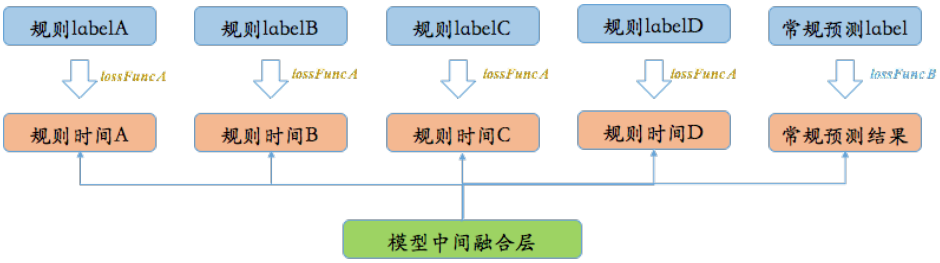
2.3 业务规则融入模型

目前的业务架构是”模型 + 规则”，在模型预估一个 ETA 值之后，针对特定业务场景，会有特定业务规则时间叠加以满足特定场景需求，各项规则由业务指标多次迭代产生。这里产生了模型和规则整体优化的割裂，在模型时间和规则时间分开优化后，即模型训练时并不能考虑到规则时间的影响，而规则时间在一年之中不同时间段，会产生不同的浮动，在经过一段时间重复迭代后，会加大割裂程度。

在尝试了不同方案后，最终将整体规则写入到了 TF 模型中，在 TF 模型内部调整整体规则参数。

- 对于简单的 (a*b+c)*d 等规则，可以将规则逻辑直接用 TF 的 OP 算子来实现，比如当 b、d 为定值时，则 a、c 为可学习的参数。
- 对于过于复杂的规则部分，则可以借助一定的模型结构，通过模型的拟合来代替，过多复杂 OP 算子嵌套并不容易同时优化。

通过调节不同的拟合部分及参数，将多个规则完全在 TF 模型中实现。最终对业务指标具备很大提升效果，且通过对部分定值参数的更改，具备部分人工干涉模型能力。



多目标补时结构

在这里，整体架构就简化为多目标预估的架构，这里采用多任务架构中常用的 Shared Parameters 的结构，训练时按比例采取不同的交替训练策略。结构上从最下面的模型中间融合层出发，分别在 TF 内实现常规预测结构及多个规则时间结构，而其对应的 Label 则仍然从常规的历史值和规则时间值中来，这样考虑了以下几点：

- 模型预估时，已充分考虑到规则对整体结果的影响（例如多个规则的叠加效应），作为整体一起考虑。
- 规则时间作为辅助 Label 传入模型，对于模型收敛及 Regularization，起到进一步作用。
- 针对不同的目标预估，采取不同的 Loss，方便进行针对性优化，进一步提升效果。

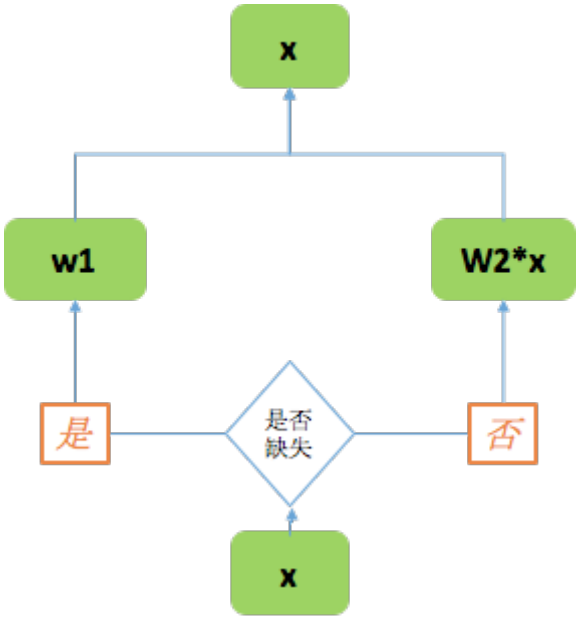
模型结构在进行预估目标调整尝试中：

- 尝试过固定共享网络部分及不固定共享部分参数，不固定共享参数效果明显。
- 通常情况下激活函数差异不大，但在共享层到独立目标层中，不同的激活函数差异很大。

2.4 缺失值处理

在模型处理中，特征层面不可避免存在一定的缺失值，而对于缺失值的处理，完全借鉴了《美团“猜你喜欢”深度学习排序模型实践》文章中的方法。对于特征 x 进入 TF 模型，进行判断，如果是缺失值，则设置 w1 参数，如果不是缺失值则进入模型数值为 w2*x，这里将 w1 和 w2 作为可学习参数，同时放入网络进行训练。以此

方法来代替均值 / 零值等作为缺失值的方法。



缺失值处理

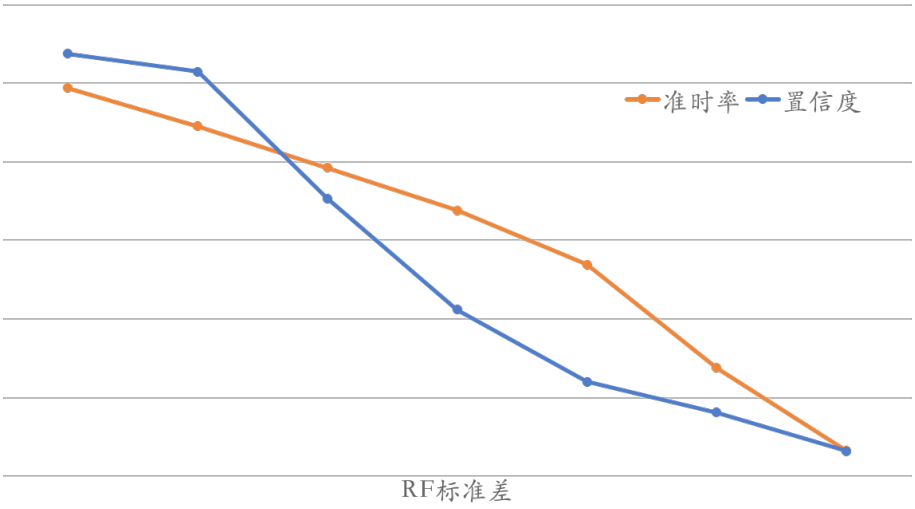
3. 长尾问题优化

3.1 模型预估结果 + 长尾规则补时

基础模型学习的是整体的统计分布，但对于一些长尾情形的学习并不充分，体现在长尾情形下预估时间偏短（由于 ETA 拥有考核骑手的功能，预估偏短对骑手而言意味着很大的伤害）。故将长尾拆解成两部分来分析：

- 业务长尾，即整体样本分布造成的长尾。主要体现在距离、价格等维度。距离越远，价格越高，实际送达时间越长，但样本占比越少，模型在这一部分上的表现整体都偏短。
- 模型长尾，即由于模型自身对预估值的不确定性造成的长尾。模型学习的是整体的统计分布，但不是对每个样本的预估都有“信心”。实践中采用 RF 多棵决策树输出的标准差来衡量不确定性。RF 模型生成的决策树是独立的，每棵

树都可以看成是一个专家，多个专家共同打分，打分的标准差实际上就衡量了专家们的“分歧”程度（以及对预估的“信心”程度）。从下图也可以看出来，随着 RF 标准差的增加，模型的置信度和准时率均在下降。



模型长尾因子

在上述拆解下，采用补时规则来解决长尾预估偏短的问题：长尾规则补时为 < 业务长尾因子，模型长尾因子 > 组合。其中业务长尾因子为距离、价格等业务因素，模型长尾因子为 RF 标准差。最终的 ETA 策略即为模型预估结果 + 长尾规则补时。

4. 工程开发实践

4.1 训练部分实践

整体训练流程

对于线下训练，采取如下训练流程：

Spark 原始数据整合 -> Spark 生成 TFRecord -> 数据并行训练 -> Tensor-Flow Serving 线下 GPU 评估 -> CPU Inference 线上预测

整个例行训练亿级数据多轮 Epoch 下流程持续约 4 小时，其中 TF 训练中，考虑到 TF 实际计算效率并不是很高，有很大比例在数据 IO 部分，通过 Spark 生成

TFRecord 部分，在此可将速度加速约 3.6 倍。而在数据并行训练部分，16 卡内的并行度扩展基本接近线性，具备良好的扩展性。由于 PS 上参数量并未达到单机无法承受，暂时未对参数在 PS 上进行切分。Serving 线下 GPU 评估部分，是整个流程中的非必需项，虽然在训练过程中 Chief Worker 设置 Valid 集合可有一定的指标，但对全量线下，通过 Spark 数据调用 Serving GPU 的评估具备短时间内完成全部流程能力，且可以指定大量复杂自定义指标。

数据并行训练方式

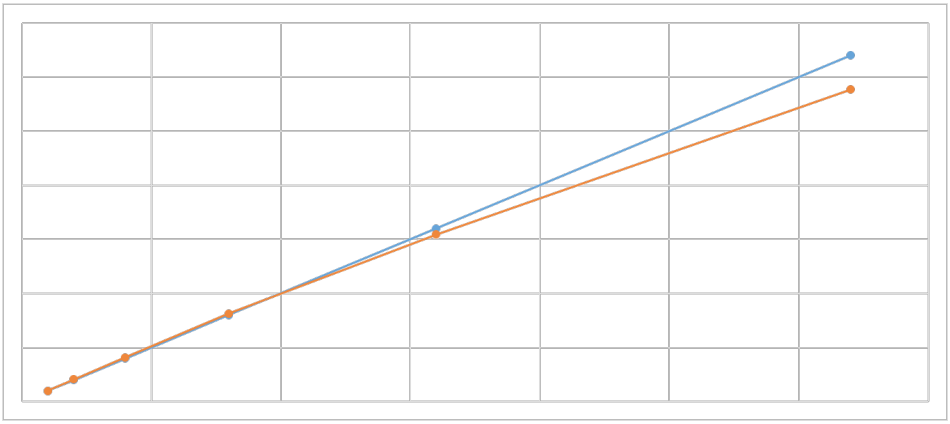
整个模型的训练在美团的 AFO 平台上进行，先后尝试分布式方案及单机多卡方案。考虑到生产及结果稳定性，目前线上模型生产采用单机多卡方案进行例行训练。

- 分布式方案：

采用 TF 自带的 PS-Worker 架构，异步数据并行方式，利用 tf.train.MonitoredTrainingSession 协调整个训练过程。整个模型参数存储于 PS，每个 Step 上每个 Worker 拉取数据进行数据并行计算，同时将梯度返回，完成一次更新。目前的模型单 Worker 吞吐 1~2W/s，亿级数据几轮 Epoch 耗时在几小时内完成。同时测试该模型在平台上的加速比，大约在 16 块内，计算能力随着 Worker 数目线性增加，16 卡后略微出现分离。在目前的业务实践中，基本上 4-6 块卡可以短时间内完成例行的训练任务。

- 单机多卡方案：

采用 PS-Worker 的方案在平台上具备不错的扩展性，但是也存在一定的弊端，使用 RPC 的通讯很容易受到其他任务的影响，整个的训练过程受到最慢 Worker 的影响，同时异步更新方式对结果也存在一定的波动。对此，在线上生产中，最终选取单机多卡的方案，牺牲一定的扩展性，带来整体训练效果和训练速度的稳定性。单机多卡方案采取多 GPU 手动指定 OP 的 Device，同时在各个 Device 内完成变量共享，最后综合 Loss 与梯度，将 Grad 更新到模型参数中。



加速比曲线

TF 模型集成预处理

模型训练过程中，ID 类特征低频过滤需要用到 Vocab 词表，连续型特征都需要进行归一化。这里会产生大量的预处理文件，在线下处理流程中很容易在 Spark 中处理成 Libsvm 格式，然后载入到模型中进行训练。但是在线上预测时，需要在工程开发端载入多个词表及连续型特征的归一化预处理文件（avg/std 值文件等），同时由于模型是按天更新，存在不同日期版本的对齐问题。

为了简化工程开发中的难度，在模型训练时，考虑将所有的预处理文件写入 TF 计算图之中，每次在线预测只要输入最原始的特征，不经过工程预处理，直接可得到结果：

- 对于 ID 类特征，需要进行低频过滤，然后制作成词表，TF 模型读入词表的 list_arr，每次 inference 通过 ph_vals，得到对应词表的 ph_idx。

```
tf_look_up = tf.constant(list_arr, dtype=tf.int64)
table = tf.contrib.lookup.HashTable(tf.contrib.lookup.
KeyValueTensorInitializer(tf_look_up, idx_range), 0)
ph_idx = table.lookup(ph_vals) + idx_bias
```

对于连续型特征，在 Spark 处理完得到 avg/std 值后，直接写入 TF 模型计算图中，作为 constant 节点，每个 ph_in 经过两个节点，得到相应 ph_out。

```
constant_avg = tf.constant(feat_avg, dtype=tf.float32, shape=[feat_dim],
name="avg")
constant_std = tf.constant(feat_std, dtype=tf.float32, shape=[feat_dim],
name="std")
ph_out = (ph_in - constant_avg) / constant_std
```

4.2 TF 模型线上预测

配送机器学习平台内置了模型管理平台，对算法训练产出的模型进行统一管理和调度，管理线上模型所用的版本，并支持模型版本的切换和回退，同时也支持节点模型版本状态的管理。

ETA 使用的 DeepFM 模型用 TensorFlow 训练，生成 SavedModel 格式的模型，需要模型管理平台支持 Tensorflow SavedModel 格式。

实现方案 S 线上服务加载 TensorFlow SavedModel 模型有多种实现方案：

- 自行搭建 TensorFlow Serving CPU 服务，通过 gRPC API 或 RESTful API 提供服务，该方案实现比较简单，但无法与现有的基于 Thrift 的模型管理平台兼容。
- 使用美团 AFO GPU 平台提供的 TensorFlow Serving 服务。
- 在模型管理平台中通过 JNI 调用 TensorFlow 提供的 Java API TensorFlow Java API，完成模型管理平台对 SavedModel 格式的支持。

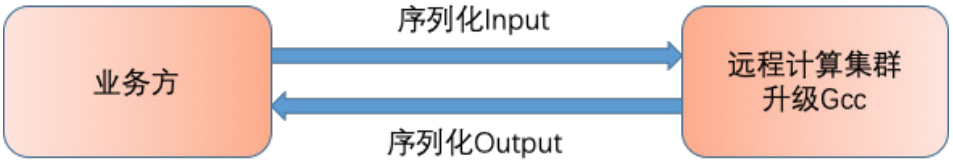
最终采用 TensorFlow Java API 加载 SavedModel 在 CPU 上做预测，测试 batch=1 时预测时间在 1ms 以内，选择方案 3 作为实现方案。

远程计算模式

TensorFlow Java API 的底层 C++ 动态链接库对 libstdc++.so 的版本有要求，需要 GCC 版本不低于 4.8.3，而目前线上服务的 CPU 机器大部分系统为 CentOS 6, 默认自带 GCC 版本为 4.4.7。如果每台线上业务方服务器都支持 TensorFlow SavedModel 本地计算的话，需要把几千台服务器统一升级 GCC 版本，工作量比较大而且可能会产生其他风险。

因此，我们重新申请了几十台远程计算服务器，业务方服务器只需要把 Input 数

据序列化后传给 TensorFlow Remote 集群，Remote 集群计算完后再将 Output 序列化后返回给业务方。这样只需要对几十台计算服务器升级就可以了。



在线序列化

线上性能

模型上线后，支持了多个业务方的算法需求，远程集群计算时间的 TP99 基本上在 5ms 以内，可以满足业务方的计算需求。

线上效果

总结与展望

模型落地并上线后，对业务指标带来较大的提升。后续将会进一步根据业务优化模型，进一步提升效果：

- 将会进一步丰富多目标学习框架，将取餐、送餐、交付、调度等整个配送生命周期内的过程在模型层面考虑，对订单生命周期内多个目标进行建模，同时提升模型可解释性。
- 模型融合特征层面的进一步升级，在 Embedding 以外，通过更多的 LSTM/ CNN/ 自设计结构对特征进行更好的融合。
- 特征层面的进一步丰富。

作者简介

基泽，美团点评技术专家，目前负责配送算法策略部机器学习组策略迭代工作。
周越，2017 年加入美团配送事业部算法策略组，主要负责 ETA 策略开发。
显杰，美团点评技术专家，2018 年加入美团，目前主要负责配送算法数据平台深度学习相关的研发工作。

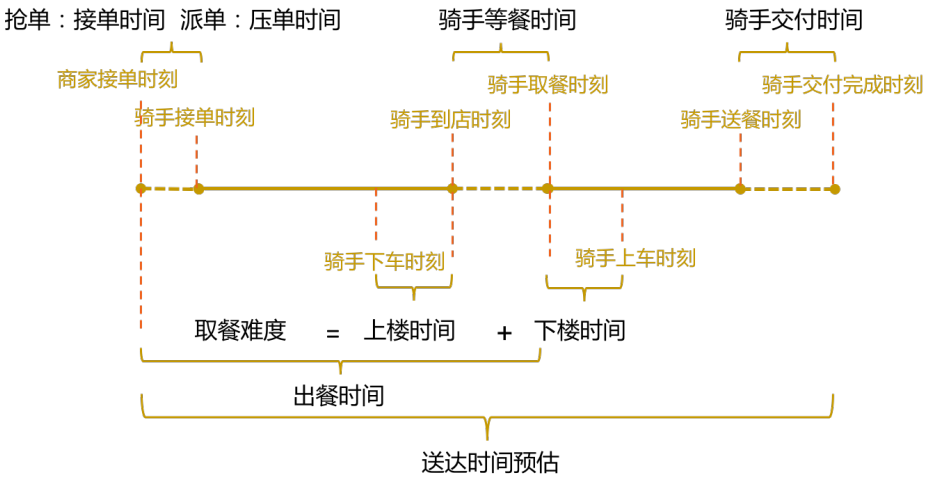
配送交付时间轻量级预估实践

基泽　闫聪

1. 背景

可能很多同学都不知道，从打开美团 App 点一份外卖开始，然后在半小时内就可以从骑手小哥手中拿到温热的饭菜，这中间涉及的环节有多么复杂。而美团配送技术团队的核心任务，就是将每天来自祖国各地的数千万份订单，迅速调度几十万骑手小哥按照最优路线，并以最快的速度送到大家手中。

在这种场景下，骑手的交付时间，即骑手到达用户附近下车后多久能送到用户手中　，就是一个非常重要的环节。下图是一个订单在整个配送链路的时间构成，时间轴最右部分描述了交付环节在整个配送环节中的位置。交付时间衡量的是骑手送餐时的交付难度，包括从骑手到达用户楼宇附近，到将餐品交付到用户手中的整个时间。



交付时间的衡量是非常有挑战的一件事，因为骑手在送餐交付到用户手中时会碰到不同的问题，例如：骑手一次送餐给楼宇内多个用户，骑手对于特定楼宇寻址特别困难，骑手在交付楼宇附近只能步行，老旧小区没有电梯，写字楼无法上楼，或者难

以等到电梯等等。交付时间预估需要具备刻画交付难度的能力，在定价、调度等多个场景中被广泛使用。例如根据交付难度来确定是否调节骑手邮资，根据交付难度来确定是否调节配送运单的顺序，从而避免超时等等。总的来说，交付时间预估是配送业务基础服务的重要一环。

但是，交付时间预估存在如下的困难：

- 输入信息较少，且多为非数值型数据，目前能够被用来预估的仅有如下维度特征：交付地址、交付点的经纬度、区域、城市，适配常规机器学习模型需要重新整理且容易丢失信息。
- 计算性能要求很高。由于是基础服务，会被大量的服务调用，需要性能 TP99 保证在 10ms 以内，整个算法平均响应时间需要控制在 5ms 内，其中包括数据处理及 RPC 的时间。且该标准为 CPU 环境下的性能要求，而非 GPU 下的性能要求。

	Name	Total	Failure	Failure%	Log View	Max	Avg	90Line	95Line	99Line	99.9Line
[:: show ::]	TOTAL	1,225,797	0	0.0000%	L S	85.0	3.8	0.0	0.0	0.0	0.0
[:: show ::]	V3:1	595,184	0	0.0000%	L S	80.0	4.3	8.1	8.8	10.1	23.7
[:: show ::]	V3P2:1	397,471	0	0.0000%	L S	85.0	3.7	5.0	5.1	6.0	18.1
[:: show ::]	V2:1	233,142	0	0.0000%	L S	45.0	2.5	3.4	3.7	4.0	6.6

上图为部分版本所对应的性能，平响时间均在 5ms 内，TP99 基本在 10ms 内

总结起来，交付时间预估的问题，在于需要使用轻量级的解决方案来处理多种数据形式的非数值型数据，并提取有效信息量，得到相对准确的结果。在相同效果的前提下，我们更倾向于性能更优的方案。

在本文中，我们介绍了交付时间预估迭代的三个版本，分别为基于地址结构的树模型、向量召回方案以及轻量级的 End-to-End 的深度学习网络。同时介绍了如何在性能和指标之间取舍，以及模型策略迭代的中间历程，希望能给从事相关工作的同学们有所启发和帮助。

2. 技术迭代路径

首先，在交付时间预估的技术迭代上，我们主要经历了三个大版本的改动，每一

版本在 5ms 计算性能的约束下，追求轻量化的解决方案，在兼顾提升效果的基础上，不显著增加性能的消耗。

本章节分别叙述了 3 个模型的迭代路径，包括技术选型、关键方案及最终效果。

2.1 树模型

技术选型

最早也是最容易被考虑到的是利用规则，核心思路是利用树结构衡量地址相似性，尽可能在相似的交付地址上积聚结构化数据，然后利用局部的回归策略，得到相对充裕的回归逻辑，而未能达到回归策略要求的则走兜底的策略。

为了快速聚积局部数据，树模型是一个较为合适的解决方案，树的规则解析能够有效地聚集数据，同时一个层级并不深的树，在计算速度上，具备足够的优势，能够在较短的时间内，得到相对不错的解决方案。

观察用户填写地址以及联系实际中地址的层级结构，不难发现，一个地址可以由四级结构组成：地址主干词（addr）、楼宇号（building）、单元号（unit）、楼层（floor）。其中的地址主干词在实际中可能对应于小区名或者学校名等地标名称。例如望京花园 1 号楼 2 单元 5 楼，解析为（望京花园，1 号楼，2 单元，5 楼）。通过分析，实际交付时长与楼层高低呈正相关关系，且不同交付地址的交付时长随楼层增加的变化幅度也有所区别，所以可以使用线性回归模型拟合楼层信息和交付时长的关系，而地址主干词、楼宇号、单元号作为其层级索引。但用户填写的地址中并不一定包含完整的四级结构，就会存在一定比例的缺失，所以利用这样的层级结构构建成一棵树，然后充分利用上一层已知的信息进行预估。预测时，只需根据结点的分支找到对应的模型即可，如果缺失，使用上一层结构进行预测。对于没有达到训练模型要求数据量的地址，使用其所在的区域平均交付时长作为交付时长的预估结果，这部分也可以看作区域信息，作为树结构的根节点。

迭代路径

整体的思路是基于离散特征训练树模型，在树的结点上基于楼层训练线性回归模型。树结点训练分裂规则：（1）数据量大于阈值；（2）分裂后 MAE（平均绝对误差）的

和小于分裂前。考虑到数据的时效性，采用加权线性回归增加近期数据的权重。

2.2 树模型 + 向量召回方案

技术选型

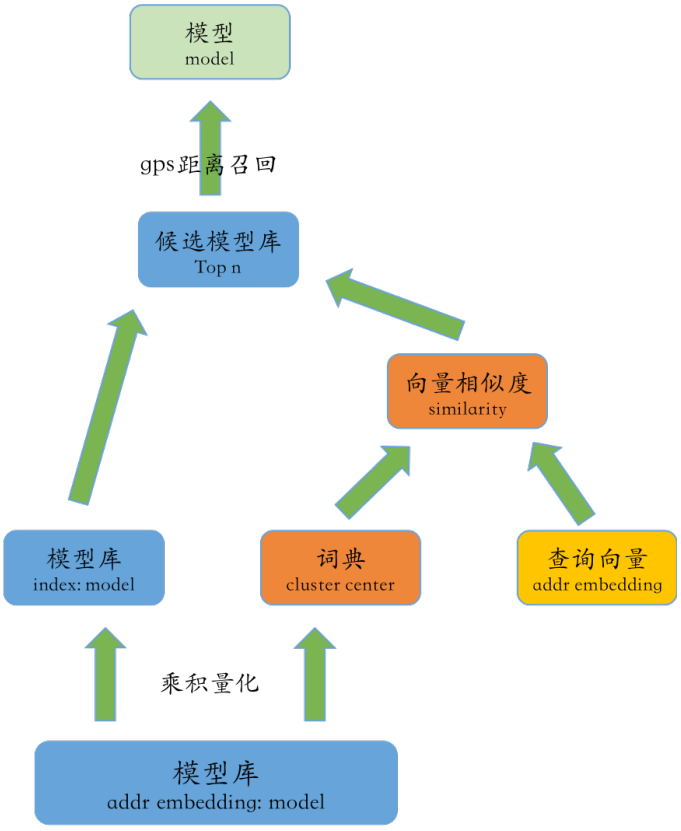
向量召回作为主流的召回方案之一，被业界广泛使用，在使用 LSH、PQ 乘积量化等常用开源工具基础上，高维向量召回性能通常在毫秒量级。

而从算法上考虑，树模型中 NLP 地址解析结果能够达到模型使用要求的仅为 70%+，剩余 20%+ 的地址无法通过训练得到的模型从而只能走降级策略。利用高维向量来表达语义相似性，即利用向量来表达地址相似性，从而用相似数据对应的模型来替代相似但未被召回数据，将地址主干词进行 Embedding 后，摆脱主干词完全匹配的低鲁棒性。

例如，在地址上可能会出现【7 天酒店晋阳街店】数据量比较充足，但【7 天连锁酒店太原高新区晋阳街店】数据量不充足从而无法训练模型的案例，这可能是同一个交付位置。我们希望尽可能扩大地址解析的成功率。

迭代路径

整个技术路径较为清晰简单，即利用 Word2Vec 将 charLevel 字符进行 Embedding，获得该地址的向量表示，并且融入 GPS 位置信息，设计相应兜底策略。



向量召回方案决策路径

最终效果

比较大地提升了整体策略的召回率，提升了 12.20pp，对于未被上一版本树模型召回的地址，指标有了显著的提升，其中 ME 下降 87.14s，MAE 下降 38.13s，1min 绝对偏差率减小 14.01pp，2min 绝对偏差率减小 18.45pp，3min 绝对偏差率减小 15.90pp。

2.3 End-to-End 轻量化深度学习方案

技术选型

在树模型的基础上，迭代到向量召回方案，整个模型的召回率有了较大幅度的增长，但仍然不是 100%。分析发现，召回率提升的障碍在于 NLP 对于地址解析的覆盖率。

整个方案的出发点：

从模型复杂度考虑，同样仅仅使用地址信息的话，在提升模型 VC 维的基础上，使用其他的模型方案至少可以持平树模型的效果，如果在这基础上还能融入其他信息，那么对于原模型的基线，还能有进一步的提升。

考虑到不仅仅需要使用地址数据，同时需要使用 GPS 数据、大量 ID 类的 Embedding，对于各类非数值类型的处理灵活性考虑，采用深度学习的方案，来保证多源且多类型特征能在同一个优化体系下优化学习。

工程上需要考虑的点：

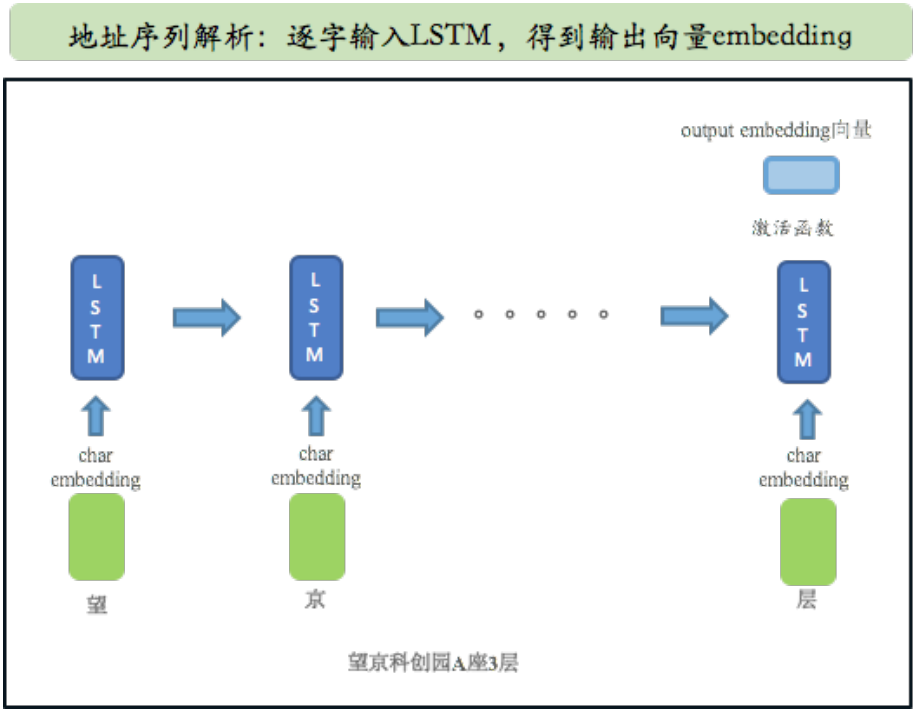
交付模型作为基础模型，被广泛应用在路径构造、定价、ETA 等各个业务中频繁调用，在树模型版本中，对于性能的要求为平均响应时间 5ms，TP99 在 10ms 左右，本方案需要考虑沿袭原业务的性能，不能显著增加计算耗时。

交付模型的难点在于非数值型特征多，信息获取形式的多样化，当前的瓶颈并不在于模型的复杂度低。如果可以轻量地获取信息及融合，没必要对 Fusion 后的信息做较重的处理方案。

所以整体的设计思路为：利用深度学习融合非数值型特征，在简单 Fusion 的基础上，直接得到输出结构，对于组件的选择，尽可能选用 Flops 较低的设计。该设计背后意图是，在充分使用原始输入信息，在尽可能避免信息损失的基础上，将非数值型的信息融入进去。并将信息充分融合，直接对接所需要的目标。而选用的融合组件结构尽可能保证高性能，且具备较高学习效率。这里分别针对地址选用了较为 Robust 的 LSTM，针对 GPS 选用了自定义的双线性 Embedding，兼顾性能和效果。

迭代路径

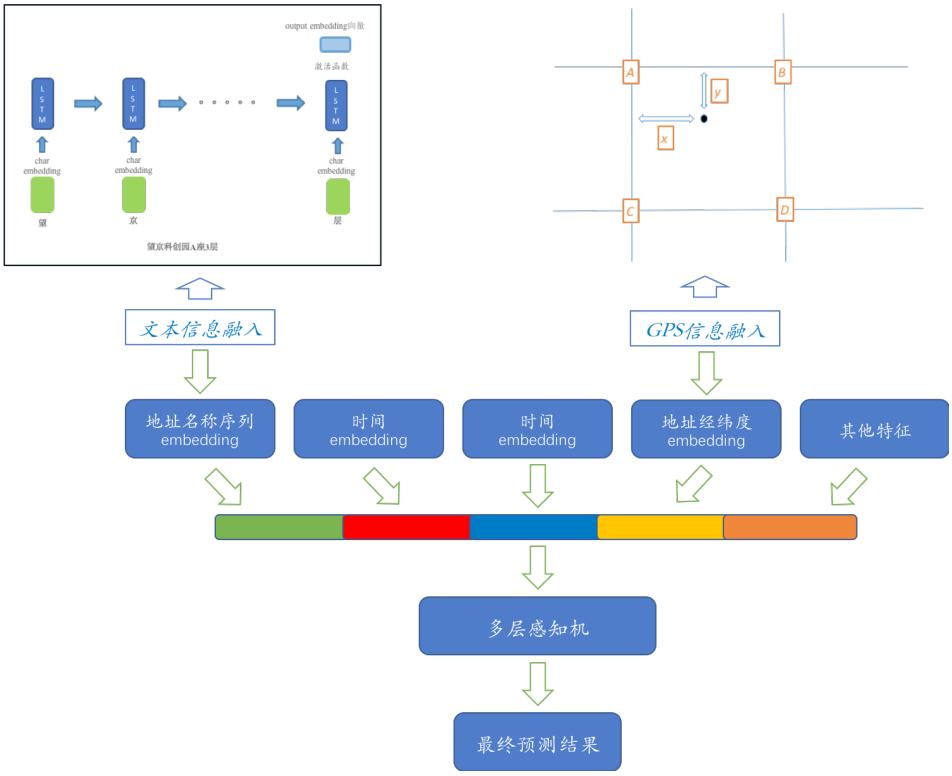
开始采用端到端的深度学习模型，这里首先需要解决的是覆盖率问题，直接采用 LSTM 读取 charLevel 的地址数据，经过全连接层直接输出交付时间。作为第一版本的数据，该版本数据基本持平树模型效果，但对于树模型未召回的 20% 数据，有了较大的提升。



地址信息输入 charLevel 模型

在采用 charLevel 的地址奏效后，我们开始采用加入用户地址 GPS 的信息，由于 GPS 为经纬度信息，非数值型数据，我们使用一种基于地理位置格点的双线性插值方法进行 Embedding。该方案具备一定的扩展性，对不同的 GPS 均能合理得到 Embedding 向量，同时具备平滑特性，对于多对偏移较小的 GPS 点能够很好的进行支持。

最终方案将地址 Embedding 后，以及 GPS 点的 Embedding 化后，加入下单时间、城市 ID、区域 ID 等特征后，再进行特征融合及变换，得到交付模型的时间预估输出。整个模型是一个端到端的训练，所有参数均为 Trainable。



模型结构示意图

扩展组件

在证实 End-to-End 路径可行后，我们开始进行扩展组件建设，包括自定义损失函数、数据采样修正、全国模型统一等操作，得到一系列正向效果，并开发上线。

特征重要性分析

对于深度学习模型，我们有一系列特征重要性评估方案，这里采用依次进行 Feature Permutation 的方式，作为评估模型特征重要性的方式。

考虑 GPS 经纬度和用户地址存在较大程度的信息重叠，评估结果如下。Shuffle 后，用户地址的特征重要性高于 GPS 经纬度的特征重要性。加入 GPS 后 ME 下降不如地址信息明显，主要是地址信息包含一定冗余信息（下文会分析），而其他信息的影响则可以忽略不计。

特征	特征重要排名
用户地址	1
GPS 经纬度	2
其他特征	……

注：在配送的其他案例中，商户 GPS 的经纬度重要性 >> 用户地址重要性 >> 用户 GPS 的经纬度重要性，该特征重要性仅仅为本案例特征重要性排序，不同学习目标下可能会有比较明显差别。

最终效果

End-to-End 深度学习模型的最终效果较为显著：对于树模型及向量召回方案的最痛点，覆盖率得到彻底解决，覆盖率提升到 100%。ME 下降 4.96s，MAE 下降 8.17s，1min 绝对偏差率减小 2.38pp，2min 绝对偏差率减小 5.08pp，3min 绝对偏差率减小 3.46pp。同时，对于之前树模型及向量召回方案未能覆盖到的运单，提升则更为明显。

3. 模型相关分析

在整个技术迭代的过程中，由于整个解决方案对于性能有着较为苛刻的要求，需要单独对方案性能进行分析。本章节对向量召回方案及深度学习方案进行了相应的性能分析，以便在线下确认性能指标，最终保证上线后性能均达到要求。下文分别着重介绍了向量匹配的工具 Faiss 以及 TensorFlow Operation 算子的选取，还有对于整体性能的影响。

同时对比 End-to-End 生成向量与 Word2Vec 生成向量的质量区别，对于相关项目具备一定的借鉴意义。

3.1 向量召回性能

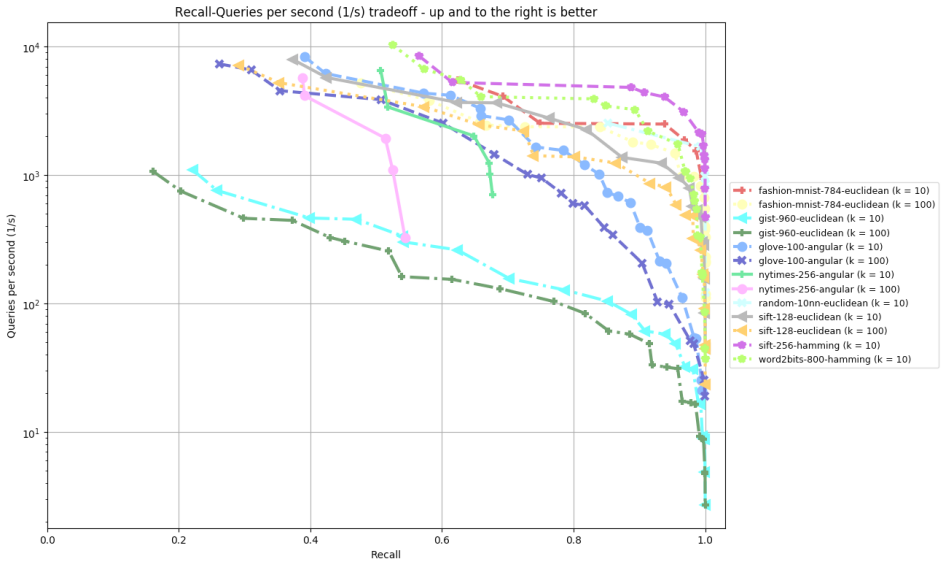
最近邻搜索（Nearest Neighbor Search）指的是在高维度空间内找到与查询点最近点的问题。在数据样本小的时候，通过线性搜索就能满足需求，但随着数据量的增加，如达到上百万、上亿点时候，倾向于将数据结构化表示来更加精确地表达向量信息。

此时近似最近邻搜索 ANN（Approximate Nearest Neighbor）是一个可参考的技术，它能在近似召回一部分之后，再进行线性搜索，平衡效率和精度。目前大体上有以下 3 类主流方法：基于树的方法，如 K-D 树等；基于哈希的方法，例如 LSH；基于矢量量化的方法，例如 PQ 乘积量化。在工业检索系统中，乘积量化是使用较多的一种索引方法。

针对向量召回的工具，存在大量的开源实现，在技术选型的过程中，我们参照 ANN-Benchmarks 以及 Erikbern/ANN-Benchmarks 中的性能评测结果。在众多 ANN 相关的工具包内，考虑到性能、内存、召回精度等因素，同时可以支持 GPU，在向量召回方案的测试中，选择以 Faiss 作为 Benchmark。

Faiss 是 FaceBook 在 2017 年开源的一个用于稠密向量高效相似性搜索和密集向量聚类的库，能够在给定内存使用下，在速度和精度之间权衡。可以在提供多种检索方式的同时，具备 C++/Python 等多个接口，也对大部分算法支持 GPU 实现。

下图为 Faiss 测评曲线：



交付时间模型召回的性能测试如下，可以达到性能需求。

- 召回候选集数量：8W 条向量【由于采用了 GPS 距离作为距离限制，故召回

测试采用 8W 数量级】

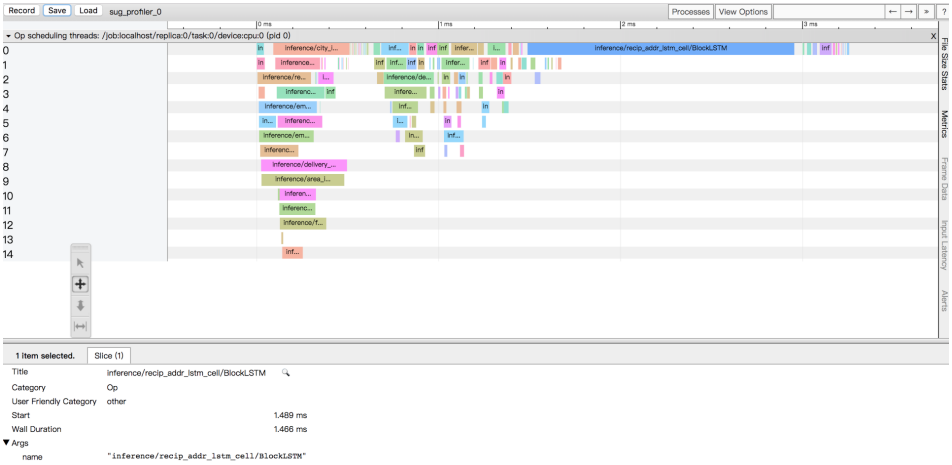
- 测试机器：Mac 本机 CPU【CPU 已满足性能，故不再测试 GPU】

单位 (ms)	最近邻数量	mean	90分位数	max	建索引
IndexFlatL2 dimention=128	1	2.63	2.73	10.83	12.80
	100	2.66	2.69	63.82	
	1000	3.14	3.27	21.54	
IndexIVFFlat L2距离 nlist=100 dimention=128 nprobe=10	1	0.65	0.67	5.59	234.52
	100	0.70	0.72	5.72	
	1000	1.06	1.08	11.95	
IndexIVFFlat Cosine距离 nlist=100 dimention=128 nprobe=10	1	0.36	0.38	5.47	153.99
	100	0.39	0.40	14.41	
	1000	0.61	0.63	5.99	
IndexFlatIP dimention=128	1	2.37	2.20	9.45	17.53
	100	2.92	3.08	17.59	
	1000	3.25	3.29	28.12	

3.2 序列模块性能

在 TensorFlow 系统中，以 C API 为界限，将系统划分为【前端】和【后端】两个子系统，前端扮演 Client 角色，完成计算图的构造，然后由 Protobuf 发送给后端启动计算图计算。计算图的基础单元是 OP，代表的是某种操作的抽象。在 TensorFlow 中，考虑到实现的不同，不同 OP 算子的选择，对于计算性能具有较大影响。

为了评测深度学习交付模型的性能瓶颈，首先对整个模型进行 Profile，下图即为 Profile 后的 Timeline，其中整个计算大部分消耗在序列模块处理部分，即下图中的蓝色部分。故需要对序列模块的计算性能进行 OP 算子的加速。



考虑到序列处理的需求，评估使用了 LSTM/GRU/SRU 等模块，同时在 TensorFlow 中，LSTM 也存在多种实现形式，包括 BasicLSTMCell、LSTMCell、LSTMBlockCell、LSTMBlockFusedCell 和 CuDNNLSTM 等实现，由于整个交付模型运行在 CPU 上，故排除 CuDNNLSTM，同时设置了全连接层 FullyConnect 加入评估。

从评估中可以发现，全连接层速度最快，但是对于序列处理会损失 2.3pp 效果，其余的序列模型效果差异不大，但不同的 OP 实现对结果影响较大。原生的 BasicLSTM 性能较差，contrib 下的 LSTMBlockFusedCell 性能最好，GRU/SRU 在该场景下未取得显著优势。

这是 LSTMBlockFusedCell 的官方说明，其核心实现是将 LSTM 的 Loop 合并为一个 OP，调用时候整个 Timeline 上更为紧凑，同时节约时间和内存：

> This is an extremely efficient LSTM implementation, that uses a single TF op for the entire LSTM. It should be both faster and more memory-efficient than LSTMBlockCell defined above.

以下是序列模块的性能测试：

- 环境：Tensorflow1.10.0，CentOS 7。
- 测试方法：CPU inference 1000 次，取最长的地址序列，求平均时间。

- 结论：LSTMBlockFused 实现性能最佳。【FullyConnect 性能最快，但对性能有损失】

注：在评估中，不仅仅包括了序列模型，也包括了其他功能模块，故参数量及模型大小按照总体模型而言

lstm结构OP	时间(ms)	FLOPs	可训练参数量	模型大小(MB)	效果差异
Fully Connect	1.18	27.83M	7.00M	29.1	-2.3pp
SRU	4.00	27.96M	7.06M	29.4	差异不显著
GRU Block	3.64	28.02M	7.10M	29.6	差异不显著
GRU	4.44	28.02M	7.10M	29.6	差异不显著
LSTMBlockFused	2.48	28.09M	7.13M	29.7	差异不显著
LSTM Block	4.34	28.09M	7.13M	29.7	差异不显著
LSTM	4.85	28.09M	7.13M	29.7	差异不显著
BasicLSTM	4.92	28.09M	7.13M	29.7	差异不显著

3.3 向量效果分析

将向量召回与深度学习模型进行横向比较，二者中间过程均生成了高维向量。不难发现，二者具备一定的相似性，这里就引发了我们的思考：

相较于向量召回，深度学习模型带来的提升主要来自于哪里?

有监督的 lstm 学习到的 Embedding 向量与自监督的 Word2Vec 得到的向量在地址相似性计算中有多大差别，孰优孰劣?

首先，我们分析第一个问题，End-to-End 模型提升主要来自哪里?

ME	MAE	1min绝对偏差率	2min绝对偏差率	3min绝对偏差率
End-to-End 模型 - Word2Vec 模型	4.14	-0.45	-0.31%	0.05%

我们直接将 End-to-End 模型得到的 char embedding 抽取出来，直接放入到 Word2Vec 方案内，取代 Word2Vec 生成的 char embedding，再进行向量召回的评估。结果如下表所示，单独抽取出来的 char embedding 在向量召回方案中，表现与 Word2Vec 生成的向量基本一致，并没有明显的优势。

注：

- 1min 绝对偏差率定义：|pred−label|<=60s
- 2min 绝对偏差率定义：|pred−label|<=120s
- 3min 绝对偏差率定义：|pred−label|<=180s

此时的变量有 2 个方面：

- a）对于 charLevel 地址的学习结构不同，一个为 Word2Vec，一个为 LSTM
- b）输入信息的不同，Word2Vec 的信息输入仅仅为地址主干词，而 End−to−End 的信息输入则包括了地址主干词、地址附属信息、GPS 等其他信息。

注：

- 完整地址：卓玛护肤造型（洞庭湖店）（洞庭湖路与天山路交叉路口卓玛护肤造型）
- 地址主干词：卓玛护肤造型店
- 地址附属信息：（洞庭湖店）（洞庭湖路与天山路交叉路口卓玛护肤造型）

为了排除第二方面的因素，即 b 的因素，使用地址主干词作为输入，而不用地址附属信息和其他模型结构的输入，保持模型输入跟 Word2Vec 一致。在测试集上，模型的效果比完整地址有明显的下降，MAE 增大约 15s。同时将 char embedding 提取出来，取代 Word2Vec 方案的 char embedding，效果反而变差了。结合 2.3 节中的特征重要性，可知，深度学习模型带来的提升主要来自对地址中冗余信息（相较于向量召回）的利用，其次是多个新特征的加入。另外，对比两个 End−to−End 模型的效果，地址附属信息中也包含着对匹配地址有用的信息。

ME	MAE	1min绝对偏差率	2min绝对偏差率	3min绝对偏差率
End−to−End 模型 − Word2Vec 模型	−1.28	0.64	0.90%	0.85%

针对第二个问题，有监督的 End−to−End 学习到的 Embedding 向量，与自监督的 Word2Vec 得到的向量在地址相似性计算中有多大差别，孰优孰劣?

采用地址主干词代替完整地址，作为 End−to−End 模型的输入进行训练，其他

信息均保持不变。使用地址主干词训练得到的 Embedding 向量，套用到向量召回方案中。

从评估结果来看，对于不同的阈值，End-to-End 的表现差异相对 Word2Vec 较小。相同阈值下，End-to-End 召回率更高，但是效果不如 Word2Vec。

从相似计算结果看，End-to-End 模型会把一些语义不相关但是交付时间相近的地址，映射到同一个向量空间，而 Word2Vec 则是学习一个更通用的文本向量表示。

> 例如，以下两个交付地址会被认为向量距离相近，但事实上只是交付时间相近：南内环西街与西苑南路交叉口金昌盛国会 <=> 辰憬家园迎泽西大街西苑南路路口林香斋酒店

如果想要针对更为复杂的目标和引入更多信息，可以使用 End-to-End 框架；只是计算文本相似性，从实验结果看，Word2Vec 更好一些。同时，通过查看 Case 也可以发现，End-to-End 更关注结果相似性，从而召回一部分语义上完全不相关的向量。两个模型目标上的不同，从而导致了结果的差异。

4. 总结与展望

在本篇中，依次展示了在配送交付场景下的三次模型策略迭代过程，以及在较为苛刻性能要求限制下，如何用轻量化的方案不断提高召回率及效果。同时，对迭代过程中的性能进行简单的分析及衡量，这对相关的项目也具备一定的借鉴意义，最后对 Word2Vec 及 End-to-End 生成的向量进行了比较。

事实上，本文中提及的向量召回及深度学习融合非数值型特征的方案，已经在业界被广泛使用。但对于差异化的场景，本文仍具备一定的借鉴价值，特别是对于订单 – 骑手匹配、订单 – 订单匹配等非搜索推荐领域的场景化应用，以及 TF OP 算子的选用及分析、Embedding 生成方式带来的差异，希望能够给大家提供一些思路和启发。

5. 关联阅读

交付时间预估与 ETA 预估及配送其他业务关系：

- 交付时间预估是 ETA 预估中的重要一环，关于 ETA 预估，请参见《深度学习在美团配送 ETA 预估中的探索与实践》。
- 具体 ETA 在整个配送业务中的位置及配送业务的整体机器学习实践，请参看《机器学习在美团配送系统的实践：用技术还原真实世界》。

6. 作者简介

基泽，美团点评技术专家
闫聪，美团点评算法工程师

7. 招聘信息

美团配送 AI 团队支撑全球领先的即时配送网络——美团配送，并建立了行业领先的美团智能配送系统。美团配送 AI 团队主要来自一线互联网公司以及知名科研机构，研发实力和氛围卓越。目前美团配送业务仍处于快速发展期，新的场景、新的技术难题不断涌现，成长空间巨大。
美团配送 AI 团队现诚聘算法策略工程师、机器学习工程师和运筹优化工程师，欢迎有兴趣的小伙伴投递简历至：tech@meituan.com（邮件标题注明：美团配送 AI 团队）

ICDAR 2019 论文：自然场景文字定位技术详解

刘曦

自然场景文字定位是文字识别中非常重要的一部分。与通用的物体检测相比，文字定位更具挑战性，文字在长宽比、尺度和方向上有更大范围的变化。针对这些问题，本文介绍一种融合文字片段及金字塔网络的场景文字定位方法。该方法将特征金字塔机制应用到单步多框检测器以处理不同尺度文字，同时检测多个文字片段以及学习出文字片段之间 8-neighbor 连接关系，最后通过 8-neighbor 连接关系将文字片段连接起来，实现对不同方向和长宽比的文字定位。此外，针对文字通常较小特点，扩大检测网络中 backbone 模型深层特征图，以获得更好性能。

本文提出的方法已发表在文档分析与识别国际会议 ICDAR2019 (International Conference on Document Analysis and Recognition) 上，审稿人评论该方法为“As it is of more practical uses”，认可了它的实用性。

ICDAR 是由国际模式识别学会 (IAPR) 组织的专业会议之一，专注于文本领域的识别与应用。ICDAR 大会每两年举办一次，目前已发展成文字识别领域的旗舰学术会议。为了提高自然场景的文本检测和识别水平，国际文档分析和识别会议（ICDAR）于 2003 年设立了鲁棒文本阅读竞赛（“Robust Reading Competitions”）。至今已有来自 89 个国家的 3500 多支队伍参与。ICDAR 2019 将于今年 9 月 20-25 日在澳大利亚悉尼举办。 美团今年联合国内外知名科研机构和学者，提出了”中文门脸招牌文字识别”比赛（ICDAR 2019 Robust Reading Challenge onReading Chinese Text on Signboards）。

背景

自然场景图像中的文字识别已被广泛应用在现实生活中，例如拍照翻译，自动驾驶，图像检索和增强现实等，因此也有越来越多的专家学者对其进行研究。自然场景文字定位是指对场景图像中所有文本的精确定位，是自然场景文字识别中第一步也是最重要的一步。由于自然场景下文本颜色、大小、宽高比、字体、方向、光照条件和

背景等具有较大变化（如图 1），因此它是非常具有挑战性的。

图 1　自然场景文字图片

深度学习技术在物体识别和检测等计算机视觉任务方面已经取得了很大进展。许多最先进的基于卷积神经网络（CNN）的目标检测框架，如 Faster RCNN、SSD 和 FPN[1] 等，已被用来解决文本检测问题并且性能远超传统方法。

深度卷积神经网络是一个多层级网络结构，浅层特征图具有高分辨率及小感受野，深层特征图具有低分辨率及大感受野。具有小感受野的浅层特征点对于小目标比较敏感，适合于小目标检测，但是浅层特征具有较少的语义信息，与深层特征相比具有较弱的辨别力，导致小文本定位的性能较差。另一方面，场景文字总是具有夸张的长宽比（例如一个很长的英文单词或者一条中文长句）以及旋转角度（例如基于美学考虑），通用物体检测框架如 Faster RCNN 和 SSD 是无法回归较大长宽比的矩形和旋转矩形。

围绕上面描述的两个问题，本文主要做了以下事情：

1. 为了处理不同尺度的文本，借鉴特征金字塔网络思路，将具有较强判别能力的深层特征与浅层特征相结合，实现在各个层面都具有丰富语义的特征金字塔。另外，当较深层中的小对象丢失时，特征金字塔网络仍可能无法检测到小对象，深层的上下文信息无法增强浅层特征。我们额外扩大了深层的特征图，以更准确地识别小文本。
2. 我们不直接回归文本行，而是将文本行分解为较小的局部可检测的文字片段，并通过深度卷积网络进行学习，最后将所有文字片段连接起来生成最终的文本行。

现有方法

最新的基于深度神经网络的文本定位算法大致可以分为两大类：（1）基于分割的文本定位；（2）基于回归的文本定位。

（1）基于分割的文本定位

当前基于分割的文本定位方法大都受到完全卷积网络（FCN [2]）的启发。全卷积网络（FCN, fully convolutional network），是去除了全连接 (fc) 层的基础网络，最初是用于实现语义分割任务。由于 FCN 网络最后一层特征图的像素分辨率较高，而图文识别任务中需要依赖清晰的文字笔画来区分不同字符（特别是汉字），所以 FCN 网络很适合用来提取文本特征。当 FCN 被用于图文识别任务时，最后一层特征图中每个像素将被分成文字行（前景）和非文字行（背景）两个类别。

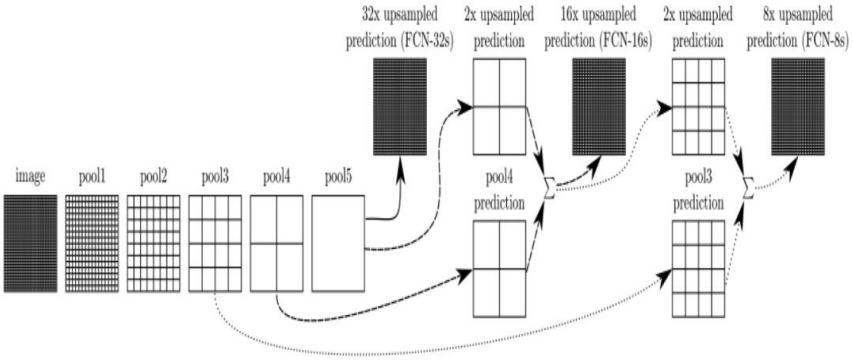


图 2　全卷积网络

（2）基于回归的文本定位

Textboxes [3] 是经典的也是最常用的基于回归的文本定位方法，它基于 SSD 框架，训练方式是端到端，运行速度也较快。为了适应文本行细长型特点，特征层也用长条形卷积核代替了其他模型中常见的正方形卷积核。为了防止漏检文本行，还在垂直方向增加了候选框数量。为了检测大小不同的字符块，在多个尺度的特征图上并行预测文本框，然后对预测结果做 NMS 过滤。

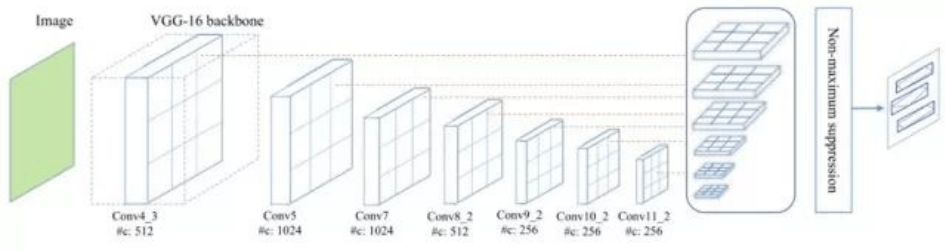


图 3　Textboxes 框架

提出方法

我们的方法也是基于 SSD，整体框架如图 4。为了应对多尺度文字尤其是小文字，对高层特征图进行间隔采样，以保持高层特征图分辨率。同时借鉴特征金字塔网络相关思路，将高层特征图上采样与底层特征叠加，构建一个新的多层级金字塔特征

图（图 4 蓝色框部分）。此外，为了处理各种方向文字，在不同尺度的特征图上预测文字片段以及片段之间的连接关系，然后对预测出的文字片段和连接关系进行组合，得到最终文本框。下面将具体介绍方法。

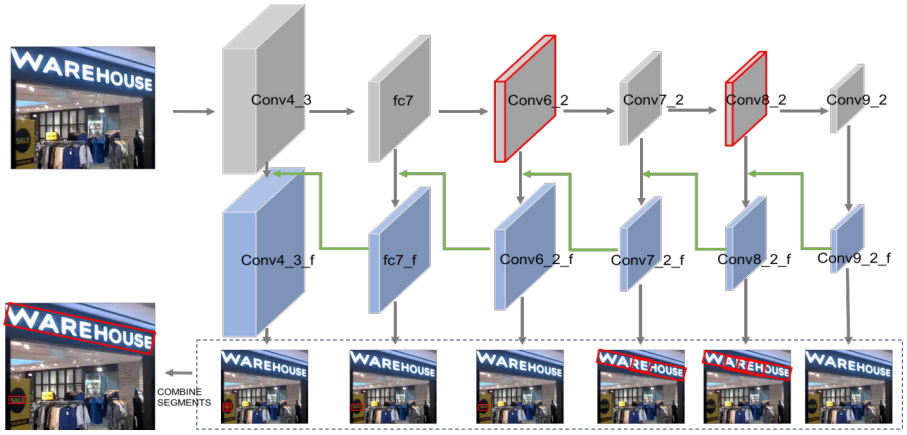


图 4　我们方法框架

（1）扩大高层特征图

深度卷积神经网络通常是逐层下采样，这对于物体分类来说是有效的，但是对于检测任务来说是有损害的。基于时间和性能的权衡考量，我们对卷积网络中最后几层特征进行间隔采样，如图 5，从 Conv6_2 层开始下采样，Conv7_2 层保持原分辨率，Conv8_2 层再下采样。

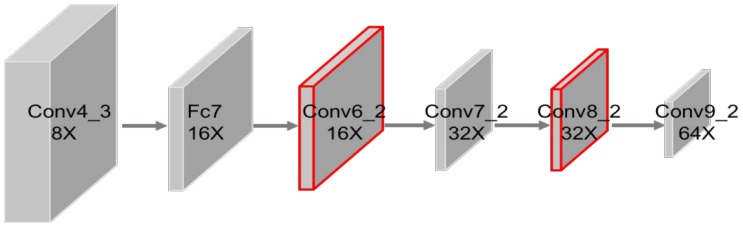


图 5　扩大特征图

（2）构建特征金字塔

虽然通过扩大深度特征图的设计可以更好地检测小文本，但较小的文本仍然难

以检测。为了更好地检测较小的文本，进一步增强较浅层（例如图 5 中 conv4_3，Fc7）的特征。我们通过融合高层和低层的特征构建了一个新的特征金字塔（图 4 中蓝色部分：conv4_3_f,fc7_f,conv6_2_f,conv7_2_f,conv8_2_f 和 conv9_2_f），新的金字塔特征具有更强辨别力和语义丰富性。

高层和低层特征融合策略如图 6 所示，高层特征图先进行上采样使之与低层特征图相同大小，然后与低层特征图进行叠加，叠加后的特征图再连接一个 3*3 卷积，获得固定维度的特征图，我们设定固定维度 d=256。

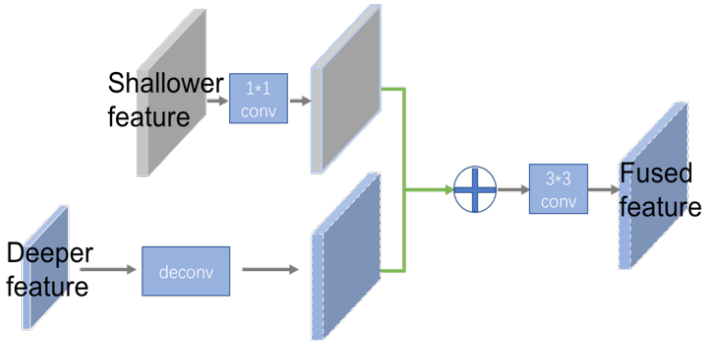


图 6　构建特征金字塔模块

（3）预测文字片段及片段之间连接关系

如图 7，先将每个文字词切割为更易检测的有方向的小文字块（segment），然后用邻近连接（link）将各个小文字块连接成词。这种方案方便于识别长度变化范围很大的、带方向的词和文本行，它不会象 Faster-RCNN 等方案因为候选框长宽比例原因检测不出长文本行，而且处理速度很快。

图 7　小文字块和近邻连接

基于第（2）小节构建的特征金字塔特征图，将每层特征图上特征点用于检测小文字块和文字块连接关系。如图 8，连接关系可以分为八种，上、下、左、右、左上、右上、左下、右下，同一层特征图、或者相邻层特征图上的小文字块都有可能被连接入同一个词中，换句话说，位置邻近、并且尺寸接近的文字块都有可能被预测到同一词中。

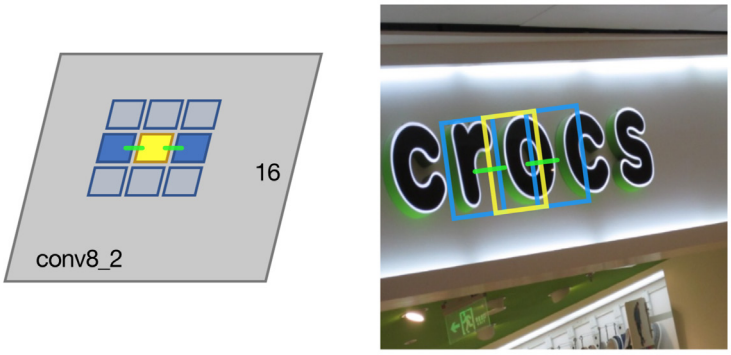


图 8　连接关系示意图

最后基于检测出的小文字块以及文字块连接，组合出文本框（如图 9），具体组合过程如下：

（a）将所有具有连接关系的小文字块组合起来，得到若干小文字块组；（b）对于每组小文字块，找到一条直线能最好的拟合组内所有小文字块中心点；（c）将组内所有小文字块的中心点投影到该直线上，找出距离最远的两个中心点 A 和 B；（d）最终文字框中心点为（A+B）/2，方向为直线斜率，宽度为 A，B 两点直线距离加上 A，B 两点的平均宽度，高度为所有小文字块的平均高度。

图 9　小文字块连接示意图

实验及应用

我们在两个公开数据集上（ICDAR2013，ICDAR2015）对方法进行评测。其中 ICDAR2013 数据集，训练图片 229 张，测试图片 233 张；ICDAR2015 数据集，训练图片 1000 张，测试图片 500 张，它们都来自于自然场景下相机拍摄的图片。

（1）我们首先对比了扩大高层特征图与不扩大高层特征图的性能比较，并在基础上对比加入特征金字塔后的性能比较，在 ICDAR2015 数据集上实验，结果如表 1：

表 1　方法中不同模块有效性验证

Method	Precision	Recall	F-measure
Baseline	79.5	73.4	76.3
扩大高层特征图	81	76.3	78.6
金字塔+扩大高层特征图	88	76.8	82

“baseline”方法是 ssd 框架 + 预测文字片段及片段之间连接关系模块，“扩大高层特征图”是在 baseline 方法基础上对高层特征图进行扩大，“金字塔 + 扩大高层特征图”是在 baseline 方法基础上对高层特征图进行扩大 并且加入特征金字塔。从表 1 中不难发现，扩大高层特征图可以带来精度和召回的提升，尤其是召回有近 3 个点的提升（73.4–>76.3），这很好理解，因为更大的特征图产生更多的特征点以及预测结果；在此基础上再加入金字塔机制，精度获得显著提升，说明金字塔结构极大增强低层特征判别能力。

（2）我们也和其他方法也做了比较，具体见表 2 和表 3：

表 2　ICDAR2013 数据集与其他方法比较

Method	Precision	Recall	F-measure
CTPN	93	83	87.7
TextBoxes++	88	74	81
PixelLink	84.4	82.3	83.3
SegLink	87.7	83.0	85.3
OUR	92.4	83.8	87.9

表 3　ICDAR2015 数据集与其他方法比较

Method	Precision	Recall	F	FPS
SegLink	73.1	76.8	75	-
East	80.5	72.8	76.4	6.5
RRPN	82.2	73.2	77.4	-
TextBoxes++	87.2	76.7	81.7	11.6
PixelLink	82.9	81.7	82.3	7.3
OUR	88	76.8	82	10.3

从上表中可以看出，我们的方法在时间和精度上取得很好的权衡。在 ICDAR2015 数据集上，虽然性能不及 PixelLink，但是 FPS 要远高于它；而相比 TextBoxes++，虽然 FPS 略低于它，但是精度更高。图 10 给出一些文字定位结果示例。

图 10　文字定位结果示意图

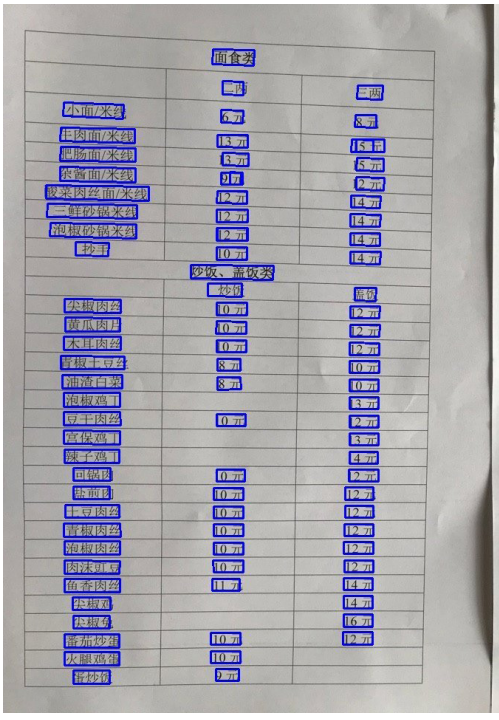
（3）此外，本方法也落地应用于实际业务场景菜单识别中。菜单上文字通常较小、较密，菜名文字可长可短，以及由于拍摄角度导致文字方向倾斜等。如图 11 所示，方法能很好的解决以上问题（小文字、密集文字行、长文本、不同方向）；并且在 500 张真实商家菜单图片上进行评测，相比 SegLink 方法，性能明显提升（近 5 个点提升）。

表 4　菜单测试结果

500菜单测试集	Precision	Recall	F-measure
改进前	85.1	84.7	84.9
改进后	89.5	90.3	89.9


白切鸡 45元/半只
白切鸡 85元/只
农家小炒肉 28元
凉瓜炒牛肉 28元
凉瓜炒蛋 18元
尖椒炒牛肉 28元
青瓜炒腰球 28元
菜心炒牛肉 28元
姜葱炒鳝片 28元
姜葱炒田鸡 28元
香辣牛肉 33元
香辣鸡杂 28元
红烧茄子 20元
韭黄炒蛋 18元
酸辣土豆丝 18元
豆豉鲮鱼炒凉瓜 18元
凉瓜炒咸蛋黄 18元


儿童套餐 19元
双人套餐 33元
全家福套餐 49元
单人套餐
吉祥三宝套餐 27元
阳光小吃组合 25元


面食类
炒饭、盖饭类

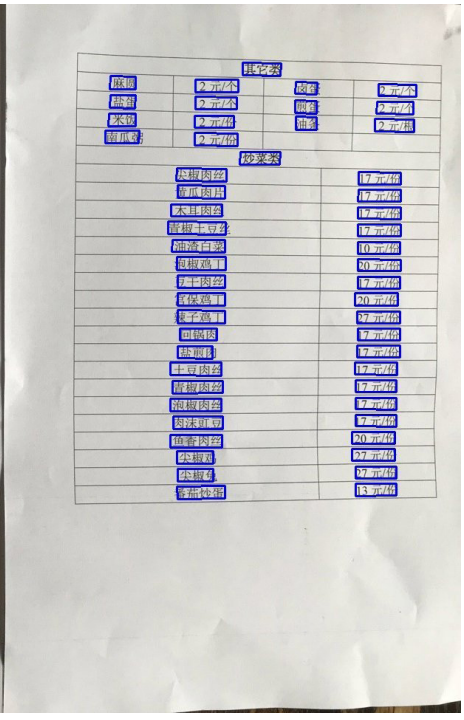

蔬菜类
炒菜类

图 11　菜单文字定位结果示意图

结论

本文我们提出了一个高效的场景文本检测框架。针对文字特点，我们扩大高层特征图尺寸并构建了一个特征金字塔，以更适用于不同比例文本，同时通过检测文本片段和片段连接关系来处理长文本和定向文本。实验结果表明该框架快速且准确，在 ICDAR2013 和 ICDAR2015 数据集上获得了不错结果，同时应用到公司实际业务场景菜单识别上，获得明显性能提升。下一步，受实例分割的方法 PixelLink [4] 的启发，我们也考虑将文本片段进一步细化到像素级，同时融合检测和分割方法各自优缺点，构建联合检测和分割的文字定位框架。

参考文献

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, Serge Belongie. “Feature Pyramid Networks for Object Detection.” arXiv preprint. arXiv: 1612.03144, 2017.

J. Long, E. Shelhamer, and T. Darrell. “Fully convolutional networks for semantic segmentation.” In CVPR, 2015.

M. Liao, B. Shi, and X. Bai. “Textboxes++: A single-shot oriented scene text detector.” IEEE Trans. on Image Processing, vol. 27, no. 8, 2018.
D. Deng, H. Liu, X. Li, and D. Cai. “Pixellink: Detecting scene text via instance segmentation.” In AAAI, pages 6773 - 6780, 2018.

作者简介

刘曦，美团视觉图像中心文字识别组算法专家。

招聘信息

美团视觉图像中心文字识别组：针对美团各项业务如商家入驻资质审核、网页信息合规审核等需求，对证照、票据、菜单、网图等图片类型开展文字识别研发工作。利用高性能文字识别功能，帮助业务方和商家实现自动审核 、自动录入，显著提升人效、降低成本，改善体验。

欢迎计算机视觉相关及相关领域小伙伴加入我们，简历可发邮件至 tech@meituan.com（邮件标题注明：美团视觉图像中心文字识别组）。

CVPR 2019 轨迹预测竞赛冠军方法总结

李鑫

背景

CVPR 2019 是机器视觉方向最重要的学术会议，本届大会共吸引了来自全世界各地共计 5160 篇论文，共接收 1294 篇论文，投稿数量和接受数量都创下了历史新高，其中与自动驾驶相关的论文、项目和展商也是扎堆亮相，成为本次会议的“新宠”。

障碍物轨迹预测挑战赛（Trajectory Prediction Challenge）隶属于 CVPR 2019 Workshop on Autonomous Driving — Beyond Single Frame Perception（自动驾驶研讨会），由百度研究院机器人与自动驾驶实验室举办，侧重于自动驾驶中的多帧感知，预测和自动驾驶规划，旨在聚集来自学术界和工业界的研究人员和工程

师，讨论自动驾驶中的计算机视觉应用。美团无人配送与视觉团队此项比赛获得了第一名。

在该比赛中，参赛队伍需要根据每个障碍物过去 3 秒的运动轨迹，预测出它在未来 3 秒的轨迹。障碍物共有四种类型，包括行人、自行车、大型机动车、小型机动车。每种障碍物的轨迹用轨迹上的采样点来表示，采样的频率是 2 赫兹。美团的方法最终以 1.3425 的成绩取得该比赛的第一名，同时我们也在研讨会现场分享了算法和模型的思路。

赛题简介

轨迹预测竞赛数据来源于在北京搜集的包含复杂交通灯和路况的真实道路数据，用于竞赛的标注数据是基于摄像头数据和雷达数据人工标注而来，其中包含各种车辆、行人、自行车等机动车和非机动车。

训练数据：每个道路数据文件包含一分钟的障碍物数据，采样频率为每秒 2 赫兹，每行标注数据包含障碍物的 ID、类别、位置、大小、朝向信息。

测试数据：每个道路数据文件包含 3 秒的障碍物数据，采样频率为每秒 2 赫兹，目标是预测未来 3 秒的障碍物位置。

评价指标

平均位移误差：Average displacement error（ADE），每个预测位置和每个真值位置之间的平均欧式距离差值。

终点位移误差：Final displacement error（FDE），终点预测位置和终点真值位置之间的平均欧式距离差值。

由于该数据集包含不同类型的障碍物轨迹数据，所以采用根据类别加权求和的指标来进行评价。

$$WSADE = D_v \cdot ADE_{\mathrm{v}} + D_p \cdot ADE_{\mathrm{p}} + D_b \cdot ADE_{\mathrm{b}},$$

$$WSFDE = D_v \cdot FDE_{\mathrm{v}} + D_p \cdot FDE_{\mathrm{p}} + D_b \cdot FDE_{\mathrm{b}},$$

现有方法

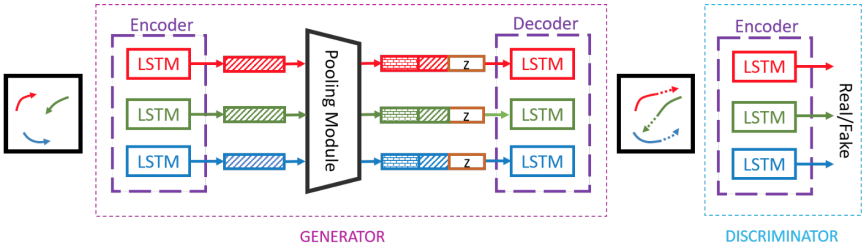
这次竞赛要解决的预测问题不依赖地图和其他交通信号等信息，属于基于非结构化数据预测问题，这类问题现在主流的方法主要根据交互性将其区分为两类：1. **独立预测**，2. **依赖预测**。

独立预测是只基于障碍物历史运动轨迹给出未来的行驶轨迹，依赖预测是会考虑当前帧和历史帧的所有障碍物的交互信息来预测所有障碍物未来的行为。

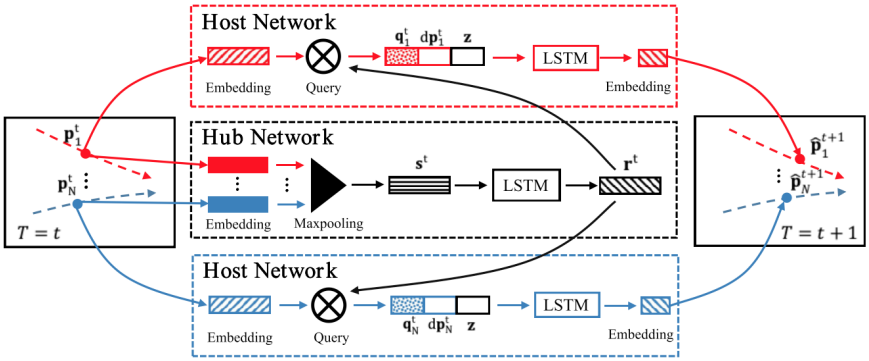
考虑交互信息的依赖预测，是当前学术界研究比较多的一类问题。但是经调研总结，我们发现其更多的是在研究单一类别的交互，比如在高速公路上都是车辆，那预测这些车辆之间的交互；再比如在人行道上预测行人的交互轨迹。预测所有类别障碍物的之间的交互的方法很少。

以下是做行人交互预测的两个方法模型：

方法 1. Social GAN，分别对每个障碍车输入进行 Encoder，然后通过一个统一的 Pooling 模块提取交互信息，再单独进行预测。



方法 2. StarNet，使用一个星型的 LSTM 网络，使用 Hub 网络提取所有障碍物的交互信息，然后再输出给每个 Host 网络独立预测每个障碍物的轨迹。

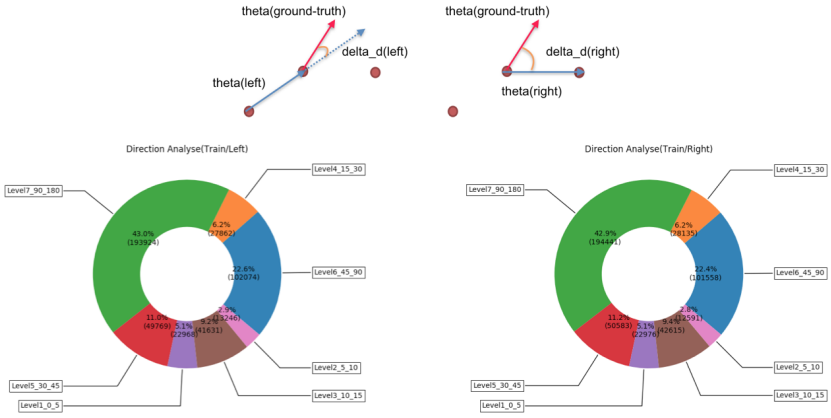


我们的方法

数据分析

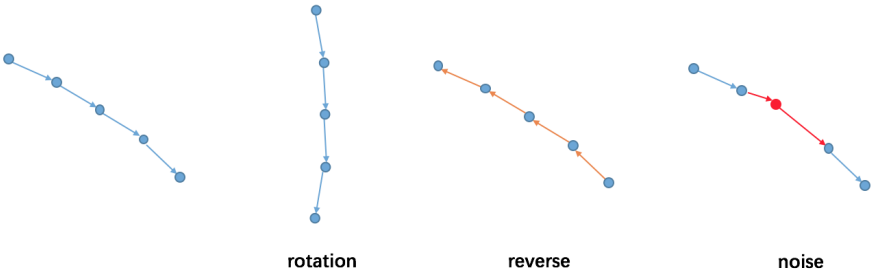
拿到赛题之后，我们首先对训练数据做了分析，由于最终的目标是预测障碍物测位置，所以标注数据中的障碍物大小信息不太重要，只要根据类别来进行预测即可。

其次，分析朝向信息是否要使用，经统计发现真值标注的朝向信息非常不准确，从下图可以看到，大部分的标注方向信息都和轨迹方向有较大差距，因此决定不使用朝向信息进行预测。



然后，分析数据的完整性，在训练过程中每个障碍物需要 12 帧数据，才可以模拟测试过程中使用 6 帧数据来预测未来 6 帧的轨迹。但是在真实搜集数据的时候，没有办法保证数据的完整性，可能前后或中间都可能缺少数据，因此，我们根据前后帧的位置关系插值生成一些训练数据，以填补数据的缺失。

最后，对数据做了增强，由于我们的方法不考虑障碍物之间的交互，仅依赖每个障碍物自身的信息进行训练，因此障碍物轨迹进行了旋转、反向、噪声的处理。

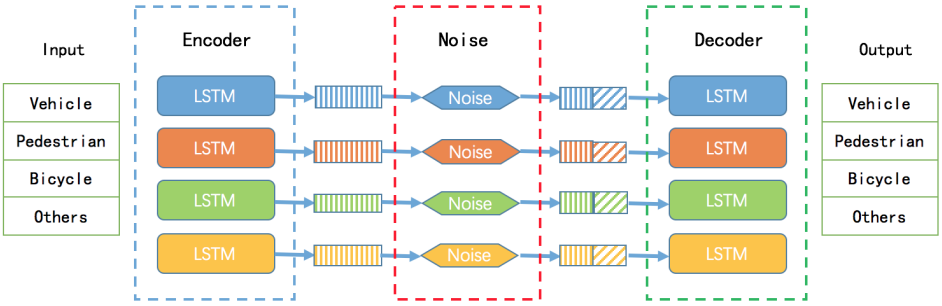


模型结构

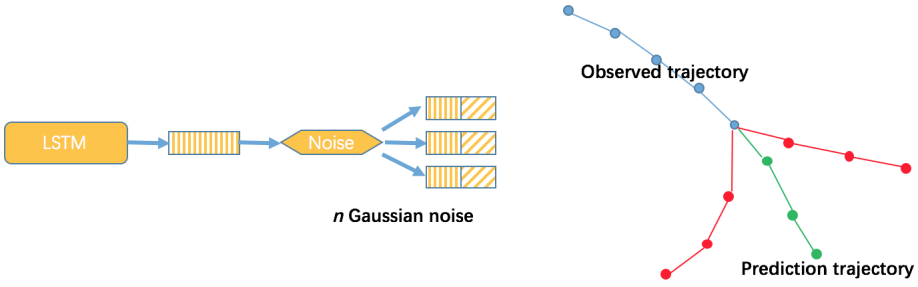
由于这次轨迹预测的问题是预测所有类别的轨迹，所以使用解决单一类别的轨迹预测模型不适用于该问题，而且如果把所有的物体放在单一的交互模型中来，不能正确提取出不同障碍物之间的交互特征。我们尝试了一些方法也证实了这一点。

因此在竞赛中，我们使用了多类别的独立预测方法，网络结构如下图，该方法针对每个类别构造一个 LSTM 的 Encoder-Decoder 模型，并且在 Encoder 和 Decoder 之间加入了 Noise 模块，Noise 模块生成固定维度的高斯噪声，将该噪声

和 Encoder 模块输出的 LSTM 状态量进行连结作为 Decoder 模块的 LSTM 初始状态量，Noise 模块主要作用是负责在多轮训练过程中增加数据的扰动，在推理过程中通过给不同的 Noise 输入，可以生成多个不同的轨迹。



最终，需要在不同的轨迹输出中选择一个最优的轨迹，这里采用了一个简单的规则，选择预测的轨迹方向和历史轨迹方向最接近的轨迹作为最终的轨迹输出。



实验结果

我们仅使用了官方提供的数据进行训练，按照前述数据增强方法先对数据进行增强，然后搭建网络结构进行训练，Loss 采用 Weighted Sum of ADE（WSADE），采用 Adam 优化方法，最终提交测试的 WSADE 结果为 1.3425。

方法	WSADE
我们的方法	1.3425
StarNet（基于交互的方法）	1.8626
TrafficPredict（ApolloScape Baseline 方法）	8.5881

总结

在这次竞赛中，我们尝试了使用多类别的独立预测方法，通过对数据增强和加入高斯噪声，以及最终人工设计规则选择最优轨迹的方法，在这次障碍物轨迹预测挑战赛（Trajectory Prediction Challenge）中获得了较好的成绩。但是，我们认为，基于交互的方法用的好的话应该会比这种独立预测方法还是要好，比如可以设计多类别内部交互和类别间的交互。另外，也关注到现在有一些基于图神经网络的方法也应用在轨迹预测上，今后会在实际的项目中尝试更多类似的方法，解决实际的预测问题。

参考文献

Yanliang Zhu, Deheng Qian, Dongchun Ren and Huaxia Xia. StarNet: Pedetrian Trajectory Prediction using Deep Neural Network in Star Topology[C]//Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2019.

Gupta A, Johnson J, Fei-Fei L, et al. Social gan: Socially acceptable trajectories with generative adversarial networks[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018: 2255-2264.

Apolloscape. Trajectory dataset for urban traffic. 2018. http://apolloscape.auto/trajectory.html.

作者简介

李鑫，美团无人配送与视觉部 PNC 组轨迹预测组算法专家。

炎亮，美团无人配送与视觉部 PNC 组轨迹预测组算法工程师。

德恒，美团无人配送与视觉部 PNC 组轨迹预测组负责人。

冬淳，美团无人配送与视觉部 PNC 组负责人。

顶会论文：基于神经网络 StarNet 的行人轨迹交互预测算法

炎亮　德恒　冬淳　华夏

1. 背景

民以食为天，如何提升超大规模配送网络的整体配送效率，改善数亿消费者在”吃“方面的体验，是一项极具挑战的技术难题。面向未来，美团正在积极研发无人配送机器人，建立无人配送开放平台，与产学研各方共建无人配送创新生态，希望能在一个场景相对简单、操作高度重复的物流配送中，提高物流配送效率。在此过程中，美团无人配送团队也取得了一些技术层面的突破，比如基于神经网络 StarNet 的行人轨迹交互预测算法，论文已发表在 IROS 2019。IROS 的全称是 IEEE/RSJ International Conference on Intelligent Robots and Systems，IEEE 智能机器人与系统国际会议，它和 ICRA、RSS 并称为机器人领域三大国际顶会。

1.1　行人轨迹预测的意义

在无人车行驶过程中，它需要对周围的行人进行轨迹预测，这能帮助无人车更加安全平稳地行驶。我们可以用图 1 来说明预测周围行人的运动轨迹对于无人车行驶的重要性。

图 1 中蓝色方块代表无人车，白色代表行人。上半部分描述的是在不带行人轨迹预测功能情况下无人车的行为。这种情况下，无人车会把行人当做静态物体，但由于每个时刻行人都会运动，导致无人车规划出来的行驶轨迹会随着时间不停地变化，加大了控制的难度，同时还可能产生碰撞的风险，这样违背了安全平稳行驶的目标。下半部分是有了行人轨迹预测功能情况下的无人车行为。这种情况下，无人车会预测周围行人的行驶轨迹，因此在规划自身行驶时会考虑到未来时刻是否会与行人碰撞，最终规划出来的轨迹更具有“预见性”，所以避免了不必要的轨迹变化和碰撞风险。

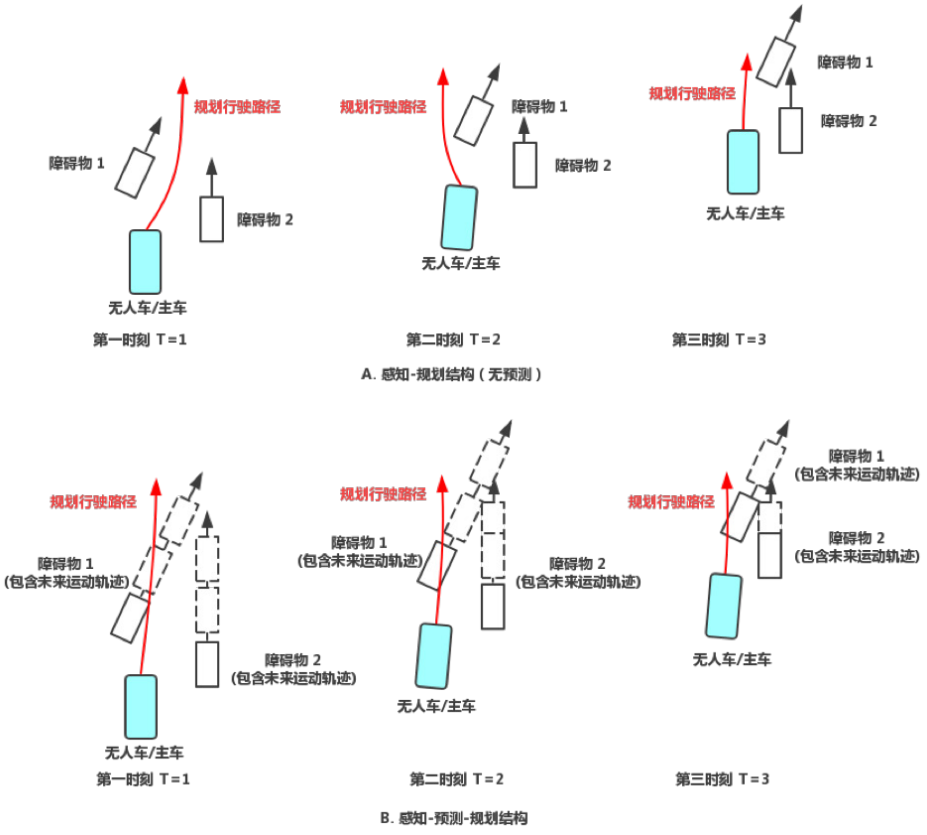


图 1　主车规划轨迹跳变问题

1.2　行人轨迹预测的难点

总体而言，行人轨迹预测的难点主要有两个：

第一，行人运动灵活，预测难度大。本身精确预测未来的运动轨迹是一个几乎不可能完成的任务，但是通过观察某个障碍物历史时刻的运动轨迹，可以根据一些算法来大致估计出未来的运动轨迹（最简单的是匀速直线运动）。在实际中，相比于自行车、汽车等模型，行人运动更加灵活，很难对行人建立合理的动力学模型（因为行人可以随时转弯、停止、运动等），这加剧了行人预测的难度。

第二，行人之间的交互，复杂又抽象。在实际场景中，某一行人未来的运动不仅受自己意图支配，同样也受周围行人的影响（例如避障）。这种交互非常抽象，在算法

中往往很难精确地建模出来。目前，大部分算法都是用相对空间关系来进行建模，例如相对位置、相对朝向、相对速度大小等。

1.3 相关工作介绍

传统算法在做预测工作时会使用一些跟踪的算法，最常见的是各类时序模型，例如卡尔曼滤波（Kalman Filter, KF）、隐马尔可夫（Hidden Markov Model, HMM）、高斯过程（Gaussian Process, GP）等。这类方法都有一个很明显的特点，就是根据历史时序数据，建立时序递推数学公式：$X^t = f\left(X^{t-1}\right)$ 或者 $p\left(X^t \mid X^{t-1}\right)$。因为这类方法具有严格的数学证明和假设，也能处理一些常规的问题，但是对于一些复杂的问题就变得“束手无策”了。这是因为这些算法中都会引入一些先验假设，例如隐变量服从高斯分布，线性的状态转换方程以及观测方程等，而最终这些假设也限制了算法的整体性能。神经网络一般不需要假设固定的数学模型，凭借大规模的数据集促使网络学习更加合理的映射关系。本文我们主要介绍一些基于神经网络的行人预测算法。

基于神经网络的预测算法（主要以长短期记忆神经网络 Long Short Term Memory，LSTM 为主）在最近 5 年都比较流行，预测效果确实比传统算法好很多。在 CVPR（IEEE Conference on Computer Vision and Pattern Recognition）2019 上，仅行人预测算法的论文就有 10 篇左右。这里我们简单介绍 2 篇经典的行人预测算法思路，如果对这方面感兴趣的同学，可以通过文末的参考文献深入了解一下。第一篇是 CVPR 2016 斯坦福大学的工作 Social-LSTM，也是最经典的工作之一。Social-LSTM 为每个行人都配备一个 LSTM 网络预测其运动轨迹，同时提出了一个 Social Pooling Layer 的模块来计算周围其他行人对其的影响。具体的计算思路是将该行人周围的区域划分成 NxN 个网格，每个网络都是相同的大小，落入这些网格中的行人将会参与交互的计算。

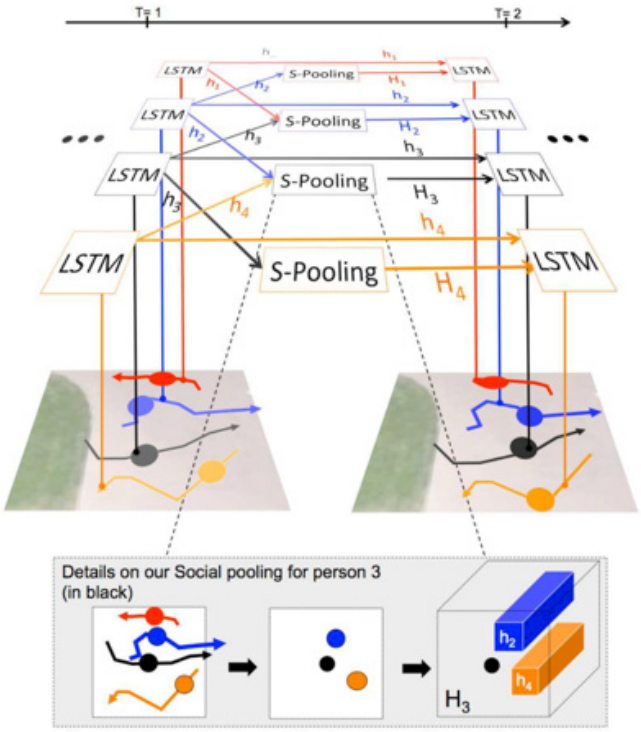


图 2　左：Social LSTM 原理 右：Social Pooling 计算过程

第二篇是 CVPR 2019 卡耐基梅隆大学 & 谷歌 & 斯坦福大学的工作，他们的工作同样使用 LSTM 来接收历史信息并预测行人的未来轨迹。不同于其他算法的地方在于，这个模型不仅接收待预测行人的历史位置信息，同时也提取行人外观、人体骨架、周围场景布局以及周围行人位置关系，通过增加输入信息提升预测性能。除了预测具体的轨迹，算法还会做粗粒度预测（决策预测），输出行人未来时刻可能所在的区域。

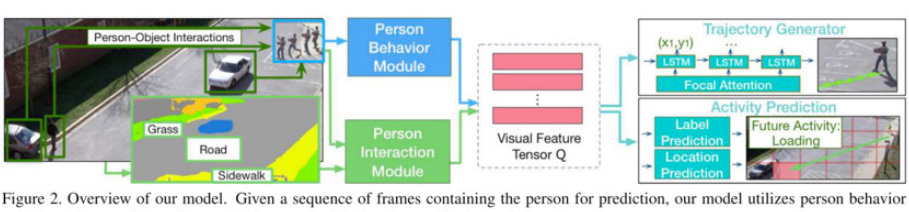


Figure 2. Overview of our model. Given a sequence of frames containing the person for prediction, our model utilizes person behavior module and person interaction module to encode rich visual semantics into a feature tensor.

图 3　算法整体结构

其他的相关工作，还包括基于语义图像 / 占有网格（Occupancy Grid Map, OGM）的预测算法，基于信息传递（Message Passing, MP）的预测算法，基于图网络（Graph Neural Network, GNN）的预测算法（GCN/GAT 等）等等。

2. StarNet 介绍

目前，现有的轨迹预测算法主要还是聚焦在对行人之间交互的建模，轨迹预测通常只使用 LSTM 预测即可。如下图 4 左，现有关于轨迹预测的相关工作基本都是考虑行人之间两两交互，很少有考虑所有行人之间的全局交互（即使是 GCN，也需要设计对应的相似矩阵来构造拉普拉斯矩阵，这也是一个难点）。我们可以举一个例子来说明现有其他算法预测的流程：

- 假设感知模块检测到当前 N 个行人的位置，如何计算第一个行人下一时刻的位置？ Step 1 计算其他人对于第一个行人的交互影响。将第 i 个行人在第 t 时刻的位置记为（一般是坐标 x 和 y）。可以通过以下公式计算第一个行人的交互向量：$Interaction_1^t = f\left(P_2^t - P_1^t, P_3^t - P_1^t, \cdots, P_N^t - P_1^t\right)$

  从上述公式可以大致看到，相对位置关系是最重要的计算指标，计算的函数 f 一般是一个神经网络。 Step 2 计算第一个行人下个时刻的位置。通常需要根据上一时刻的位置与交互向量：$P_1^{t+1} = g\left(P_1^t, Interaction_1^t\right)$ 上述公式中，计算的函数 g 同样是神经网络，即上面提到的长短期记忆神经网络 LSTM。

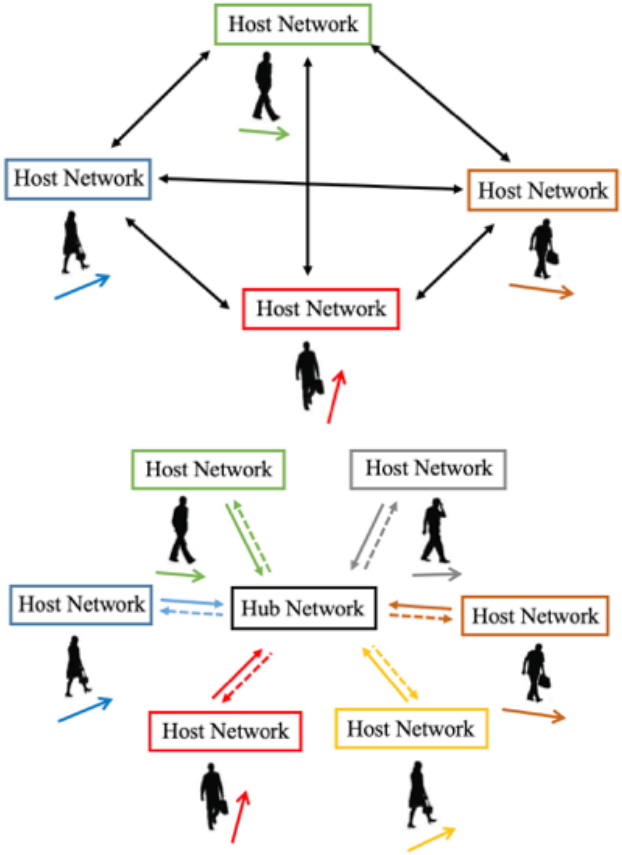


图 4　算法思路对比图 上：传统算法 下：StarNet

两两交互的方式存在两个问题：

1. 障碍物 2 和 3 确实会影响障碍物 1 的运动，但是障碍物 2 和 3 之间同样也存在相互影响，因此不能直接将其他障碍物对待预测障碍物的影响单独剥离出来考虑，这与实际情况不相符。
2. 两两计算消耗的资源大，如果有 N 个障碍物，那么两两交互就需要 N 的平方次计算，随着 N 的变大，计算量呈平方倍增长。我们希望障碍物之间的交互能否只计算 1 次而非 N 次，所有障碍物的轨迹预测都共享这个全局交互那就更好了。

基于上述两个问题，我们提出了一种新的模型，该模型旨在高效解决计算全局交互的问题。因为传统算法普遍存在计算两两交互的问题（即使是基于 Attention 注意力机制的 Message Passing 也很难考虑到全局的交互），本文想尝试通过一些更加简单直观的方式来考虑所有障碍物之间的全局交互，我们的算法大致思路如下：

每个时刻所有障碍物的位置可以构成一张静态的“地图”，随着时间的变化，这些静态地图就变成了一张带有时序信息的动态图。这张动态图中记录了每个区域内的障碍物运动信息，其中运动信息是由所有障碍物一起影响得到的，而非单独地两两交互形成。对于每个障碍物的预测阶段，只要根据该障碍物的位置，就可以在这张时序地图中查询该区域在历史时刻的障碍物运动信息（例如这个区域在历史时刻中，障碍物 1、2、4、5 都有其运动的轨迹）。通过“共享全局交互地图 + 个体查询”的方式，就可以做到计算全局交互以及压缩计算开销。

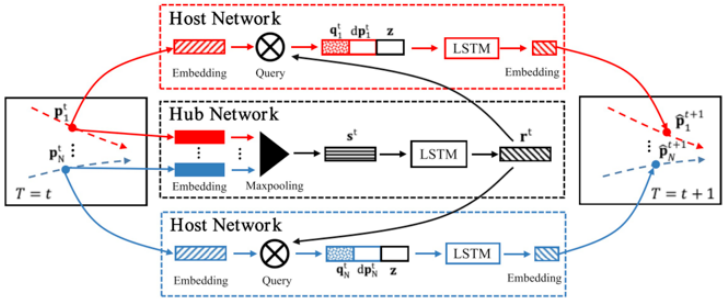


Fig. 2: The process of predicting the coordinates. At time step t, StarNet takes the newly observed (or predicted) coordinates $\{\mathbf{p}_i^t\}_{i=1}^N$ (or $\{\widehat{\mathbf{p}}_i^t\}_{i=1}^N$) and outputs the predicted coordinates $\{\widehat{\mathbf{p}}_i^{t+1}\}_{i=1}^N$.

图 5　StarNet 网络结构图

我们的算法结构如上图 5 所示，Host Network 是基于 LSTM 的轨迹预测网络；Hub Network 是基于 LSTM 的全局时序交互计算网络。在论文具体的实现中，首先 Hub Network 的静态地图模块是通过接受所有障碍物同一时刻的位置信息、全连接网络和最大池化操作得到一个定长的特征向量 s^t；然后动态地图模块使用 LSTM 网络对上述的特征向量 s^t 进行时序编码，最终得到一个全局交互向量 r^t。Host Network 首先根据行人（假设要预测第一个行人下时刻的位置）的位置 P_1^t 去动态地图 r^t 中查询自己当前位置区域内的交互 q_1^t，具体我们采用简单的点乘操作（类似于 Attention 机

制）。最终自己的位置 P_1^t 和交互 q_1^t 一起输入 LSTM 网络预测下时刻的的位置 P_1^{t+1} 。

实验阶段，我们与 4 种经典的算法作比较，使用的数据集为 UCY&ETH 数据集，这两个数据集包含 4 个子场景，分别为 ZARA-1/ZARA-2、UNIV、ETH、HOTEL。在预测过程中，所有算法根据每个行人过去 3.2 秒的运动轨迹，预测出它在未来 3.2 秒的轨迹。每 0.4 秒采样一个离散点，因此 3.2 秒的轨迹可以用 8 个轨迹离散点表示。对比的指标有：

（a）平均距离差 ADE（Average Displacement Error）：用算法预测出的轨迹到真实轨迹所有 8 个点之间的平均距离差。（b）终点距离差 FDE（Final Displacement Error）：用算法预测出的轨迹与真实轨迹最后一个终点之间的距离差。（c）前向预测时间以及参数量。

最终的实验结果如下表：

TABLE II: Prediction Errors

Metric	Dataset	LSTM	Social LSTM	Social GAN	Social Attention	StarNet (Ours)
ADE	ZARA-1	0.25	0.27	**0.21**	1.66	0.25
	ZARA-2	0.31	0.33	0.27	2.30	**0.26**
	UNIV	0.36	0.41	0.36	2.92	**0.21**
	ETH	0.70	0.73	0.61	2.45	**0.31**
	HOTEL	0.55	0.49	0.48	2.19	**0.46**
Average ADE	-	0.43	0.45	0.39	2.30	**0.30**
Variance of ADE	-	0.028	0.026	0.021	0.166	**0.008**
FDE	ZARA-1	0.53	0.56	**0.42**	2.64	0.47
	ZARA-2	0.65	0.70	0.54	4.75	**0.53**
	UNIV	0.77	0.84	0.75	5.95	**0.40**
	ETH	1.45	1.48	1.22	5.78	**0.54**
	HOTEL	1.17	1.01	0.95	4.94	**0.91**
Average FDE	-	0.91	0.91	0.78	4.81	**0.57**
Variance of FDE	-	0.118	0.101	0.802	1.394	**0.031**

TABLE III: Computational Time

Metric	LSTM	Social LSTM	Social GAN	Social Attention	StarNet (Ours)
Inference Time (Seconds)	0.029	0.504	0.202	3.714	**0.073**
Number of Paramters (Kilo)	22.87	156.06	108.03	874.95	**31.90**

从实验结果可以看到，我们的算法在 80% 的场景下都优于其他算法，且实时性高（表中 LSTM 的推理时间为 0.029 秒，最快速是由于该算法不计算交互，因此速度最快参数也最少，但是性能较差）。

总结一下，我们提出算法 StarNet 的优势主要包括以下两点：

- 使用全局动态地图的形式来描述行人之间在时间和空间上的相互影响，更加合理，也更加准确。

• Hub Network 全局共享的特征提升了整个算法的计算效率。

3. 未来工作

首先，我们会进一步探索新的模型结构。虽然我们的算法在数据集上取得了不错的效果，但这是我们的第一次尝试，模型设计也比较简单，如果提升模型结构，相信可以取得更好的结果。

其次，我们会提升预测的可解释性。同现有算法一样，目前的模型对计算到的交互缺乏可解释性，仍然依赖于数据驱动。在今后的工作中，我们将通过对交互的可解释建模来提升预测的准确性。

最后，在构建时序的动态地图过程中，引入对于每个障碍物的跟踪信息。换句话说，我们知道每块区域在各个时间点障碍物的位置，但目前算法没有对障碍物在时序上做跟踪（例如时刻 1 有三个障碍物，时刻 2 三个障碍物运动了得到新的位置，网络输入为三个障碍物的位置信息，但是网络无法理解两个时刻中障碍物的对应关系，这降低了交互的性能），这点在以后的工作中还需要继续改进。

参考文献

[1] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, F. Li and S. Savarese, “Social lstm: Human trajectory prediction in crowded spaces,” in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE 2016, pp. 961–971.

[2] H. Wu, Z. Chen, W. Sun, B. Zheng and W. Wang, “Modeling trajectories with recurrent neural networks,” in 28th International Joint Conference on Artificial Intelligence (IJCAI). 2017, pp. 3083–3090.

[3] A. Gupta, J. Johnson, F. Li, S. Savarese and A. Alahi, “Social GAN: Socially acceptable trajectories with generative adversarial networks,” in 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2018, pp. 2255–2264.

[4] A. Vemula, K. Muelling and J. Oh, “Social attention: Modeling attention in human crowds,” in 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018, pp. 1–7.

[5] Y. Xu, Z. Piao and S. Gao S, “Encoding crowd interaction with deep neural network for pPedestrian trajectory prediction,” in 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2018, pp. 5275–5284.

[6] D. Varshneya, G. Srinivasaraghavan, “Human trajectory prediction using spatially

aware deep attention models," arXiv preprint arXiv:1705.09436, 2017.
[7] T. Fernando, S. Denma, S. Sridharan and C. Fookes, "Soft+hardwired attention: An lstm framework for human trajectory prediction and abnormal event detection," arXiv preprint arXiv:1702.05552, 2017.
[8] J. Liang, L. Jiang, J. C. Niebles, A. Hauptmann and F. Li, "Peeking into the future: Predicting future person activities and locations in videos," in 2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2019, pp. 5725-5734.
[9] A. Sadeghian, V. Kosaraju, Ali. Sadeghian, N. Hirose, S. H. Rezatofighi and S. Savarese, "SoPhie: An attentive GAN for predicting paths compliant to social and physical constraints," in 2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2019, pp. 5725-5734.
[10] R. Chandra, U. Bhattacharya and A. Bera, "TraPHic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions," in 2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2019, pp. 8483-8492.
[11] J. Amirian, J. Hayet and J. Pettre, "Social Ways: Learning multi-modal distributions of pedestrian trajectories with GANs," arXiv preprint arXiv:1808.06601, 2018.

作者简介

朱炎亮，美团无人配送部
钱德恒，美团无人配送部
任冬淳，美团无人配送部
夏华夏，美团无人配送部

招聘信息

美团轨迹预测组招聘深度学习算法工程师，我们希望你：

- 具有扎实的编程能力，能够熟练使用 C++ 或 Python 作为编程语言。
- 具有深度学习相关知识，能熟练使用 TensorFlow 或 Pytorch 作为深度学习算法研发框架。
- 对预测无人车周围障碍物的未来轨迹感兴趣。

欢迎有兴趣的同学投送简历到 tech@meituan.com（邮件标题注明：美团轨迹预测组）。

微信扫码关注技术团队公众号

tech.meituan.com
美团技术博客

新年
快乐