清华大学学报（自然科学版）
J Tsinghua Univ (Sci & Technol)

Scalable Token-based Ranking Models: 比较综述

算法小小怪下士

2025 年 12 月 19 日

摘要

随着 Scaling Law 在推荐系统领域的渗透，架构设计正从传统的 DLRM 向基于 Token 的生成式范式演进。本文对 RankMixer [7]、OneTrans [6]、HHFT [5] 和 STORE [4] 代表性模型进行了深度综述。这些工作的核心目标是一致的：**即在计算资源受限与 Scaling Law 的大势下，如何解决高维异构推荐特征的“碎片化”与“语义对齐”难题，以最大化模型的表达能力**。

本文从四个关键维度对上述模型展开系统性探讨：

(1) **Feature Processing**：在保留特征语义独立性与最大化 GPU 的 MFU 之间取得平衡；

(2) **Information Routing**：在 Attention 的计算冗余与高效性之间寻找最优解；

(3) **Interaction Paradigm**：解决不同特征子空间的异质冲突；

(4) **Scaling Strategy**：分析模型参数量、序列长度、模型宽度及样本数在 Scaling 过程中的收益。

1 特征处理 (Feature Processing)

推荐系统通常拥有数百个特征。如果将每个特征视为独立 Token，会导致计算碎片化，MFU 极低；若合并为单一超长 Token（如 DLRM 的 MLP），则会丧失特征子空间的区分度。如何平衡二者?

1.1 RankMixer: 语义分组

RankMixer 通过人工定义的语义边界，将特征压缩为数量适中且维度对齐的 Token。依赖先验知识将语义相近特征归组，拼接后进行**定长切分**并投影。

这里有一个问题，为什么要进行定长切分呢？如果不定长切分：不同组拼接后的长度不一，导致输入 Tensor 是参差不齐的，无法堆叠成 $[Batch, Token\_Num, Input\_Dim]$ 的矩形张量，必须使用 For 循环处理，打断 GPU 流水线。

RankMixer 方案：强制切分为 T 个长度为 d 的片段，形成标准的矩形张量 $[Batch, T, d]$，利用共享投影矩阵实现一次性 GEMM 运算，最大化 MFU。Rankmixer 论文中没有特别强调 Projector 是否是共享的，但感觉按照表述是共享的。

1.2 OneTrans: 全自动切分

首先，必须要提到的 OneTrans 的重大意义是统一序列建模与特征交叉。序列建模（如 DIN、LONGER）和特征交叉（如 Wukong、Rankmixer）通常是独立的，一般是 encode-then-interaction 范式，先做序列建模学习用户兴趣表示，再与非序列特征 concat 进行特征交叉。

这样的范式有两方面的局限性：一方面限制了双向信息交换, 比如 static/context 的非序列特征难以影响序列建模的兴趣表示。这一点和美团发布的 HoMER [1] 动机是一致的。该论文指出推荐系统中的特征异质性：序列侧信息较为有限，导致兴趣表达较粗粒度，而非序列特征信息丰富，表达更细致。HoMer 通过 panoramic sequence 方案，将行为序列的侧信息对齐为完整的非序列特征，使每个行为都包含丰富的上下文、用户/商品画像及交互特征，实现细粒度兴趣建模。

另一方面，分阶段训练不利于 Scaling。

出于上面的局限性，OneTrans 的统一序列建模与特征交叉，也因此这里特征的 token 化，我们需要分为两个方面讨论，分别是非序列特征和序列特征的 token 化。

1.2.1 非序列特征的 token 化

将所有非序列特征的 emb 拼接为超长向量，通过巨大 MLP 投影，再自动切分为固定数量的 Token。

虽然 OneTrans 也是通过一个巨大的 MLP 进行投影，但与传统的 DLRM 中的 MLP 输出一个向量有两点主要的不同。首先，OneTrans 紧接着通过 Split 将 MLP 的输出切分成了独立的 Token 向量组合。虽然这些 Token 的物理含义不像“用户 ID”那么明确，但它们依然是多个独立的特征载体，而不是一个单一的混合向量。

其次，OneTrans 和传统 DLRM 中特征之间的交互方式也不一样。传统 DLRM 主要是靠 MLP 的非线性能力去“猜”特征之间的高阶组合。这是一种隐式且静态的交互。高频特征的梯度很容易在反向传播中淹没长尾特征的梯度。OneTrans：生成的非序列 Token 随后进入了 Transformer Block，并且 OneTrans 为它们分配了独立的 Q/K/V 投影权重和独立的 FFN 权重。Token 之间通过 Attention 机制进行交互。这是一个显式且动态的过程。即使某个 Token 包含了高频信息，Attention 机制允许其他 Token 选择性地“关注”或“忽略”它，从而保留了长尾特征被独立激活的可能性。

1.2.2 序列特征的 **token** 化

假设总共有 n 种行为序列，对于第 i 种行为序列的所有 item，会用一个相同的 MLP，来处理 concat 后的 Item 信息。

在融合不同的行为序列的时候，可以使用 **Timestamp-aware**。按照时间顺序交错所有事件，并添加序列类型指示符。

在融合不同的行为序列的时候，也可以使用 **Timestamp-agnostic**。按照事件重要性对序列排序，如购买 → 加购 → 点击，并在序列中间插入可学习的 SEP token

论文通过实验证明，Timestamp-aware 策略会优于 Timestamp-agnostic 策略。

1.3 **HHFT: 语义分区**

HHFT 明确说了将特征显式划分为 User, Item, Context，User Behavior 等 Block，并在独立的 Embedding 空间内维护。通过这个方式更加强调特征的领域知识，防止跨域特征混淆。

1.4 **STORE: 语义量化与旋转**

为了解决 ID Embedding 的稀疏和低秩问题，STORE 重构底层表示。对高维稀疏 ID 使用 OPMQ 量化为 Semantic ID；对低维特征使用正交旋转。

首先是**高基数稀疏特征语义 Token 化**：通过将特征分解为紧凑的稳定语义 Token 集合，解决特征异构性与稀疏性问题。以 Item ID 为例，为了更高效的将编码为紧凑且并行的 SID，本文提出一个 OPMQ 网络，对每个 item，OPMQ 网络会使用 K 个专家来将预训练的 Item emb 编码成 K 个隐藏表示，对每个隐藏向量，找到离他最近邻的码字作为最终的 codebook 向量。同时端到端的训练这个网络，最小化原始 emb 与解码出的量化向量之间的重建误差。在损失函数中引入了 Orthogonal Regularization 损失，从而强制不同专家关注的方面是多元、非冗余。

然后是**低基数静态特征进行正交旋转变换**。通过正交旋转构成的子空间，促进更高效的特征交互。对低基数静态特征（比如年龄、性别），用其最原始的 emb，为了提升效率，按照领域知识进行人工语义分组，这些特征被分为 K 个语义组，每个组包含几个特征，对于每个特征 group 用 MLP 来学习组内特征交互。

为了更好的在高维空间进行特征交互，STORE 对语义组各自进行正交变换，引入正交矩阵的 Diversity Regularization。每个分组的正交矩阵会更新，但它不是跟着主网络一起做普通的“端到端反向传播”更新的。The rotation matrices and the parameters of the main network are alternative optimized。先固定正交矩阵，更新 main network，然后再固定 main network，反向传播更新这个正交矩阵，这必须强制正交矩阵回到正交群（Stiefel Manifold）上。

作者没有具体说怎么在反向传播回传梯度的时候更新正交矩阵，这里查了一些通常的实现方法包括：投影法 (Projected Gradient Descent)：先按梯度更新一步，然后用 SVD 分解或者 Gram-Schmidt 正交化把矩阵强行“拉回”成正交矩阵。流形优化：直接在黎曼流形上做优化（例如使用 Cayley 变换），保证每一步更新都在正交约束内。

2 不同特征块交互——信息路由

该部分不同工作的核心分歧在于是否使用 Attention，以及如何降低其计算成本。

2.1 RankMixer: Token Mixing

作者认为推荐特征异构性强，Attention 性价比低。改用类似 MLP-Mixer 的 Token Mixing。该方法利用 Reshape/Transpose 等内存操作代替 MatMul 计算，零 FLOPs 消耗实现信息混合，速度极快。具体张量操作如下。

1. 将 Token Embedding (D 维) 切分为 H 个头，维度变化：
   $[Batch, Tokens, Heads, D/Heads]$
2. 置换 Tokens 和 Heads 维度：
   $[Batch, Heads, Tokens, D/Heads]$
3. 拼接后两维：
   $[Batch, Heads, Tokens \times (D/Heads)]$

通过这种物理上的维度重组，让属于 Token A 的一部分信息和 Token B 的的信息“混合”起来。这里需要注意的一点是，实际操作的时候，config 中 Heads=Tokens，这样能够让经过维度变化后，依然能够与初始的 Tensor 维度一致，从而进行残差连接。

2.2 OneTrans: 统一 Causal Transformer

OneTrans 将序列与非序列特征拼接，输入 Attention，为了防止由于序列特征和非序列特征的值的范围和统计差异导致的训练不稳定性，使用 RMSNorm 进行预归一化。。使用 Attention 的好处是架构统一，可以直接复用 FlashAttn v2、混合精度训练和 KV Cache 优化技术。对于 KV Cache 的实现，具体来说对于一个用户的同一请求，序列特征是共享的，非序列特征是不共享的。因此使用了 KV cache 的两阶段，对于序列 token，缓存其 KV cache 和 attn_output；对于非序列 Token，利用缓存的 KV cache 进行 Cross Attention。

关于 Mask，这里对于 timestamp agonistic 组织的样本，序列特征中通过 mask 过滤掉同一个 item 的弱意图的行为。对于非序列特征，每个非序列 token 可以关注整个序列历史和非序列 token。

最后需要提一下，OneTrans 提出的金字塔式结构。在 Attention 计算时，逐步减少序列的 Query Token 的数量，而 Key 和 Value 仍然在整个序列上计算。这样的方式，一方面可以通过逐步减少 Query Token 的数量，迫使模型将长序列信息逐步浓缩到非序列 Token 中。另一方面，这样的结构可以提高训练和推理速度。

2.3 HHFT: 异构 Transformer + HiFormer

采用层级结构。底层异构 Transformer 处理 Pairwise 交互，顶层 Hiformer Layer 处理高阶组合。

实验证明，仅仅堆叠 Transformer 层（低阶交互）的效果不如“Transformer + Hiformer”（低阶 + 高阶）的效果好。

2.3.1 HiFormer [2]

由于推荐领域的特征是异构的，所以设计异构 Attention 进行特征间的交互，增大参数量，并且提升了模型效果。由于独立的 QKV 太多了，所以通过 LoRA 来降低训练开销。通过实验分析 Attention Maps，发现原始 Self Attention 注意力集中在对角线上，即关注同构的特征较多，但是 HiFormer 的注意力权重分布更加均衡。

2.4 STORE: 稀疏高效 Attention [3]

Attention 中的 Query 利用 MoE 的 TopK Routing 策略选择最相关的 K-V Pair 对，而非全量计算。好处是可以降低复杂度，并通过稀疏性过滤低贡献度的 Token 噪声。

$$\mathrm{MoBA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{Softmax}\left(\mathbf{QK}[\mathit{Ind}]^T\right)\mathbf{V}[\mathit{Ind}],$$
$$\mathit{Ind}_i = [(i-1) \times B + 1, i \times B]$$

3 不同特征块的交互——如何保证特征语义独立性

3.1 RankMixer: Per-token FFN → MoE

考虑到不同特征（如热门 ID vs 长尾 ID）分布差异大，每个 Token 位置拥有独立的 FFN 参数。为了高效 scaling，使用 Sparse MoE。

针对 Sparse MoE 负载不均和专家训练不足的问题，使用 Dense activation Training 和 Sparse activation Infer，定义 h_{train} 和 h_{infer} 双路由。稀疏正则化项 $\mathcal{L}_{reg}$ 仅作用于 h_{infer}。使用 ReLU($h(x)$) 门控替代 Softmax Top-k，ReLU 的特性使得低权重的专家输出自然为 0，从而实现了真正的动态稀疏推理

3.2 OneTrans: 混合参数化

针对推荐数据的特殊性（序列特征同质化高，画像特征异质性高），设计了混合参数化处理 QKV 和 FFN 的权重，从而兼顾归纳偏置与异构特性。。对于所有序列特征共享一组 QKV 和 FFN 的权重，这种共享机制提高了计算效率，并且促进跨时间步的泛化能力。对于每个非序列特征，都有其特定的 QKV 和 FFN 权重，保留了非序列特征的独特语义含义，使模型能更好的处理静态特征。

3.3 HHFT: 物理参数隔离

强制不同特征 Block 使用完全独立的 QKV 投影矩阵，防止语义混淆。

3.4 STORE: 数学空间隔离

交互前对特征乘上可学习的正交矩阵。利用几何旋转使子空间在数学上保持正交，无需物理隔离即可自然区分不同组的特征。

4 Scaling Law 与总结

各模型在扩大规模时的侧重点不同：

- **RankMixer:** 拼参数容量与硬件效率。引入 MoE 以在不增加 FLOPs 下扩展参数至 >1B。
- **OneTrans:** 拼序列长度。利用 KV Cache 处理超长行为序列收益最大。
- **HHFT:** 拼模型宽度。Embedding Size 的收益大于层数。
- **STORE:** 拼样本效率。通过 Semantic ID 打破”One-Epoch” 瓶颈，使模型能通过多轮训练持续提升。

表 1 四种 Scalable Token-based Ranking 模型对比总结

维度	RankMixer (字节)	OneTrans (字节)	HHFT (阿里)	STORE (阿里)
特征输入	人工语义分组 (Group-wise)	自动大一统切分 (Auto-split)	语义物理分区	语义量化 (SIDs)
核心架构	去 Attention (Mixing)	Causal Attention	异构 Attn + 高阶层	稀疏 Attn
交互参数	全独立 (Per-token/MoE)	混合 (Mixed)	物理隔离 (Block-specific)	正交旋转 (Rotation)
扩展方向	硬件效率 & 参数规模	序列长度	模型宽度	训练轮数

参考文献

[1] Chen, S., Cui, J., Xu, Z., Zhang, F., Fan, J., Zhang, T., and Wang, X. Homer: Addressing heterogeneities by modeling sequential and set-wise contexts for ctr prediction, 2025.

[2] Gui, H., Wang, R., Yin, K., Jin, L., Kula, M., Xu, T., Hong, L., and Chi, E. H. Hiformer: Heterogeneous feature interactions learning with transformers for recommender systems, 2023.

[3] Lu, E., Jiang, Z., Liu, J., Du, Y., Jiang, T., Hong, C., Liu, S., He, W., Yuan, E., Wang, Y., Huang, Z., Yuan, H., Xu, S., Xu, X., Lai, G., Chen, Y., Zheng, H., Yan, J., Su, J., Wu, Y., Zhang, N. Y., Yang, Z., Zhou, X., Zhang, M., and Qiu, J. Moba: Mixture of block attention for long-context llms, 2025.

[4] Xu, Y., Fan, C., Hu, J., Zhang, Y., Xiaoyi, Z., and Zhang, J. Store: Semantic tokenization, orthogonal rotation and efficient attention for scaling up ranking models, 2025.

[5] Yu, L., Zhang, W., Zhou, S., Zhang, T., Zhang, Z., and Ou, D. Hhft: Hierarchical heterogeneous feature transformer for recommendation systems, 2025.

[6] Zhang, Z., Pei, H., Guo, J., Wang, T., Feng, Y., Sun, H., Liu, S., and Sun, A. Onetrans: Unified feature interaction and sequence modeling with one transformer in industrial recommender, 2025.

[7] Zhu, J., Fan, Z., Zhu, X., Jiang, Y., Wang, H., Han, X., Ding, H., Wang, X., Zhao, W., Gong, Z., Yang, H., Chai, Z., Chen, Z., Zheng, Y., Chen, Q., Zhang, F., Zhou, X., Xu, P., Yang, X., Wu, D., and Liu, Z. Rankmixer: Scaling up ranking models in industrial recommenders, 2025.