

# 推荐系统中的长序列建模

算法小小怪下士

2025 年 12 月 30 日

## 摘要

随着推荐系统的发展，用户行为序列的建模长度已逐渐跃升至十万级甚至全生命周期。众多厂家都已经验证了在序列长度上 Scaling up 的有效性，而长序列建模的核心问题在于如何在有限的在线时延约束下，尽可能无损地捕捉用户跨越长周期的兴趣演变。

本文梳理了推荐系统长序列建模的技术演进路线：**基础序列建模**：回顾 DIN 与 DIEN 如何通过 Target Attention 和 GRU 开启序列建模的大门；**两阶段检索范式**：分析 SIM、ETA、SDIM 等模型如何通过 GSU/ESU 架构及 LSH 哈希技术解决计算瓶颈；**一致性与全周期**：探讨 TWIN 系列如何解决检索与精排的目标不一致问题，并利用聚类实现全生命周期建模；**端到端 Scaling Law**：重点解读 LONGER 与 STCA 如何通过底层系统优化，验证 Scaling Law 在推荐系统中的有效性；**多模态融合**：介绍 MUSE、DMGIN 等模型如何利用语义信息弥补 ID Embedding 的语义缺失与长尾问题。

## 1 关于推荐系统长序列的深度思考

### 1.1 演进路线：从“两阶段检索”到“端到端 Scaling”

为解决长序列建模，工业界主要经历了以下几种范式的演变：

- **记忆增强**：如 MIMN，试图用固定大小的 Memory 矩阵来压缩无限的未来。但在面对复杂多变的用户兴趣时，固定容量的 Memory 往往成为信息瓶颈，且难以利用现代 GPU 的并行能力。
- **两阶段检索**：这是目前最主流的工业界范式。通过 GSU 先从海量行为中粗排检索出 Top-K，再交由 ESU 进行精排。这种方法成功将复杂度从  $O(N)$  降至  $O(K)$ ，但也引入了著名的“不一致性”问题。为了进一步极致压缩时延，ETA 和 SDIM 利用 LSH 将“检索”转化为“哈希碰撞”。SDIM 更是激进地提出哈希碰撞概率在数学期望上等价于 Attention 权重。
- **预训练离线 Embedding**：如 Meta 的 DV365。该方法放弃了在线实时处理超长序列，转而通过离线“基础模型”预先计算用户长序列表示。其核心基于“稳定兴趣假设，认为长周期兴趣对实时性不敏感”。
- **端到端暴力美学**：以字节的 LONGER 和 STCA 为代表。在算力充裕的背景下，不再依赖“检索”这种有损的过滤手段，而是通过 Global Token、KV Cache、混合精度训练等底层系统优化，直接对超长序列进行完整的 Attention 计算。这标志着推荐系统正式进入了遵循 Scaling Law 的时代。

### 1.2 迈向“超长”序列

早期的长序列仅指 1000 长度（SIM 时代），而现在的“超长”通常指覆盖用户全生命周期的  $10^5$  级别。当序列长度达到  $10^5$ ，即使是高效的 LSH 或两阶段模型也面临巨大的存储和 IO 压力。

- **分治与聚类**：TWIN v2 和 C-Former 均采用了“分治”思想。既然无法处理十万个 Raw Item，那就将其聚类为数千个 Cluster。C-Former 更是巧妙地将聚类过程参数化，利用 Transformer 的 Query-Key 机制实现端到端的语义聚类。
- **系统级协同设计**：LONGER 的成功不仅是算法的胜利，更是系统工程的胜利。通过引入 Token Merge、Global Tokens 以及 Parameter Server 到全 GPU 同步训练的架构迁移，证明了只要系统优化得当，Attention 机制本身具有强大的抗噪和长距离捕捉能力。

### 1.3 The Bitter Lesson: Trick 终将被 Scaling Law 碾压？

在阅读完字节大刀阔斧的 LONGER 和 STCA, 以及对比了精巧设计的 ETA/SDIM 后, 我不禁想起了 Rich Sutton 的著名文章《The Bitter Lesson》。

“在人工智能的历史长河中, 利用算力的通用方法, 最终总是会打败利用人类知识的特定方法。”

- **短期来看:** 利用我们对推荐领域的深刻理解去设计特殊的架构或 Trick (如 SIM 的类目索引、SDIM 的哈希采样、两阶段的漏斗设计), 是在算力受限条件下“戴着镣铐跳舞”的最优解, 往往能带来立竿见影的 ROI。
- **长期来看:** 随着摩尔定律和硬件架构的演进, 那些能够充分利用算力进行大规模搜索和学习的通用算法 (如 Standard Attention、Transformer), 会因为其强大的 Scaling 能力, 无情地碾压那些依赖人类“小聪明”设计的近似方法。

目前的趋势已经初现端倪: 从精心设计的 Hard Search/LSH, 逐渐转向基于聚类的 Soft Search, 最终迈向全量 Attention。作为算法工程师, 我们既要掌握当下的精巧 Trick 以解决实际问题, 更要时刻关注 Scaling Law 的洪流, 因为那代表着未来的方向。

## 2 序列建模: DIN&DIEN

相较于早期仅关注特征交叉 (如 Wide and Deep) 并将序列信息固化为统计特征的做法, DIN [16] 革新了用户行为序列的建模思路。虽然 SASRec 等序列推荐模型也几乎在同一时期利用自注意力机制探索了 SR, 但 DIN 在工业界 CTR 预估场景中的贡献在于: 它率先指出了‘Embedding 向量定长压缩’这一核心痛点, 并提出了基于 Target Attention 的解决方案。

### 2.1 DIN 和 Target Attention 的区别

在传统的注意力机制中, 注意力权重的计算通常基于欧几里得空间内的简单线性变换。给定用户行为序列中的某一历史交互物品嵌入向量  $\mathbf{e}_u \in \mathbb{R}^d$  与目标候选物品嵌入向量  $\mathbf{e}_v \in \mathbb{R}^d$ , 其相关性分数  $\alpha$  往往通过点积或双线性形式计算得出。

尽管这种计算方式计算高效, 但其本质上强加了一种线性几何距离的归纳偏置, 将高维向量间的复杂交互关系过早地压缩为一个标量, 导致了大量潜在交互信息的损失。相比之下, DIN 不再直接计算标量距离, 而是显式地构建了包含原始特征及其交互信息的组合输入向量  $\mathbf{x}_{in}$ 。DIN 将历史物品向量  $\mathbf{e}_u$ 、目标物品向量  $\mathbf{e}_v$  以及两者的逐元素差值进行 Concat:

$$\mathbf{x}_{in} = [\mathbf{e}_u; \mathbf{e}_v; \mathbf{e}_u - \mathbf{e}_v]$$

某些变体中还会进一步引入逐元素积,  $\mathbf{e}_u \circ \mathbf{e}_v$ ) 以增强交互表达。随后, 该组合向量  $\mathbf{x}_{in}$  被输入到一个 MLP 中, 以拟合复杂的非线性关系:

$$w_{u,v} = g(\mathbf{e}_u, \mathbf{e}_v) = \text{MLP}(\mathbf{x}_{in}) = \mathbf{w}_2^\top \sigma(\mathbf{W}_1 \mathbf{x}_{in} + \mathbf{b}_1) + b_2$$

其中  $\sigma(\cdot)$  为激活函数 (如 Dice)。

这种设计的核心优势在于信息无损性。通过保留原始向量  $\mathbf{e}_u$  和  $\mathbf{e}_v$  的完整维度信息, 并显式引入差分特征  $\mathbf{e}_u - \mathbf{e}_v$  作为辅助输入, 模型得以在全连接层的高维空间中自适应地学习特征间的交互模式。

### 2.2 DIEN 和 DIN 在刻画用户短期兴趣上有什么区别

首先, DIN 和 DIEN [15] 的第一个区别在于对于兴趣本质的定义。DIN 直接将用户的显式行为视为兴趣本身, 即用户的历史行为即代表了其过去的兴趣点。DIEN 则认为显式的行为只是潜隐兴趣的表现载体, 潜隐兴趣很难完全通过显式行为直接反映。因此, DIEN 使用了一个兴趣提取层, 从用户的行为序列中提取出兴趣 emb。作者选择了 GRU 网络来对用户行为之间的依赖进行建模。GRU 既能够克服 RNN 梯度消失的问题, 同时又比 LSTM 网络具有更少的参数, 训练时收敛速度更快。

其次, 两者的区别在于对于行为序列依赖关系的捕捉。为了更精准地刻画短期内的兴趣导向, DIEN 提出了辅助损失。该机制利用“当前兴趣状态导致了下一个行为发生”这一原则, 使用下一个真实行为和负采样行为来监督当前时刻兴趣状态的学习。这使得隐藏状态能够更有效地代表每一步的潜隐兴趣。

### 2.3 DIEN 如何更好的刻画用户长期兴趣的变化

DIEN 认为用户的兴趣是随时间动态演变的，且存在兴趣漂移现象（即用户的意图在相邻访问中可能大不相同，或者当前的决策依赖于很久之前的行为）。因此，DIEN 设计了兴趣演化层，结合了注意力机制和序列学习。DIEN 提出了带有注意力更新门的 GRU (AUGRU)。它不仅像 DIN 一样计算历史兴趣与当前目标的关联度，还进一步将这种关联度融入到序列的更新过程中。AUGRU 强化了与目标物品相关的兴趣演化轨迹，同时削弱了由兴趣漂移导致的无关兴趣的影响。这使得模型能够捕捉到针对特定目标物品的兴趣演化趋势，而不仅仅是相关的历史点。

## 3 两阶段检索：阿里，美团，快手

### 3.1 SIM

SIM (Search-based Interest Model) [9] 针对用户超长生命周期行为序列建模难题，开创了由 GSU+ESU 组成的两阶段级联搜索范式，通过从海量历史行为中快速检索出与当前候选目标最相关的子序列，实现了计算效率和效果的平衡。

#### 3.1.1 Motivation

DIN & DIEN 强调面向候选物品的多样兴趣，用注意力在短序列里做软检索，赋予不同历史行为不同的权重，但这样的注意力计算随序列长度呈平方级增长，在线上时延上难以承载过长序列。

在 SIM 之前，阿里使用 MIMN，把长期兴趣压进固定内存矩阵。这种解耦的设计虽然便于线上服务，但当序列极长且兴趣点分散时，固定容量的内存矩阵会导致信息压缩损耗，进而“稀释”了与当前目标物品高度相关的局部细节信息，限制了模型捕捉精准兴趣的能力。

为了解决上述计算效率与信息无损之间的矛盾，SIM 提出两阶段建模，第一阶段使用 GSU，从用户行为长序列中检索出 Top-k 个行为组成序列。这里检索的方式分为 Hard search 和 Soft Search。

#### 3.1.2 GSU

Hard search 是基于规则进行匹配，仅保留与候选物品同类目的历史物品。在实现 Soft Search 时，直接复用 CTR 模型中基于短期行为训练得到的 Embedding 进行检索存在风险。由于长期用户行为的分布与短期用户行为的分布存在显著的不一致性，仅依靠短期样本训练出的 Embedding 难以准确表征长期历史中的语义关系，这会对检索结果产生误导。为了消除这种分布差异，SIM 引入了一个辅助任务来预训练用于 Soft Search 的 Embedding。该辅助任务旨在利用用户的全量长期行为序列来预测下一个点击行为。具体做法是将用户的 Life-long 行为序列的 Embedding 进行 Sum Pooling，随后与候选物品 Embedding 进行 Concat，最后通过全连接层计算交叉熵损失。通过这种联合训练，模型能够学习到适用于长序列检索的 Embedding 空间，使其具备更好的泛化能力和语义匹配度，从而在 GSU 阶段实现更精准的 Top-k 筛选。

#### 3.1.3 ESU

在经过 GSU 筛选得到长度可控的 Top-k 子序列后，ESU 负责进行精细化的兴趣建模。让候选物品作为 Query，与子序列中的 Key 和 Value 进行交互。最终输出的 Interest Embedding 与其他特征 Embedding 拼接后，输入 MLP 层进行最终的 CTR 预估。

#### 3.1.4 Time Info 的特征工程

由于用户的兴趣随时间衰减，近期行为往往比远期行为具有更大的参考价值。SIM 采用了一种显式的时间编码方式：首先，计算历史行为发生时间与当前请求时间的时间差  $\Delta t$ 。然后，将连续的时间差  $\Delta t$  离散化为 categorical ID，并通过 Lookup Table 映射为时间嵌入向量  $\mathbf{e}_{\text{time}}$ 。最后，将该时间嵌入向量与对应的行为物品嵌入向量  $\mathbf{e}_{\text{item}}$  进行 Concat，形成包含时序信息的这种处理方式显式地引入了时间偏置，使得 Attention 机制在计算权重时能够同时考量内容相关性和时间远近，从而显著提升了模型对动态兴趣的捕捉能力。

### 3.2 ETA

#### 3.2.1 Motivation

针对长序列建模中两阶段不一致以及检索索引更新滞后这两个核心痛点，ETA [4] 提出了一种基于局部敏感哈希 (LSH) 的端到端检索范式，其核心改进如下：

1. 传统 Soft Search 依赖离线构建的倒排索引，导致检索用的 Embedding 与线上实时更新的 CTR

模型 Embedding 存在版本代差。ETA 摒弃了外挂的辅助索引，直接复用 CTR 主模型中的 Item Embedding。通过 SimHash 算法，将高维的浮点数 Embedding 实时映射为紧凑的 code。由于 SimHash 具备局部敏感特性 (Locality-Sensitive)，指纹的相似性能够保真地反映原始 Embedding 的几何距离，且随着模型参数的在线更新，code 也保持同步更新，从而彻底消除了检索与排序之间的语义鸿沟。

2. 为了满足线上极严苛的时延要求，ETA 将 Top-K 检索的计算复杂度从  $O(L \cdot B \cdot d)$  的浮点矩阵乘法降低为  $O(L \cdot B)$  的汉明距离计算。在线推理时，模型只需查表获取长序列中物品的二进制指纹，并通过位运算 (XOR 和 Bit Count) 快速计算其与目标物品指纹的汉明距离，筛选出 Top-K 个相关行为。

### 3.2.2 SimHash

LSH 的设计目标是让相似的数据点以更高的概率发生哈希冲突，即映射到同一个哈希桶或拥有相同的指纹；而不相似的数据点发生冲突的概率则尽可能低。

最常用的 LSH 是针对余弦相似度 (Cosine Similarity) 设计的 SimHash。

首先，生成  $k$  个随机向量  $\mathbf{r}_1, \dots, \mathbf{r}_k$ ，其中每个分量服从标准正态分布 (对应  $k$  个随机超平面)。然后，对于每一个随机向量  $\mathbf{r}_i$ ，计算其与输入向量  $\mathbf{v}$  的内积符号：

$$h_{\mathbf{r}_i}(\mathbf{v}) = \text{sign}(\mathbf{v} \cdot \mathbf{r}_i) = \begin{cases} 1, & \text{if } \mathbf{v} \cdot \mathbf{r}_i \geq 0 \\ 0, & \text{if } \mathbf{v} \cdot \mathbf{r}_i < 0 \end{cases}$$

最后，将  $k$  个哈希位拼接，得到最终的  $k$ -bit 结果：

$$H(\mathbf{v}) = [h_{\mathbf{r}_1}(\mathbf{v}), h_{\mathbf{r}_2}(\mathbf{v}), \dots, h_{\mathbf{r}_k}(\mathbf{v})]$$

SimHash 的几何解释：利用随机超平面将空间划分为不同的区域。对于两个向量  $\mathbf{u}, \mathbf{v}$ ，它们之间的夹角为  $\theta$ 。根据几何概率，它们落在随机超平面同一侧 (即哈希值相等) 的概率为：

$$\Pr[h(\mathbf{u}) = h(\mathbf{v})] = 1 - \frac{\theta}{\pi}$$

这表明，两向量夹角  $\theta$  越小 (余弦相似度越高)，它们的 SimHash 指纹相同的概率越大。

### 3.2.3 RQ-VAE 是否能在 ETA 中替代 LSH

这里引发了小小怪的思考，是否可以用 RQ-VAE 替代 SimHash 作为 ETA 的 Semantic Indexing for Retrieval。目前是看到这样的工作。ETA 使用随机投影的 SimHash 生成指纹，这是一种“无监督”且“非语义”的哈希。RQ-VAE 生成的 Code 本身就是基于语义聚类学习得到的。如果 RQ-VAE 替代 LSH 的好处是，RQ-VAE 的 Code 包含了层级语义信息，比随机投影的 SimHash 更能反映物品的类别和语义相似度。SimHash 是使用汉明距离，按位比对。RQ-VAE 的 Code 是整数元组，且具有层级结构，可以设计 Beam-Search 策略检索。

## 3.3 SDIM

Motivation 虽然 ETA 利用 LSH 实现了端到端的长序列建模，并在一定程度上解决了检索与 CTR 目标不一致的问题，但 SDIM [1] 认为 ETA 依然存在“检索”的思维定势：ETA 使用 LSH 将 Item 转换为哈希签名后，依然执行了一个“计算汉明距离并选取 Top-K”的步骤。SDIM 认为这种检索 Top-K 的操作会导致信息丢失，且难以平衡检索算法的效果与效率。因此 SDIM，不采用先检索再做 Attention，采样 (哈希碰撞) 本身就可以近似 Attention。SDIM 试图证明：利用 LSH 的碰撞概率直接聚合行为，在数学期望上等价于 Softmax Attention。

### 3.3.1 Method: 基于哈希采样的注意力

若用户行为序列中的物品和目标物品发生了碰撞则认为他们相似，直接将所有发生碰撞的物品 gather 起来，作为用户兴趣的表达。为了降低方差，SDIM 并行采样  $m$  个哈希函数，并按宽度  $\tau$  组合成签名。最终的注意力分数是多次哈希结果的平均 10101010。

### 3.3.2 Sampling is all you need

效果上，论文证明了 SimHash 的碰撞概率与两个向量夹角的线性关系：

$$E[\tilde{p}_j] = 1 - \frac{\arccos(q^T s_j)}{\pi}$$

这意味着, SDIM 计算的哈希碰撞概率分布, 在形状上与 Softmax Target Attention 高度一致。因此, 正如论文所说的: Sampling is all you need——不需要显式的 Softmax, 哈希采样本身就是一种 Attention。

效率上, 通过提前预计算历史序列的 hash 值, CTR Server 对长序列处理的时间复杂度从  $O(L)$  降到了  $O(1)$ , 实现了 Latency-free。

### 3.4 TWIN

TWIN [3] 核心想解决的是长序列用户行为建模中, 两阶段架构存在的“不一致性”问题。上面提到的方法中, GSU 和 ESU 使用的相关性度量标准不一致。一方面, GSU 可能因为算法太粗糙, 把 ESU 认为非常重要的行为给过滤掉了。另一方面, GSU 召回了一堆 ESU 认为不相关的行为, 浪费了 ESU 的计算资源。

TWIN 使用的方法是强制 GSU 使用与 ESU 完全相同的 Target Attention 逻辑和参数。为了在 GSU 阶段处理上万长度序列时还能跑得快, TWIN 将特征拆分为固有特征 (Inherent, 如视频 ID, 可预计算缓存) 和交叉特征 (Cross, 如点击时间, 压缩为 Bias 项)。对于固有特征, 可以离线把每个物品的固有特征做好映射缓存起来, 线上并不需要重复计算。对于交互特征, 由于它不是共享的, 不能提前计算, 因此作者采用简化线性投影的方式。将每个交叉特征压缩为 1 维, 这等于将投影矩阵限制为对角块矩阵。在计算 Target attention 时, 经过压缩变为 1 维的交叉特征直接作为 bias 加入到计算中, 同时加入一个可学习的参数, 调整这个 bias 的幅度。

### 3.5 Twinv2

Twin-v2 [12] 的出发点是进一步提升长序列, 希望能建模用户全生命周期的序列特征。为了在保持高效在线推理的同时实现全生命周期建模, TWIN V2 采用了“分治法”, 将系统分为离线压缩和在线推理两个部分。离线部分完成全生命周期分层聚类。

#### 3.5.1 离线: 层聚类算法

层聚类算法: 首先根据视频的播放完成率将历史行为分为  $M$  组。这是为了保证聚类内部的用户兴趣强度分布是均衡的。在每个组内, 利用 Item Embedding 进行 K-means 聚类。采用递归方式, 直到每个簇内的大小不超过设定的阈值  $\gamma$  (文中设为 20)。通过这种方式, 原本长度为  $T$  的序列被压缩为长度约为  $T/10$  的簇序列  $C$ 。

簇表示提取: 为了让簇能像普通 Item 一样参与计算, 需要把每个簇抽象为一个“虚拟 Item”: 数值类特征: 取簇内所有 Item 的平均值。类别类特征: 取簇内与质心距离最近的那个 Item 的特征。

层次聚类每 2 周完整更新一次, 每个簇的最大数量设置为 20, 另外, Embedding Server 会从 GSU 中的固有属性中, 每隔 15 分钟进行同步。

#### 3.5.2 在线: Cluster-Aware Target Attention。

在线推理阶段, 模型使用压缩后的簇序列作为输入, 依然沿用两阶段 (GSU + ESU) 架构, 但 GSU 和 ESU 的输入不再是单条行为, 而是离线生成的簇序列。这意味着即使只检索 Top 100 个输入, 实际上覆盖了背后上千条原始行为。这里 TWINv2 在计算相关性得分的时候和 TWIN 是一样的, 会对交叉特征部分进行简化。

注意力分数重加权: 传统的 Target Attention 计算出的相关性分数  $\alpha$  无法反映簇的大小。论文认为, 如果两个簇与 Target 的相关性相同, 那么包含更多行为数量的簇应该更重要, 这代表用户更强烈的用户偏好。因此, 通过引入“簇大小”作为 Attention 的 Bias, 修正后的注意力得分为:  $\alpha' = \alpha + \ln n$ , 其中  $n$  是簇的大小 (包含的原始 Item 数量)。这里之所以加上  $\ln n$ , 是因为在计算注意力得分时, 还会做 Softmax, 这样就变成了使用样本数进行加权注意力了。

### 3.6 C-Former

TWINv2 采用的是非参数化的聚类方法, 在离线阶段使用 k-means 对历史行为依据 Item Embedding 的欧氏距离进行聚类。C-Former [7] 认为这种聚类方式无法处理高维稀疏数据之间的语义关系, 并且聚类过程与 CTR 推荐任务是 decoupled。

而 Transformer 中的 Query-Key 交互机制与聚类中的 Centroid-Data 交互机制非常相似。Query 就像聚类中心, 去 Attend 相关的数据点/Keys。C-Former 旨在利用这种相似性, 将聚类从欧氏距离转变为语义距离, 并实现端到端的训练。

C-Former 提出一种参数化的聚类方法。它利用 Transformer 的 Encoder 结构, 将一组可学习的“Clustering Anchor Points”作为 Query, 去“查询”用户的历史行为序列 (Keys/Values)。这种交互过

程本质上就是一种聚类, Attention 权重决定了哪些行为属于哪个 Cluster。Anchor Points 根据注意力权重自动聚合语义相似的行为, 逐步迭代更新, 最终输出精炼后的聚类表示 ( $O^{enc}$ ) 和注意力权重矩阵 ( $A$ )。

在训练策略上, C-Former 又有点借鉴了 VAE 的思想, 构建了一个辅助模块 C-Former Decoder, 目标是为了保证聚类中心保留了足够丰富的原始信息, 防止信息丢失。原始用户行为序列 ( $E^S$ ) 作为 Queries; Encoder 输出的精炼 Anchor Points ( $O^{enc}$ ) 作为 Keys 和 Values。这个模块的目标是尝试利用这些 Anchor Points 来重建原始的全生命周期行为序列。

最终 C-Former 由三部分 loss 组成:

$\mathcal{L}_{bce}$  Binary Cross-Entropy Loss: CTR 预测任务的主损失, 利用去噪后的 Centroids 进行推荐, 保证聚类结果对推荐有效。

$\mathcal{L}_{recon}$  Reconstruction Loss: Encoder 输入与 Decoder 输出之间的重构损失, 确保聚类中心的信息量。

$\mathcal{L}_{orth}$  Orthogonal Loss: 对 Anchor Points 施加正交约束, 强制不同的聚类中心之间保持差异性, 促进用户兴趣的多样性表达。

## 4 端到端入图

这两篇论文, 我们都将从算法和工程两个角度分析。

### 4.1 字节: LONGER

LONGER [2] 在算法上的 trick 主要有以下三个点。

1. Global Token。LONGER 的输入分为两个部分, 一个是用户行为 token, 一个是 global token。这个 global token 包括目标物品、用户 ID、CLS Token。该 global token 的作用一方面是作为信息锚点, 聚合全序列信息, 另一方面, 借鉴 StreamLLM 的发现, 利用全局 Token 对抗 “Attention Sink” 现象, 帮助模型在长序列中保持对长距离依赖的捕捉能力。

2. Token merge with Inner Transformer。为了降低计算量, 将长序列中相邻的  $K$  个 Token 合并为一个。简单的拼接会导致信息丢失, LONGER 在合并前, 先在每个小分组内部运行一个轻量级的 Inner Transformer 来捕捉局部交互细节, 然后再进行合并。

3. Hybrid Attention。论文提到, 这种混合注意力设计的动机是观察到模型性能相对于序列 token 的数量表现出强烈的边际效应: 仅采样完整序列的 40% 就保留了超过 95% 的性能改进, 同时减少了大约 50% 的 FLOPs。第一层 Attention 是 Cross-Causal Attention: 让 Query 去 “扫描” 全序列, 提取关键信息, 将超长序列压缩为较短的表示。Query 是由  $m$  个 Global Tokens 和  $k$  个采样出的近期 Sequence Tokens 组成 (实验证明 “最近的  $k$  个” 效果最好)。Key/Value: 完整的超长输入序列  $R$ 。后续层是 Self-Causal Attention (自因果注意力): 在第一层输出的较短序列上堆叠标准的 Self-Attention 层, 进行高阶特征交互。

为了实现端到端建模, LONGER 进行了一系列底层优化。抛弃了传统的 Parameter Server 架构, 采用基于 GPU 的全同步训练。稠密参数和稀疏参数都在 GPU 集群上同步更新, 利用 HBM-MEM-SSD 分层存储来解决大 Embedding 表的存储问题。除此之外, 还使用了 KV cache, 混合精度训练和重计算。

### 4.2 字节 STCA

STCA [5] 算法上的 Trick 如下:

1. 移除 Self-Attention。论文认为, 预测用户是否点击 Target Item, 最重要的是 Target 与 History 的交互, 而不是 History 内部物品之间的交互。因此, STCA 完全移除了极其昂贵的 History Self-Attention。

2. 单 Query 交叉注意力: 每一层只使用 Target Item 作为唯一的 Query。全量历史序列作为 Key 和 Value。

3. 深层堆叠: 为了弥补移除 Self-Attention 带来的表达能力下降, STCA 堆叠了多层。通过 Target-Conditioned Fusion, 将低层的输出融合后作为下一层的 Query, 从而逐步捕获高阶依赖关系。

系统优化: 1. Request Level Batching, 这是一个针对工业界流量特点的系统级优化。在抖音等推荐场景, 一次 Request 通常需要给同一个用户打分多个候选视频。传统做法: 把 (User, Item A), (User, Item B) 当作独立样本处理。这意味着同一个用户的 10k 长序列会被重复编码多次, 浪费了巨大的带

宽和计算资源。RLB 则在请求级别，只对用户的长序列编码一次。将编码后的 User Representation 复用于该请求下的所有 Target Item。

2. 稀疏训练，稠密推理。为了解决训练阶段 10k 序列带来的显存墙问题，论文借鉴了 LLM 的外推思想。在训练时，不使用全长的 10k 序列，而是随机采样较短的序列（平均长度 2k）进行训练。在线上服务时，则使用完整的 10k 序列。为了让模型既见过短序列也见过长序列，学会适应不同长度，使用 U 型 Beta 分布采样序列长度，在采样的时候保留最近的时间切片，确保捕捉最新的用户兴趣。

## 5 预训练离线 Embedding

### 5.1 DV365

在 Meta 的 DV365 [8] 的论文中，作者假设长历史主要反映用户的“稳定兴趣”，这种兴趣不会随时间剧烈变化。因此，不需要像捕捉“即时兴趣”那样进行高频实时训练，离线生成的嵌入即使有一定延迟也是有效的。除此之外，Instagram 有 20 多个不同的推荐模型（Reels, Feed, Explore 等）。离线计算一次嵌入，分发给 15+ 个下游模型使用，极大地分摊了计算成本。最后，虽然 E2E 方法需要在每次请求时实时提取和处理长序列，而离线方案只需查表，节省了大量的特征工程和推理计算资源。因此该论文选择离线 Embedding 表征用户长序列。

DV365 的核心是一个上游基础模型，它定期训练并导出通用的用户嵌入，供下游模型使用。

#### 5.1.1 数据建模：UniTi 与 Multi-slicing

为了搞笑处理超长序列，作者采用了规则式的切片和聚合策略。首先，统一了用户历史的时间线。将用户历史标准化为显式（Explicit，如点赞、分享）和隐式（Implicit，如停留时长）两条时间线。

然后，根据 Multi-slicing 策略将长序列切分为多个子序列，从而进行特征工程。切分策略可以按动作类型切片：例如将“点赞”和“评论”分开。也可以按观看时间切片：根据停留时长或观看比例（Watch Ratio）切分隐式反馈。另外，针对视频时长带来的偏差（长视频天然停留久），设计了去偏的加权特征。最后，对切片后的子序列进行 Mean-pooling 或 Weighted mean-pooling，最终生成 200 个聚合特征嵌入。

#### 5.1.2 上游模型架构

为了让生成的嵌入能适应各种下游任务，上游模型被设计为一个多任务学习的 DLRM 结构。该网络包含预测点击、点赞等分类任务和预测观看时长等回归任务。为了强迫模型学习长期稳定的兴趣，而不是依赖短期行为，作者在训练时移除了最近 24 小时的用户行为数据。这不仅让模型专注于长期模式，还提高了模型对部署延迟的鲁棒性。

#### 5.1.3 用户编码器

用于将上述 200 个聚合特征进一步压缩为紧凑的嵌入，以便下游存储和使用。这一步有点像 Rankmixer 的 Token-mixing 操作。作者将输入张量转置为  $[D, N]$ （Embedding Dimension, Number of Features），并在 Token 维度上应用神经网络。这促进了不同特征嵌入之间的参数共享。接着使用 Funnel Transformer 结构，逐层在 Token 维度上进行池化，逐步减少特征数量，最终将 200 个输入压缩为 58 个嵌入向量。为了进一步节省存储，输出时使用了 4-bit 量化。

## 6 多模态 + 长序列

### 6.1 快手：MISS

之前的超长序列主要是用在 Ranking，召回阶段难以利用超长序列。除此之外，现有的召回方法往往基于 ID 进行检索。出于上述的 gap，快手的 MISS [6] 基于 TDM 召回架构，简单提一下 TDM，TDM 是将语义相似的物品在树结构上聚集在同一子树，这使得索引结构更具有物理意义。叶子节点是真实物品的 embedding，非叶子节点的 embedding 就是其所有叶子节点的 Mean Pooling。

MISS 引入两个 GSU，一个是基于 ID 交互信息的 GSU，通过 Target Attention 计算候选节点和历史行为序列的相关性分数。还有一个 GSU 是基于多模态 embedding，直接用内积计算候选节点和历史行为序列相关性。

这里在多提一句，字节的 LongRetriever [10] 也是在召回阶段引入长序列，它没有 TDM 这样天然可以作为 Target 的结构，于是采取了 Interest Selection+Multi-Context 来执行检索。在兴趣筛选这

部分, 模型首先需要预测用户当前可能感兴趣的类目, 然后采取“Random in Top”策略, 它根据用户历史行为的时间衰减因子, 计算用户对每个类目的 score。在 Multi-Context 这部分, 对于选出的每一个兴趣, 运行一次 User Encoder, 生成一个该兴趣下的用户向量。将物品库按类目切分成多个子库, 用户向量只在对应的类目子库中进行检索, 最终将各路检索结果合并。由于是在召回阶段使用长序列, 而且训练函数涉及到 in-batch 负采样, 如果简单地使用筛选出的子序列  $S_u$  (例如全是“食品”类行为) 去预测正样本 (“食品”), 而负样本是随机的其他类目 (如“服装”), 模型会“偷懒”。它只需要判断类目是否匹配 (Food vs. Clothes), 而不需要学习用户具体喜欢哪个食品, 导致离线指标虚高但线上无效。因此, LongRetriever 重构了 Mini-batch 的采样方式, 确保在训练时, 对于给定的子序列  $S_u$ , 其对应的正样本和负样本都共享同一个类目。这迫使模型在特定兴趣领域内进行区分, 而不是简单地做类目匹配。

## 6.2 美团 DMGIN

美团本地生活场景 [13] 发现: 用户的点击后行为序列中包含大量重复或高度相关的实体 (例如同一家店铺), 这些实体可以通过店铺名称和图片等一致的标识符来识别, 因此, 用户的兴趣在很大程度上是由这些多模态内容 (描述和图片) 塑造。于是, DMGIN 就像利用 MLLM 对这些高度相关的店铺进行分组, 将数万条原始交互重组为数百个兴趣组。这样既能利用多模态信息, 又几乎不增加额外的计算开销。

为了实现有效的聚类, 首先需要对其的鲁棒的多模态表征。DMGIN 预训练一个 CLIP, 将文本和图片进行对比学习。然后为所有的店铺生成多模态 embedding, 基于 embedding 的相似度进行 Kmeans 聚类。通过聚类, 将用户数万次交互的长序列, 映射并重组为数百个兴趣簇。

为了缓解因分组带来的信息损失问题, 论文提出了两种补偿策略。统计组内各类行为 (如点击、下单、收藏) 的次数、最大/平均停留时间以及平均消费金额。第二个补偿策略是使用组内 Transformer 来建模组内不同时间戳、地点和行为类型的演变轨迹, 捕捉细粒度的兴趣动态。

## 6.3 阿里: MUSE

MUSE [14] 通过实验发现: 在 GSU 阶段, 简单的多模态余弦相似度就足够了, 复杂的机制带来的提升微乎其微; 而在 ESU 阶段, 显式的多模态序列建模和 ID-多模态融合能带来显著提升。基于上述洞察, 作者提出了 MUSE 框架, 其核心设计原则是: “GSU 追求简单, ESU 追求精细”。

多模态增强 ESU, 在精排阶段, MUSE 通过两个模块深度融合多模态信息:

SimTier [11]: 输入是 GSU 输出的相似度分数序列  $R$ 。操作: 将相似度分数 (区间  $[-1, 1]$ ) 划分成  $N$  个桶 (Tiers), 统计落在每个桶里的行为数量, 生成一个直方图向量。该模块的作用是显式地告诉模型 “用户历史中有多少个非常相关的物品”、“有多少个比较相关的物品”, 捕捉整体的语义相关性分布。这个 SimTier 有一篇单独的论文。

SA-TA, 由于传统的 Target Attention 依赖 ID 计算权重, 对长尾物品不准。因此将多模态相似度  $R$  融合进 Attention 的计算中。公式:  $\alpha^{Fusion} = \gamma_1 \alpha^{ID} + \gamma_2 R + \gamma_3 (\alpha^{ID} \odot R)$ 。利用多模态语义相似度来校准 ID Attention, 特别是当 ID 训练不充分的长尾物品时, 语义信息能提供 “保底” 的关联度判断。

为了在淘宝展示广告上线 100k 长度的序列模型, MUSE 采用了异步读取策略: 解耦: 将 GSU 的检索过程与主链路解耦。GSU Server 可以和召回并行工作, 将检索到的 Top-K 行为和排序模型需要的特征 embedding 缓存到 GPU 显存中。

## 参考文献

- [1] Cao, Y., Zhou, X., Feng, J., Huang, P., Xiao, Y., Chen, D., and Chen, S. Sampling is all you need on modeling long-term user behaviors for ctr prediction, 2022.
- [2] Chai, Z., Ren, Q., Xiao, X., Yang, H., Han, B., Zhang, S., Chen, D., Lu, H., Zhao, W., Yu, L., Xie, X., Ren, S., Sun, X., Tan, Y., Xu, P., Zheng, Y., and Wu, D. Longer: Scaling up long sequence modeling in industrial recommenders, 2025.
- [3] Chang, J., Zhang, C., Fu, Z., Zang, X., Guan, L., Lu, J., Hui, Y., Leng, D., Niu, Y., Song, Y., and Gai, K. Twin: Two-stage interest network for lifelong user behavior modeling in ctr prediction at kuaishou, 2023.

- 
- [4] Chen, Q., Pei, C., Lv, S., Li, C., Ge, J., and Ou, W. End-to-end user behavior retrieval in click-through rate prediction model, 2021.
  - [5] Guan, L., Yang, J.-Q., Zhao, Z., Zhang, B., Sun, B., Luo, X., Ni, J., Li, X., Qi, Y., Fan, Z., Wang, H., Chen, Q., Cheng, Y., Zhang, F., and Yang, X. Make it long, keep it fast: End-to-end 10k-sequence modeling at billion scale on douyin, 2025.
  - [6] Guo, C., She, J., Cai, K., Wang, S., Hu, Q., Luo, Q., Gai, K., and Zhou, G. Miss: Multi-modal tree indexing and searching with lifelong sequential behavior for retrieval recommendation, 2025.
  - [7] Hatefi, A., Vu, X.-S., Bhuyan, M., and Drewes, F. Cformer: Semi-supervised text clustering based on pseudo labeling. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (New York, NY, USA, 2021), CIKM '21, Association for Computing Machinery, p. 3078–3082.
  - [8] Lyu, W., Tyagi, D., Yang, Y., Li, Z., Somani, A., Shanmugasundaram, K., Andrejevic, N., Adeputra, F., Zeng, C., Singh, A. K., Ransan, M., and Jain, S. Dv365: Extremely long user history modeling at instagram, 2025.
  - [9] Qi, P., Zhu, X., Zhou, G., Zhang, Y., Wang, Z., Ren, L., Fan, Y., and Gai, K. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction, 2020.
  - [10] Ren, Q., Chai, Z., Xiao, X., Zheng, Y., and Wu, D. Longretriever: Towards ultra-long sequence based candidate retrieval for recommendation, 2025.
  - [11] Sheng, X.-R., Yang, F., Gong, L., Wang, B., Chan, Z., Zhang, Y., Cheng, Y., Zhu, Y.-N., Ge, T., Zhu, H., Jiang, Y., Xu, J., and Zheng, B. Enhancing taobao display advertising with multimodal representations: Challenges, approaches and insights, 2024.
  - [12] Si, Z., Guan, L., Sun, Z., Zang, X., Lu, J., Hui, Y., Cao, X., Yang, Z., Zheng, Y., Leng, D., Zheng, K., Zhang, C., Niu, Y., Song, Y., and Gai, K. Twin v2: Scaling ultra-long user behavior sequence modeling for enhanced ctr prediction at kuaishou. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management* (Oct. 2024), CIKM '24, ACM, p. 4890–4897.
  - [13] Wei, Z., Xie, Q., and Liu, Q. Dmgin: How multimodal llms enhance large recommendation models for lifelong user post-click behaviors, 2025.
  - [14] Wu, B., Yang, F., Chan, Z., Gu, Y.-R., Feng, J., Yi, C., Sheng, X.-R., Zhu, H., Xu, J., Ye, M., and Zheng, B. Muse: A simple yet effective multimodal search-based framework for lifelong user interest modeling, 2025.
  - [15] Zhou, G., Mou, N., Fan, Y., Pi, Q., Bian, W., Zhou, C., Zhu, X., and Gai, K. Deep interest evolution network for click-through rate prediction, 2018.
  - [16] Zhou, G., Song, C., Zhu, X., Fan, Y., Zhu, H., Ma, X., Yan, Y., Jin, J., Li, H., and Gai, K. Deep interest network for click-through rate prediction, 2018.