LazUnirec：多场景工业推荐中的统一检索与排序

Xiaodong Li、Junfeng Zhang、Shichao Nie、Ziyue Guo、Yiming Liu、Xinxin Xu、Xurong Li、Jinyuan Dang、Zhanming Ou、Jianan Yan*、Tao Luo、Xiaoyi Zeng 阿里巴巴国际数字商业集团，中国北京 {limingxi.lxd, sichu.zjf, nsc383441, guoziyue.gzy,lym559243,xxx558853, xurong.li, dangjinyuan.djy, zhanming.ozm, jianan.yjn, luotao.lt,yuanhan}@alibaba-inc.com

arXiv:2611.32590v1 [cs.IR] 24 Nov 2026

摘要

现代工业推荐系统通常将检索和排序划分为两个串联的独立阶段，导致优化目标不一致、用户序列重复编码以及计算资源浪费。以往的研究探索了使用 Transformer 同时建模检索和排序的方法。然而，这些研究仍面临两大局限：（1）由于检索和排序共用单一 Transformer 骨干网络，当两个任务的梯度方向不一致时，会产生梯度冲突。（2）发生梯度冲突时，模型无法确定可靠的梯度更新方向。为此，我们提出了一种面向多场景工业推荐的统一检索与排序方法，称为 **LazUnirec**。具体而言，我们在 Transformer 骨干网络上构建任务共享参数和任务私有参数，以缓解梯度冲突。其中，对于注意力组件，我们提出了 LoRA调制注意力，使检索和排序能够共享注意力参数，同时保留任务私有参数。对于前馈网络（FFN），我们提出了任务自适应稀疏混合专家（MoE），通过构建任务共享专家和任务私有专家来缓解梯度冲突。此外，我们提出了一种不确定性估计方法，用于评估检索和排序梯度的可靠性，从而在发生梯度冲突时引导模型采用更可靠的梯度更新方向。此外，从排序到检索的知识蒸馏进一步实现了跨任务知识迁移。大量离线实验和在线A/B测试证明了 LazUnirec 的有效性和优越性。LazUnirec 已部署于线上多场景工业推荐系统，目前为平台带来了显著收益。

CCS 概念

• **信息系统** →**推荐系统**。

关键词

推荐系统、统一检索与排序、多场景推荐

1 引言

现代工业推荐系统通常将检索和排序分为两个独立级联的阶段。检索 [4, 18, 19, 31] 旨在从物品语料库中高效识别出用户可能互动的物品子集，而排序 [13, 17, 36] 则预测多种概率（即，点击率、转化率）

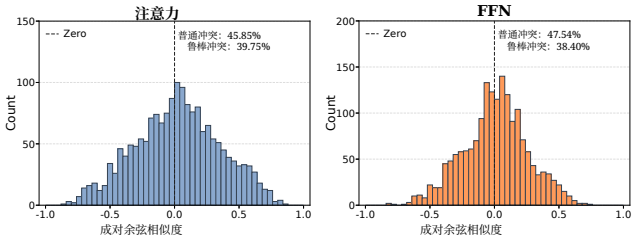


图1：Transformer骨干网络中检索与排序的梯度冲突百分比分布统计，包括注意力和 FFN 组件。

以刻画用户对候选物品的精确偏好。这种划分长期以来一直是工业推荐中的主导范式，因为它有效平衡了高效搜索与准确分类。然而，这两个阶段通常基于相同的用户序列进行编码，同时采用相互独立的参数和优化目标。因此，这种级联设计引入了不一致的优化目标、对用户序列的冗余编码，以及不必要的计算资源消耗。随着工业系统处理更长的用户序列并服务于多个推荐场景，这些成本变得越来越显著。

与此同时，检索和排序都越来越多地采用 Transformer [25] 架构。一方面，检索利用 Transformer 对历史交互进行编码，并提取用户偏好以生成候选。一些研究 [6, 22, 27,29, 33, 35]进一步采用生成式范式，使用语义ID表示每个物品，以进行自回归预测。另一方面，一些排序方法 [16, 34] 采用 Transformer，同时进行用户序列建模和特征交互，并从深度、宽度和序列长度的扩展中获益。尽管它们的输出和预测目标不同，但这两个阶段都依赖于对相同用户交互序列进行编码。这种架构上的趋同自然促使人们在单一 Transformer骨干网络中统一检索与排序，共享用户序列编码，并为两者的监督信号共同优化这一共享骨干网络提供机会。

近期研究 [2, 15, 24, 30] 探索了统一检索并在单一Transformer骨干网络内进行排序。具体而言，UniPin- Rec [15]采用掩码动作建模，以支持使用兼容输入表示的检索与排序，并在不同阶段复用用户序列的KV缓存。UniR^2 [30] 将用户上下文、语义ID生成和排序特征组织为一个仅解码器序列，并通过排序侧LoRA在优化隔离下实现任务特定的适配。OneTrans-V2 [2]进一步扩展了

* 通讯作者。

这种统一化方案，将其应用于检索、预排序和精排序，在共享用户上下文的同时结合各阶段特定的计算。这些研究表明，统一模型能够复用用户侧编码并降低计算成本，同时保留检索与排序各自不同的功能。

尽管取得了令人鼓舞的进展，但要让检索与排序在共享的Transformer 中协同工作，仍面临严重的梯度冲突 [1, 8, 21] 问题，这可归因于以下两点局限：（1）检索与排序具有不同的优化目标，这可能导致共享的 ***Transformer*** 骨干网络中出现相互冲突的梯度方向。如图 1 所示，我们对 Transformer 骨干网络中检索与排序的梯度冲突比例分布进行了统计分析，其中包括注意力和前馈网络（FFN）组件。我们发现，注意力和 FFN 组件中的普通冲突（即，cos ＜ 0）比例分别为 45.85% 和 47.54%。而相应的鲁棒冲突（即，cos ＜ -0.05）比例仍分别为 39.75% 和 38.40%，表明注意力和 FFN 组件中都存在显著的梯度冲突。（2）发生梯度冲突时，如何确定可靠的梯度更新方向？换言之，可靠的梯度更新方向 [3, 7, 28] 可以理解为能够反映检索与排序整体优化趋势、且不受梯度冲突影响的方向。这一问题在多场景推荐中尤为突出，因为不同场景的交互模式和监督信号各不相同。因此，在缓解梯度冲突的同时确定可靠的更新方向，仍是统一检索与排序面临的一项重大挑战。

为解决上述局限性，我们提出了一种面向多场景工业推荐的统一检索与排序方法，称为**LazUnirec**。具体而言，我们在Transformer骨干网络上构建任务共享参数和任务私有参数，以缓解梯度冲突。其中，对于注意力组件，我们引入LoRA调制注意力，共同学习检索和排序的任务共享注意力参数与任务私有低秩适配。对于FFN组件，我们提出任务自适应稀疏混合专家（MoE），将任务私有专家与任务共享专家相结合。此外，我们提出不确定性估计，基于历史方向稳定性评估检索和排序的梯度可靠性，从而在发生梯度冲突时确定更可靠的梯度更新方向。除此之外，知识蒸馏将候选偏好分布从排序传递至检索，建立显式的知识传递路径。为实现高效的在线部署，我们进一步引入KV缓存、算子融合、低精度服务和跨场景批处理。

我们的主要贡献总结如下：

- 我们提出了 LazUnirec，用于多场景工业推荐中的统一检索与排序。LoRA 调制的注意力机制和任务自适应稀疏混合专家通过结合任务共享参数与任务私有参数，缓解了梯度冲突。
- 我们引入不确定性估计，在发生梯度冲突时引导模型朝着更可靠的方向进行梯度更新。此外，知识蒸馏进一步建立了从排序到检索的显式知识传递路径。
- 我们通过广泛的离线实验验证了 LazUnirec 的有效性。在线A/B测试进一步证明 LazUnirec 为平台带来了可观的收入增长（商品交易总额最高提升 4.01%）。LazUnirec 已部署于真实的多场景推荐系统中，为平台带来了显著收益。

2 问题定义

在多场景工业推荐中，我们的目标是使用同一个Transformer骨干网络，同时完成检索、排序以及多场景预测。具体而言，**检索**阶段旨在有效检索出用户可能与之交互的物品子集。而**排序** 阶段旨在预测多个概率（即点击率、转化率），以反映用户对目标（候选）物品的偏好。

形式化地，令 $\mathcal{U}$、$\mathcal{V}$ 和 $\mathcal{S}$ 分别表示用户集合、物品集合和场景集合。用户 $u_i \in \mathcal{U}$ 的历史交互序列可表示为 $\mathcal{H}_i = \{v_j,\ s_j\}_{j=1}^{|\mathcal{H}_i|}$，其中 $v_j \in \mathcal{V}$ 表示发生交互的物品，$s_j \in \mathcal{S}$ 表示交互发生的场景。此外，我们使用 $\mathcal{Y}$ 表示排序阶段多目标预测的真实标签（即，$y^{ctr} \in \{0,\ 1\}$ 和 $y^{cvr} \in \{0,\ 1\}$）。例如，给定一个观测交互对（$u_i,\ v_j$），我们使用 $y_{ij}^{ctr} \in \{0,\ 1\}$ 表示用户 u_i 是否点击物品 v_j。此外，排序阶段的目标（候选）物品可表示为 $\mathcal{T}_i = \{v_j | v_j \in \mathcal{V}\}_{j=1}^{|\mathcal{T}_i|}$。

因此，对于用户 u_i的训练实例，召回阶段采用如下采样 Softmax 目标：

$$\mathcal{L}_{\text{retrieval}} = -\frac{1}{|\mathcal{P}_i|} \sum_{j \in \mathcal{P}_i} \log \frac{\exp(r_{ij})}{\exp(r_{ij}) + \sum_{n \in \mathcal{N}_{ij}} \exp(r_{in})}, \quad (1)$$

其中，$r_{ij} = \boldsymbol{u}_i^\top \boldsymbol{v}_j$ 表示用户 u_i 与物品 v_j之间的匹配分数，$\mathcal{P}_i$表示正样本物品的索引集合，$\mathcal{N}_{ij}$表示正样本物品 v_j对应的采样负样本物品索引集合。

而排序阶段则最小化标准的负对数似然，其形式如下：

$$\mathcal{L}_{\text{ranking}} = -\sum_{xtr}^{ctr,cvr} [y_{ij}^{xtr} \log(\hat{y}_{ij}^{xtr}) + (1 - y_{ij}^{xtr}) \log(1 - \hat{y}_{ij}^{xtr})], \quad (2)$$

其中，y_{ij}^{xtr}表示用户 u_i对物品 v_j的真实标签，$\hat{y}_{ij}^{xtr}$表示 XTR（即点击率、转化率）的预测概率。

3 方法

本节介绍了我们提出的 LazUnirec，如图 2 所示，该方法为多场景工业推荐实现了统一检索与排序方案。具体而言，LazUnirec 主要由五个部分组成：1）**统一序列词元化**旨在将用户交互序列标记、各场景的召回与排序标记整合到统一序列中；2）**LoRA调制注意力**从 LoRA [10] 中汲取灵感，通过为检索和排序构建任务私有参数，在共享底层注意力参数的同时保持任务特异性；3）**任务自适应稀疏混合专家**通过构建任务共享专家和任务私有专家，进一步缓解梯度冲突；4）**不确定性估计**旨在梯度冲突发生时确定可靠的梯度更新方向；5）**知识蒸馏**建立从排序到检索的显式知识传递路径。各组件的详细信息将在以下部分介绍。

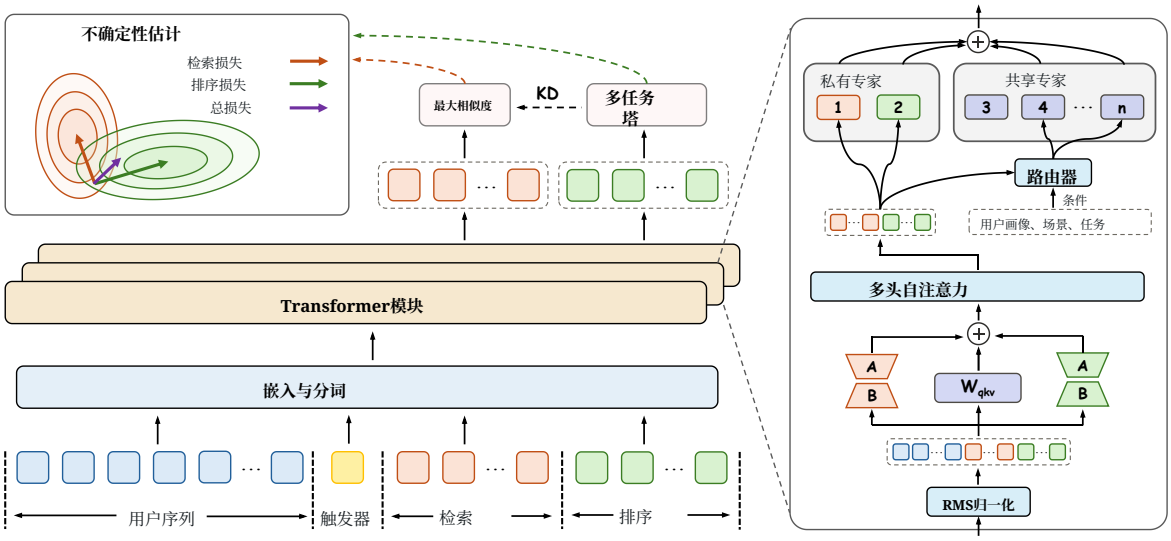


图2：LazUnirec 的整体框架。

3.1 统一序列分词

为了统一不同场景下的检索与排序，我们将用户 u_i的输入组织成统一序列 $\boldsymbol{X}_i = [\boldsymbol{H}_i \| \boldsymbol{g}_i \| \boldsymbol{R}_i \| \boldsymbol{T}_i]$，该序列由交互序列标记 $\boldsymbol{H}_i$、触发令牌 $\boldsymbol{g}_i$、检索令牌 $\boldsymbol{R}_i$和排序令牌 $\boldsymbol{T}_i$组成。

3.1.1 用户序列令牌。 给定交互序列项 $(v_j,\ s_j) \in \mathcal{H}_i$，我们通过结合基于 ID 的（即，物品 ID）、基于侧信息的（即，物品的属性信息）和场景特征来构建序列词元 $\boldsymbol{h}_j$：

$$\begin{aligned} \boldsymbol{h}_j &= [\boldsymbol{e}_j^{id} \| \boldsymbol{e}_j^{side} \| \boldsymbol{e}_j^{scenario}], \\ \boldsymbol{H}_i &= [\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_{|\mathcal{H}_i|}], \end{aligned} \tag{3}$$

其中，$\|$ 表示拼接操作。

3.1.2 触发令牌。 在我们的多场景工业推荐中，当存在触发项时，场景被归类为基于触发项的场景，否则则为无触发项场景。因此，我们添加一个触发令牌 $\boldsymbol{g}_i$ 来区分这两类场景。具体而言，基于触发项的场景会对触发项进行编码，以获得 $\boldsymbol{g}_i$，与公式（3）类似。对于无触发项场景，则使用一个可学习占位符替代触发项。因此，触发令牌通过注意力汇总用户序列，并为构建后续检索令牌提供场景上下文。

*3.1.3*检索令牌。 我们引入多个可学习检索查询 $\{\boldsymbol{b}_k\}_{k=1}^{K}$，以捕捉多样化的用户偏好。每个查询都与触发表征结合，构建一个检索令牌 $\boldsymbol{r}_k$，其形式如下：

$$\begin{aligned} \boldsymbol{r}_k &= \boldsymbol{g}_i + \boldsymbol{b}_k, \\ \boldsymbol{R}_i &= [\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_K], \end{aligned} \tag{4}$$

这些召回令牌形成 $\boldsymbol{R}_i$ 并行提取多个用户偏好表征 [20]。

3.1.4 排序令牌。 工业日志通常会在同一请求中包含多个目标物品。传统工业推荐系统会为每个目标物品分别构建一个样本，导致对同一用户序列进行冗余编码。因此，我们将同一请求中的所有目标物品组织起来 [16]，以共享用户序列的存储、传输和编码。具体而言，对于每个目标物品 v_j，我们构建与式（3）类似的排序令牌 $\boldsymbol{t}_j$，如下所示：

$$\begin{aligned} \boldsymbol{t}_j &= [\boldsymbol{e}_j^{id} \| \boldsymbol{e}_j^{side} \| \boldsymbol{e}_j^{scenario}], \\ \boldsymbol{T}_i &= [\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_{|\mathcal{T}_i|}], \end{aligned} \tag{5}$$

因此，LazUnirec 的统一输入序列可表述如下：

$$\begin{aligned} \boldsymbol{X}_i &= [\boldsymbol{H}_i \| \boldsymbol{g}_i \| \boldsymbol{R}_i \| \boldsymbol{T}_i], \\ &= [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{|\mathcal{H}_i|} \parallel \boldsymbol{g}_i \parallel \boldsymbol{r}_1, \ldots, \boldsymbol{r}_K \parallel \boldsymbol{t}_1, \ldots, \boldsymbol{t}_{|\mathcal{T}_i|}], \end{aligned} \tag{6}$$

3.2 Transformer骨干网络

现有研究 [2, 15, 24, 30] 通过共享的 Transformer骨干网络统一检索与排序，但不同的优化目标可能会在共享参数上产生梯度冲突。因此，我们在注意力和前馈网络（FFN）组件中引入任务共享参数和任务私有参数。这些私有参数提供了额外的容量，以适应任务特定模式，在保留检索与排序之间知识共享的同时，缓解梯度冲突。

3.2.1 ***LoRA*调制注意力。** 我们将任务私有低秩适配 [10] 引入查询、键和值投影。具体而言，对于召回令牌 $\boldsymbol{R}_k$ 和排序令牌 $\boldsymbol{T}_j$，我们通过任务私有低秩残差增强共享的查询、键和值投影，如下所示：

$$\begin{aligned} \boldsymbol{p}_k^{\text{retrieval}} &= (\boldsymbol{W}_p + \Delta \boldsymbol{W}_p^{\text{ret}}) \boldsymbol{R}_k, \\ \boldsymbol{p}_j^{\text{ranking}} &= (\boldsymbol{W}_p + \Delta \boldsymbol{W}_p^{\text{rank}}) \boldsymbol{T}_j, \\ \Delta \boldsymbol{W}_p^{\tau} &= \boldsymbol{B}_p^{\tau} \boldsymbol{A}_p^{\tau}, \quad \tau \in \{\text{retrieval}, \text{ranking}\}, \end{aligned} \tag{7}$$

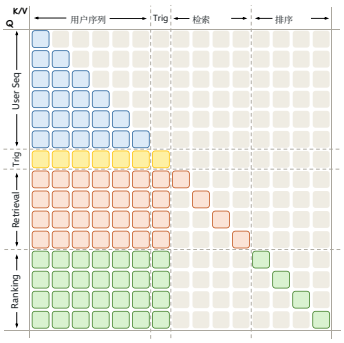


图3：注意力掩码示意图。

其中，$p \in \{Q、K、V\}$表示投影类型，A_p^τ和B_p^τ是任务私有低秩矩阵。用户序列令牌H_i和触发令牌g_i仅使用共享投影。总体而言，任务共享参数和任务私有参数均进行联合优化，使两个任务能够共享知识，同时适应各自不同的目标。

此外，我们引入结构化注意力掩码M_i，以控制统一序列X_i中的信息流，如图 3 所示。具体而言，用户序列令牌会关注自身和先前交互，而触发令牌会关注用户序列和自身。每个检索或排序令牌仅关注用户序列、触发令牌和自身，从而保持不同检索查询和排序候选项之间的隔离。

每个注意力头的注意力运算定义如下：

$$O_i = \mathrm{Softmax}\left(\frac{Q_i K_i^\top}{\sqrt{d_h}} + M_i\right) V_i, \tag{8}$$

其中，Q_i、K_i和V_i分别表示查询、键和值矩阵，d_h表示注意力头维度。通过 LoRA调制注意力，LazUnirec 联合学习任务共享的注意力参数，以及用于检索和排序的任务私有低秩适配，从而在适应任务特定目标的同时保留跨任务知识共享，以缓解梯度冲突。

3.2.2 任务自适应稀疏混合专家。 为进一步缓解 FFN 组件中的梯度冲突，我们引入一种任务自适应稀疏混合专家，将任务共享专家与任务私有专家相结合。具体而言，该混合专家由一个共享专家池和每个任务各自的一个任务私有专家组成，从而实现跨任务知识共享，同时保留任务特异性。

对于 LoRA 调制注意力生成的检索或排序令牌x，其任务私有专家始终处于激活状态。同时，可学习路由器根据条件信息选择共享专家，条件信息包括令牌表示、用户画像、场景和任务，具体可表示如下：

$$a^\tau = \mathrm{Router}(x, e_i^{profile}, e_i^{scenario}, e_\tau^{task}), \tag{9}$$

其中，a^τ表示任务$\tau \in \{\text{retrieval}, \text{ranking}\}$当前令牌对应的共享专家路由分数。

然后，路由器按如下方式选出排名前k的共享专家：

$$\begin{aligned} \mathcal{D}^\tau &= \mathrm{TopK}(a^\tau, k), \\ w_e^\tau &= \frac{\exp(a_e^\tau)}{\sum_{m\in\mathcal{D}^\tau}\exp(a_m^\tau)}, \quad e \in \mathcal{D}^\tau, \end{aligned} \tag{10}$$

其中，$\mathcal{D}^\tau$包含选定专家索引，w_e^τ表示为任务τ的当前 token 分配给共享专家e的归一化路由权重。我们的实现中将$k = 2$设为该值。

最终，MoE 将任务私有专家的输出与所选共享专家的加权输出相结合：

$$\mathrm{MoE}^\tau(x) = \mathrm{E}^\tau_{\text{private}}(x) + \sum_{e\in\mathcal{D}^\tau} w_e^\tau \, \mathrm{E}_e^{\text{shared}}(x), \tag{11}$$

因此，任务私有专家为任务特定模式提供专属容量，而选择性激活的共享专家则促进检索与排序之间的知识共享。路由器根据用户画像、场景和任务调整专家选择，同时通过稀疏激活控制计算成本。

3.3 不确定性估计

除了通过构建任务共享参数和任务私有参数来缓解梯度冲突外，我们还提出一种不确定性估计方法，用于评估检索和排序的梯度可靠性 [3, 7, 28]，从而在发生梯度冲突时，引导模型采用更可靠的梯度更新方向。具体而言，我们基于历史方向稳定性估计梯度不确定性，对稳定梯度赋予相对较高的权重，对波动较大的梯度赋予较低的权重。

对于场景$s \in \mathcal{S}$和任务$\tau \in \{\text{retrieval}、\text{ranking}\}$，令$\gamma_s^\tau$表示通过定期探测一小部分共享参数所获得的归一化梯度表示，其中归一化旨在将估计重点放在梯度方向而非其幅值上。我们维护指数移动平均（EMA）以估计方差，并将其转换为梯度证据：

$$\begin{aligned} (\sigma_s^\tau)^2 &= [\mathrm{EMA}((\gamma_s^\tau)^2) - (\mathrm{EMA}(\gamma_s^\tau))^2]_+, \\ e_s^\tau &= \min(e_{\max}, ((\sigma_s^\tau)^2 + \epsilon)^{-\frac{1}{2}}), \end{aligned} \tag{12}$$

其中，$[x]_+ = \max(x, 0)$，ϵ和$e_{\max}$用于确保数值稳定性。历史方差越小，表示梯度方向越稳定，所产生的证据也越强。

接着，我们汇总这些证据以估计不确定性，并据此确定损失权重：

$$\begin{aligned} \mathcal{E}_s^\tau &= \frac{1}{1 + \mathrm{Mean}(e_s^\tau)}, \\ \omega_s^\tau &= \frac{1 - \mathcal{E}_s^\tau}{(1 - \hat{\mathcal{E}}_s)_{\text{batch}}}, \end{aligned} \tag{13}$$

其中，Mean 表示对证据项取平均，$1 - \mathcal{E}_s^\tau$表示梯度可靠性，$(1 - \mathcal{E}_s)_{\text{batch}}$表示当前小批量中检索和排序观测值的平均可靠性。

最后，我们将可靠性权重应用于任务损失，得到不确定性加权目标函数：

$$\mathcal{L}_{\text{ue}} = \sum_\tau \mathrm{sg}(\omega_s^\tau)\mathcal{L}^\tau, \tag{14}$$

其中，$\mathcal{L}^\tau$表示检索损失或排序损失，sg 表示停止梯度，确保可靠性权重不参与梯度计算。

通过不确定性估计，具有更稳定历史方向的梯度在共享参数更新中获得更大的相对影响，从而在发生梯度冲突时，引导联合优化朝向更可靠的方向。

3.4 知识蒸馏

为进一步促进跨任务知识迁移，我们引入从排序到检索的列表级知识蒸馏。具体而言，排序分支捕捉细粒度的用户-物品交互，并在同一请求中提供目标物品之间的相对偏好信号。我们通过匹配候选偏好分布，将这些信号传输至检索任务。

对于目标物品 $\mathcal{T}_i$，我们通过将时间戳相同的候选物品分组，构建请求级候选列表。用于蒸馏的召回分数计算如下：

$$\tilde{r}_{ij} = \max_{1\le k\le K} \left(\pi_{ik} \boldsymbol{u}_{ik}^{\top} \boldsymbol{v}_j\right), \tag{15}$$

其中，$\boldsymbol{u}_{ik}$ 是用户兴趣表示（即，k号召回标记 $\boldsymbol{r}_k$ 的 Transformer 输出），π_{ik} 表示相应的兴趣概率。

令 $\boldsymbol{z}_i^{ctr}$ 收集候选列表上的排序点击率预测逻辑值，并令 $\tilde{\boldsymbol{r}}_i$ 收集相应的召回分数，我们构建教师分布和学生分布如下：

$$\begin{aligned} \boldsymbol{p}_i^{\text{ranking}} &= \text{Softmax}\left(\frac{\text{sg}(\boldsymbol{z}_i^{ctr})}{T_{\text{ranking}}}\right), \\ \boldsymbol{p}_i^{\text{retrieval}} &= \text{Softmax}\left(\frac{\tilde{\boldsymbol{r}}_i}{T_{\text{retrieval}}}\right), \end{aligned} \tag{16}$$

其中 T_{ranking}、$T_{\text{retrieval}}$ 是控制分布尖锐度的温度超参数，停止梯度会阻断经由排序教师的反向传播。

随后，我们按如下方式最小化从排序分布到检索分布的KL散度：

$$\mathcal{L}_{\text{kd}} = \sum_{v_j\in\mathcal{T}_i} p_{ij}^{\text{ranking}} \log \frac{p_{ij}^{\text{ranking}}}{p_{ij}^{\text{retrieval}}}, \tag{17}$$

知识蒸馏通过学生分支更新召回表示和兴趣概率，从而实现从排序到检索的显式知识迁移。

总体而言，LazUnirec 通过结合不确定性加权损失、知识蒸馏和混合专家正则化进行优化，可表述如下：

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{\text{ue}} + \lambda_{\text{kd}}\mathcal{L}_{\text{kd}} + \lambda_z\mathcal{L}_z \\ &= \text{sg}(\omega_s^{\text{retrieval}})\mathcal{L}_{\text{retrieval}} + \text{sg}(\omega_s^{\text{ranking}})\mathcal{L}_{\text{ranking}} \\ &\quad + \lambda_{\text{kd}}\mathcal{L}_{\text{kd}} + \lambda_z\mathcal{L}_z, \end{aligned} \tag{18}$$

其中 $\mathcal{L}_z$ 表示混合专家路由器 z 损失，λ_{kd} 和 λ_z 分别控制蒸馏与混合专家正则化的贡献。

3.5 在线部署

在在线推理中，我们部署两个独立的模型，分别用于召回和排序。具体而言，我们通过 KV缓存、算子融合、低精度服务和跨场景批处理对 LazUnirec 进行优化。这些优化通过联合处理来自不同场景的请求，减少重复的序列编码，降低计算和内存开销，并提高 GPU 利用率。

3.5.1 ***KV缓存。*** 我们在检索阶段构建用户KV缓存，存储用户序列逐层的键和值。在同一请求中，排序阶段会在执行其特定任务计算时复用该缓存。该缓存会在用户序列和场景条件保持不变时进一步跨请求复用。KV缓存支持跨任务和跨请求复用，减少冗余的用户序列编码。

3.5.2 **算子融合。** 为了减少内存流量和内核启动开销，我们将共享相同输入的一系列操作进行合并。具体而言，我们将共享的查询、键和值投影权重进行拼接，使三个投影能够通过一次矩阵乘法完成。在 SwiGLU（即，专家）中，我们融合门控/上投影、SiLU激活和逐元素乘法，以减少中间张量的物化。对于稀疏混合专家，将分配给同一专家的词元在 GPU内存中连续组织。我们对门控/上投影和下投影均应用分组GEMM（即，通用矩阵乘法），在每个分组中通过一次内核启动，跨多个专家执行相应的矩阵乘法。这种组织方式减少了单独的专家启动次数，并提高了 GPU计算的利用率。

3.5.3 **低精度服务。** 在整个 LazUnirec 中，我们采用 FP16 作为默认服务精度，并有选择地将 FP8 用于 SwiGLU 内部的矩阵乘法，包括门控/上投影和下投影。该配置旨在降低前馈网络执行的计算和内存开销，同时将 FP8量化限制在模型的特定部分。

3.5.4 **跨场景批处理。** LazUnirec 在在线推理中同时服务于多个场景。因此，我们聚合来自不同场景的在线请求，以提高 GPU 利用率。在极短时间窗口内到达的请求会被归入同一个推理批次，而不考虑其所属场景。这种策略将不同场景的流量合并为更大的批次，从而最大限度地减少额外等待时间。

4 实验

在本节中，我们针对多场景工业推荐系统开展了广泛的离线实验和在线 A/B 测试，以回答以下研究问题：

- **RQ1：** 与现有的召回与排序基线相比，我们的 LazUnirec 表现如何?
- **研究问题2：** LazUnirec 的不同组件对模型整体性能有何贡献?
- **研究问题3：** LazUnirec 中提出的 LoRA 调制注意力和任务自适应稀疏混合专家，是否确实缓解了梯度冲突?
- **研究问题4：** LazUnirec 中提出的不确定性估计，是否确实能在发生梯度冲突时引导模型沿着更可靠的梯度更新方向前进?
- **RQ5：** 多场景训练数据是否确实能提升实验性能?
- **研究问题6：** LazUnirec 的在线效率如何?
- **研究问题7：** LazUnirec 在多场景工业部署环境下的真实世界在线A/B测试中表现如何?

4.1 实验设置

4.1.1 **数据集。** 我们使用来自真实世界电子商务

表1：以 AUC 和 GAUC 衡量的离线排序性能，其中最佳结果以粗体显示，最佳基线结果加下划线。% Improve 表示 LazUnirec 相较于最佳基线的相对提升。

模型	场景1				场景2				场景3			
	CTR		CVR		CTR		CVR		CTR		CVR	
	AUC	GAUC	AUC	GAUC	AUC	GAUC	AUC	GAUC	AUC	GAUC	AUC	GAUC
Transformer	0.7623	0.6678	0.8923	0.6904	0.7642	0.6765	0.8454	0.5750	0.7239	0.6486	0.8506	0.5701
UniPinRec	0.7630	0.6677	0.8929	0.6907	0.7650	0.6777	0.8457	0.5761	0.7245	0.6495	0.8506	0.5689
UniR^2	0.7596	0.6635	0.8900	0.6913	0.7606	0.6717	0.8424	0.5751	0.7198	0.6444	0.8471	0.5691
UniSGR	0.7630	0.6687	0.8936	0.6934	0.7650	0.6775	0.8471	0.5730	0.7249	0.6496	0.8517	0.5707
OneTrans-V2	0.7634	0.6695	0.8935	0.6928	0.7654	0.6781	0.8478	0.5778	0.7257	0.6499	0.8524	0.5705
LazUnirec	**0.7655**	**0.6722**	**0.8966**	**0.6972**	**0.7678**	**0.6828**	**0.8502**	**0.5803**	**0.7274**	**0.6518**	**0.8555**	**0.5733**
%提升 ↑	+0.28%	+0.40%	+0.34%	+0.55%	+0.31%	+0.69%	+0.28%	+0.43%	+0.23%	+0.29%	+0.36%	+0.46%

表2：离线检索性能 ，其中最佳结果以粗体显示，最佳基线结果加下划线 he 标出。 “% Improve”表示 LazUnirec 相较于最佳基线的相对提升。

模型	场景1			场景2			场景3		
	HR@100	HR@500	HR@1000	HR@100	HR@500	HR@1000	HR@100	HR@500	HR@1000
Transformer	0.3275	0.5502	0.6402	0.5508	0.7556	0.8156	0.7954	0.9009	0.9207
UniPinRec	0.3264	0.5503	0.6416	0.5532	0.7583	0.8189	0.7980	0.9034	0.9236
UniR^2	0.3305	0.5553	0.6447	0.5536	0.7596	0.8198	0.7970	0.9047	0.9214
UniSGR	0.3305	0.5508	0.6433	0.5541	0.7567	0.8171	0.7967	0.9059	0.9239
OneTrans-V2	0.3346	0.5544	0.6484	0.5557	0.7591	0.8203	0.7984	0.9050	0.9221
LazUnirec	**0.3620**	**0.5892**	**0.6802**	**0.5906**	**0.7873**	**0.8432**	**0.8181**	**0.9169**	**0.9389**
%提升 ↑	+8.19%	+6.10%	+4.90%	+6.28%	+3.65%	+2.79%	+2.47%	+1.21%	+1.62%

平台的大规模工业数据，涵盖多个无触发（即，场景1）和基于触发器（即，场景2、3）的推荐场景。对于检索和排序，我们在请求级别组织训练样本，其中每个样本包含用户按时间顺序排列的交互序列、当前场景，以及带有观测反馈的候选物品。我们使用连续25天的日志进行训练，并使用后续一天的数据进行测试。

4.1.2 基线方法。 我们将**LazUnirec** 与原始**Trans-former**以及近期具有代表性的检索与排序统一方法进行比较，包括 **UniPinRec** [15], **UniR^2** [30], **UniSGR** [24], 和**OneTrans-V2** [2]。

4.1.3 评估指标。 遵循以往方法 [2, 15, 30],，我们评估 LazUnirec 和基线模型在检索与排序任务上的效果。具体而言，检索任务报告 HR@N（命中率），其中 $N \in \{100,500,1000\}$；排序任务报告 AUC 和 GAUC。

4.1.4 实现细节。 我们使用 PyTorch 实现 LazUnirec，并采用 AdamW [14] 在 NVIDIA H100 GPU 上进行优化。所有基线模型与 LazUnirec 使用相同的骨干网络配置，包括 5 层 Transformer 层、隐藏层大小为 256，以及 4 个注意力头。最大历史长度为 2,048，所有模型均训练一个 epoch。

每个请求最多包含 15 个排序令牌和 $K = 8$ 个检索查询。检索和排序的 LoRA 秩分别设置为 8 和 16。每个任务自适应稀疏混合专家包含 14 个共享专家，以及每个任务 1 个私有专家，并激活任务私有专家和前-k 个共享专家， $k = 2$。每个专家的中间维度为 1,024，混合专家正则化系数设置为 $\lambda_z = 0.001$。

对于不确定性估计，我们每 100 个优化步骤探测一次梯度，并将 EMA 衰减率设为 0.99。可靠性权重 ω_s^τ被裁剪至 $[0.8, 1.2]$。对于知识蒸馏，温度参数 T_{ranking} 和 $T_{\text{retrieval}}$ 分别设为 0.5 和 0.01，其中 λ_{kd} 随训练线性递增，并在第 5,000 个微步时达到 0.05。

4.2 离线性能（RQ1）

在本节中，我们在表 1 和表 2 中将 LazUnirec 的离线性能与基线方法进行比较，并总结出以下观测值。

在三个场景中，LazUnirec 在检索和排序方面始终优于所有基线方法。这归功于 LoRA调制注意力和任务自适应稀疏混合专家，它们在保留任务特定容量的同时，实现了跨任务知识共享。此外，不确定性估计会为历史方向更稳定的梯度分配相对较高的权重，从而在发生梯度冲突时，引导模型朝着更可靠的梯度更新方向前进。另一个观察结果是，UniR^{2}在大多数排序指标上的表现最差，甚至在所有场景中都低于原始 Transformer。原因在于，其停止梯度机制阻止排序梯度更新

表3：场景1上的消融实验。

模型	CTR		CVR		HR		
	AUC	GAUC	AUC	GAUC	@100	@500	@1000
不使用 LoRA	0.7646	0.6709	0.8955	0.6952	0.3532	0.5819	0.6732
不使用 SMoE	0.7634	0.6692	0.8943	0.6923	0.3408	0.5659	0.6531
无 用户嵌入	0.7643	0.6704	0.8951	0.6941	0.3528	0.5755	0.6682
无 知识蒸馏	0.7653	0.6719	0.8969	0.6977	0.3502	0.5781	0.6666
LazUnirec	**0.7655**	**0.6722**	**0.8966**	**0.6972**	**0.3620**	**0.5892**	**0.6802**

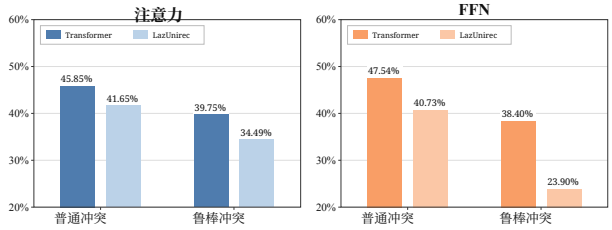


图 4：分别统计原版 Transformer 和 LazUnirec 中注意力及前馈网络组件内梯度冲突所占的比例。

共享基础 Transformer 参数。尽管任务特定组件保留了排序适应性，但这一操作限制了共享参数对排序目标的适应。相比之下，LazUnirec 允许检索和排序共同优化共享参数，同时缓解梯度冲突。此外，场景感知的专家路由和不确定性估计能够适应不同场景，同时受益于共享知识；多场景建模则使 LazUnirec 能够从跨场景交互中学习可迁移的用户偏好。我们将在第 4.6 节进一步分析多场景建模。

4.3 消融实验（RQ2）

为了评估 LazUnirec 中各组件的有效性，我们在场景1上开展了消融实验。具体而言，我们构建了四种变体进行对比分析，具体如下：

• 不使用 LoRA：该变体不采用LoRA调制注意力。
• 不使用 SMoE：该变体不采用任务自适应稀疏混合专家。• 不使用 用户嵌入：该变体移除了不确定性估计。• 不使用 知识蒸馏：在该变体中，我们移除了知识蒸馏。

从表3所示的结果中，我们发现，移除LoRA调制注意力或任务自适应稀疏混合专家中的任一项，都会导致检索和排序性能下降。这些结果证明了在保持任务共享知识的同时保留任务特定容量的有效性。同时，移除不确定性估计也会导致两个任务上的性能下降，表明在发生梯度冲突时，为具有更稳定历史方向的梯度分配相对更高的权重是有效的。此外，我们还观察到，移除知识蒸馏

lation主要降低检索性能。这一观测值表明，知识蒸馏在将知识从排序传输到检索方面起着关键作用。

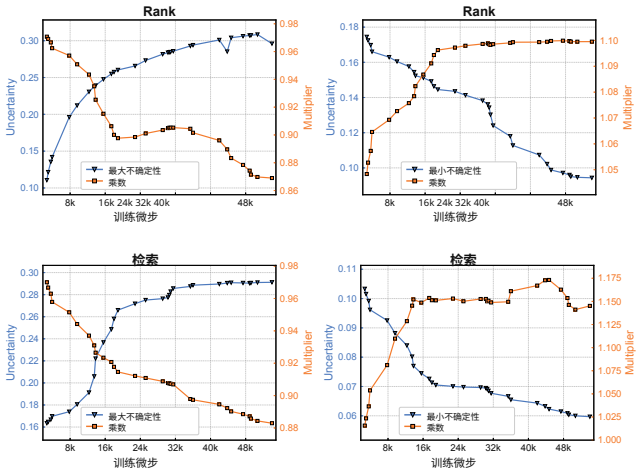


图5：检索和排序过程中，与最大/最小不确定性样本点对应的相对降权/升权乘数。

4.4 梯度冲突分析（RQ3）

为探究LazUnirec是否缓解了梯度冲突，我们在图4中将注意力与前馈网络组件内的普通冲突（即，cos ＜ 0）和鲁棒冲突（即，cos ＜ -0.05）百分比与原始 Transformer进行了比较。与原始 Transformer相比，LazUnirec使注意力内的普通冲突与鲁棒冲突百分比分别相对降低了9.16%和13.23%，并使前馈网络内相应降低了14.32%和37.76%。这些观测值证明了任务共享与任务私有范式的有效性。具体而言，LoRA调制注意力在保留共享注意力参数的同时，提供任务私有适配以适应不同目标。同时，任务自适应稀疏混合专家为检索和排序分配专用专家，并选择性激活共享专家，从而在保持知识共享的同时缓解梯度冲突。鲁棒冲突的尤为显著降低进一步表明，LazUnirec能够在联合优化过程中有效缓解更强的梯度冲突。

4.5 不确定性估计分析（RQ4）

为了研究不确定性估计如何引导梯度更新的方向，我们分别在检索和排序训练过程中随机抽取30个最大不确定性观测值和30个最小不确定性观测值。对于每个观测值，我们记录其不确定性 $\mathcal{E}_s^\tau$及相应的损失加权乘数 ω_s^τ，如图5所示。

从结果中可以观察到，高不确定性观测值获得的乘数较低，从而降低了其对总损失的相对贡献。由于其不确定性通常会在训练过程中增加，相应的乘数往往会降低。与此同时，低不确定性观测值获得的乘数较高，从而增加了其相对贡献。在检索和排序中均一致观察到这一趋势，表明不确定性估计会根据梯度可靠性自适应地调整不同观测值的影响，

表 4：场景1、场景2 和场景 X 上的场景消融研究。

变体	场景1		场景2		场景 X	
	CTR GAUC	HR @100	CTR GAUC	HR @100	CTR GAUC	HR @100
w/ 场景1	0.6647	0.3560	N.A.	N.A.	N.A.	N.A.
w/ 场景2	N.A.	N.A.	0.6777	0.5706	N.A.	N.A.
w/ 场景 X	N.A.	N.A.	N.A.	N.A.	0.6417	0.3086
LazUnirec	**0.6722**	**0.3620**	**0.6828**	**0.5906**	**0.6572**	**0.3686**

表 5：在线A/B测试结果。

场景	IPV	订单	买家	GMV
无触发	+5.25%	+5.75%	+4.57%	+4.01%
基于触发	+4.01%	+4.64%	+3.06%	+3.06%

因为不确定性反映了梯度方向的历史波动，对高不确定性观测值进行降权可以降低不稳定梯度的影响，而对低不确定性观测值进行增权则会更加突出稳定梯度。因此，当发生梯度冲突时，参数更新会更加重视可靠的方向。

4.6 多场景分析（RQ5）

在本节中，我们从十余个场景中选取无触发场景1、基于触发的场景2和无触发长尾场景 X，以评估多场景建模的有效性。对于每个选定的场景，我们仅保留其训练数据，并排除所有其他场景，从而构建一个单场景变体。我们根据点击率、GAUC 和 HR@100 对这些变体与 LazUnirec 进行比较，如表 4 所示。

从结果来看，我们发现，多场景训练提升了三个场景中的检索和排序效果。这表明，来自不同场景的交互提供了互补的监督信号，使 LazUnirec 能够通过共享参数学习可迁移的用户偏好。此外，在长尾场景 X 中，效果提升尤为显著，说明来自其他场景的监督信号可以帮助缓解目标场景中的数据稀疏问题。

4.7 在线效率分析（研究问题6）

为评估 LazUnirec 的在线效率，我们研究了四项部署优化带来的在线每秒查询数提升。具体而言，KV缓存通过在检索和排序之间，以及不同请求之间复用用户序列键和值，使在线每秒查询数提升了 20%。算子融合通过减少内存流量和内核启动开销，使在线每秒查询数提升了 44%。此外，低精度服务通过在 SwiGLU 中选择性地采用 FP8 计算，使在线每秒查询数提升了 5%。跨场景批处理则通过将来自不同场景的请求聚合为更大的批次，进一步提升 GPU 利用率，使在线每秒查询数提升了 59%。

4.8 在线A/B测试（研究问题7）

我们在一个大型工业级电子商务平台上，分别于无触发和基于触发的场景中部署了 LazUnirec。LazUnirec 同时替代了原有的基于 MIMA 的检索 [20] 和基于 LazFormer 的排序 [16]。在线A/B测试于2026年10月启动，持续一周，覆盖 10% 的在线流量。结果如表 5 所示。具体而言，在无触发场景中，LazUnirec 使商品详情页浏览量（即，商品详情页浏览量）相对提升了 5.25%，订单提升了 5.75%，买家提升了 4.57%，商品交易总额提升了 4.01%。在基于触发项的场景中，相应的提升分别为 4.01%、4.64%、3.06% 和 3.06%。在线结果表明，LazUnirec 提升了用户参与度和收入，证明了其在真实场景下的多场景工业推荐中的有效性和商业价值。

5 相关工作

Transformer [25] 已在大语言模型中展现出良好的性能，近期的排序模型 [9, 11, 12, 16, 26, 32, 34] 也研究了**Transformer 在工业推荐中的应用**。具体而言，OneTrans [34]在共享的 Transformer骨干网络中整合了用户序列建模与特征交互。MixFormer [11]实现了稠密特征交互与序列建模的联合扩展，而 HyFormer [12]则迭代地结合了长序列解码与异构特征交互。此外，Laz- Former [16] 研究了可迁移生成式预训练，以缓解稠密参数的负迁移和稀疏参数的过拟合。这些研究表明，通过改进架构和训练策略，Transformer 具备实现可扩展排序的潜力。然而，这些进展主要使排序阶段受益，而基于 Transformer 的排序与检索在架构上的相似性，为将这两个阶段整合到统一模型中提供了基础。在此方向上，LazUnirec 应用 Transformer 在检索和排序之间共享用户序列编码，并进一步解决由二者不同优化目标所引发的梯度冲突问题。

统一检索与排序 因能够减少冗余的用户序列编码并促进跨任务知识共享而受到越来越多的关注 [2, 5, 15, 23, 24, 30] 。UniPin- Rec [15] 通过共享的 Transformer 和任务特定头，统一了输入格式、训练与服务流程。UniR^2 [30] 将用户上下文、语义 ID 轨迹和物品特征组织在单个解码器序列中，引入排序特定LoRA和梯度隔离，以控制优化干扰。UniSGR [24]将多场景预训练与联合生成和多目标排序相结合，利用任务感知标记对齐这两个任务。此外，OneTrans-V2 [2] 将统一范围扩展至检索、预排序和精排序，共享用户序列计算，并通过稀疏混合专家扩展骨干网络。然而，这些方法忽视了跨场景任务特定适配的协调以及可靠的共享参数更新，从而导致梯度冲突问题。LazUnirec允许检索和排序共同优化共享参数，同时在注意力和前馈网络组件中引入任务私有容量，以缓解梯度冲突。此外，该方法根据历史方向稳定性估计梯度可靠性，以指导联合优化，并引入知识蒸馏，实现明确的跨任务知识迁移。

6 结论

本文提出了 LazUnirec，一种面向多场景工业推荐的统一检索与排序方法。我们首先引入LoRA调制注意力和任务自适应稀疏混合专家，通过构建任务共享参数和任务私有参数，缓解注意力与前馈网络组件中的梯度冲突。此外，我们提出不确定性估计，以评估检索和排序的梯度可靠性，从而在发生冲突时引导模型朝更可靠的梯度更新方向优化。此外，知识蒸馏进一步建立了从排序到检索的明确知识传递路径。大量离线实验和在线A/B测试证明了 LazUnirec 的有效性。LazUnirec 已部署于真实的多场景推荐系统中，为平台带来了显著收益。

参考文献

[1] Yimeng Bai、Yang Zhang、Fuli Feng、Jing Lu、Xiaoxue Zang、Chenyi Lei 和 Yang Song。2024。GradCraft：通过整体梯度构造提升多任务推荐。在 第*30*届*ACM SIGKDD*知识发现与数据挖掘会议论文集 中。4774–4783。[2] Hannan Cao、Jun Guo、Haolei Pei、Zhaoqi Zhang、Tianyu Wang、Ziyang Wang、Youchen Sun、Yue Xue、Yucheng Mao、Lintao Yan 等。2026。OneTrans-V2：在工业推荐系统中通过单一Transformer统一检索、预排序和精排序。*arXiv*预印本 *arXiv:2609.28589*（2026）。[3] Minghan Chen、Guikun Chen、Wenguan Wang 和 Yi Yang。2025。Seed-grpo：通过语义熵增强的不确定性感知策略优化。*arXiv*预印本 *arXiv:2505.12346*（2025）。[4] Paul Covington、Jay Adams 和 Emre Sargin。2016。用于YouTube推荐的深度神经网络。在 第*10*届*ACM*推荐系统会议论文集 中。191–198。[5] Sunhao Dai、Jiakai Tang、Jiahua Wu、Kun Wang、Yuxuan Zhu、Bingjun Chen、Bangyang Hong、Yu Zhao、Cong Fu、Kangle Wu 等。2026。Onepiece：将上下文工程与推理引入工业级级联排序系统。在第*32*届*ACM SIGKDD*知识发现与数据挖掘会议论文集第*2*卷 中。7117–7128。[6] Jiaxin Deng、Shiyao Wang、Kuo Cai、Lejian Ren、Qigen Hu、Weifeng Ding、Qiang Luo 和 Guorui Zhou。2025。Onerec：通过生成式推荐器和迭代式偏好对齐统一检索与排序。*arXiv*预印本 *arXiv:2502.18965*（2025）。[7] Peizheng Guo、Jianqi Zhang、Xingyu Zhang、Yun Fan、Jiahuan Zhou、Chang- wen Zheng 和 Wenwen Qiang。2026。GUPO：用于大语言模型后训练的梯度不确定性感知策略优化。*arXiv*预印本 *arXiv:2608.17411*（2026）。[8] Wei Guo、Chang Meng、Enming Yuan、Zhicheng He、Huifeng Guo、Yingxue Zhang、Bo Chen、Yaochen Hu、Ruiming Tang、Xiu Li 等。2023。面向多行为推荐的压缩交互图框架。在 *2023*年 *ACM Web*会议论文集 中。960–970。[9] Ruidong Han、Bin Yin、Shangyu Chen、He Jiang、Fei Jiang、Xiang Li、Chi Ma、Mincong Huang、Xiaoguang Li、Chunzhen Jing 等。2025。Mtgr：美团工业级生成式推荐框架。在 第*34*届*ACM*信息与知识管理国际会议论文集 中。5731–5738。[10] Edward J Hu、Yelong Shen、Phillip Wallis、Zeyuan Allen-Zhu、Yuanzhi Li、Shean Wang、Liang Wang、Weizhu Chen 等。2022。LoRA：大语言模型的低秩适配。*Iclr* 1，2（2022），3。[11] Xu Huang、Hao Zhang、Zhifang Fan、Yunwen Huang、Zhuoxing Wei、Zheng Chai、Jinan Ni、Yuchao Zheng 和 Qiwei Chen。2026。MixFormer：工业推荐系统中稠密与序列的协同扩展。*arXiv*预印本 *arXiv:2602.14110*（2026）。[12] Yunwen Huang、Shiyong Hong、Xijun Xiao、Jinqiu Jin、Xuanyuan Luo、Zhe Wang、Zheng Chai、Shikang Wu、Yuchao Zheng 和 Jingjian Lin。2026。HyFormer：重新审视序列建模与特征交互在点击率预测中的作用。*arXiv*预印本 *arXiv:2601.12681*（2026）。[13] Yuchen Jiang、Jie Zhu、Xintian Han、Hui Lu、Kunmin Bai、Mingyu Yang、Shikang Wu、Ruihao Zhang、Wenlin Zhao、Shipeng Bai 等。2026。Tokenmixer-large：扩展工业推荐系统中的大型排序模型。在 第*32*届*ACM SIGKDD*知识发现与数据挖掘会议论文集第*2*卷 中。7489–7500。[14] Diederik P Kingma 和 Jimmy Ba。2014。Adam：一种随机优化方法。*arXiv*预印本 *arXiv:1412.6980*（2014）。

[15] Hanyu Li、Yi-Ping Hsu、Aditya Mantha、Prabhat Agarwal、Laksh Bhasin、Jialu Wang、Hongtao Lin、Bella Huang、Yaxin Li、Xinyi Li 等。2026。UniPinRec：在 Pinterest 规模下统一生成式检索与排序。*arXiv* 预印本 *arXiv:2606.00422*（2026）。[16] Xiaodong Li、Alin Fan、Mingyang Li、Yan Xiao、Shichao Nie、Junfeng Zhang、Shaochuan Lin、Zhanming Ou、Tao Luo 和 Xiaoyi Zeng。2026。LazFormer：通过可迁移生成式预训练扩展 Transformer，实现工业推荐。*arXiv* 预印本 *arXiv:2609.14978*（2026）。[17] Xiaodong Li、Ruochen Yang、Shuang Wen、Shen Wang、Yueyang Liu、Guoquan Wang、Weisong Hu、Qiang Luo、Jiawei Sheng、Tingwen Liu 等。2026。FARM：面向跨域直播推荐的频率感知模型。载于第*32*届 *ACM SIGKDD*知识发现和数据挖掘会议论文集，第*1*卷。2302–2311。[18] Mingrui Liu、Sixiao Zhang 和 Cheng Long。2025。面向序列推荐的方面感知多头混合专家模型。载于第十八届 *ACM*国际网络搜索与数据挖掘会议论文集。127–135。[19] Fuyu Lv、Taiwei Jin、Changlong Yu、Fei Sun、Quan Lin、Keping Yang 和 Wil- fred Ng。2019。SDM：面向在线大规模推荐系统的序列深度匹配模型。载于第*28*届 *ACM*国际信息与知识管理会议论文集。2635–2643。[20] Xingyuan Mao、Alin Fan、Shichao Nie、Junfeng Zhang、Yan Xiao、Tao Luo 和 Xiaoyi Zeng。2026。MIMA：通过多正样本独占分配实现多兴趣推荐。*arXiv* 预印本 *arXiv:2609.12842*（2026）。[21] Zibin Pan、Zhichao Wang、Chi Li、Kaiyan Zheng、Boqi Wang、Xiaoying Tang 和 Junhua Zhao。2025。结合梯度下降与冲突缓解的联邦遗忘。载于*AAAI*人工智能会议论文集，第39卷。19804–19812。[22] Shashank Rajput、Nikhil Mehta、Anima Singh、Raghunandan Hulikal Keshavan、Trung Vu、Lukasz Heldt、Lichan Hong、Yi Tay、Vinh Tran、Jo nah Samost 等。2023。基于生成式检索的推荐系统。神经信息处理系统进展 36（2023），10299–10315。[23] Dekai Sun、Yiming Liu、Jiafan Zhou、Xun Liu、Chenchen Yu、Yi Li、Jun Zhang、Huan Yu 和 Jie Jiang。2026。OneRanker：工业广告推荐中使用单一模型实现统一生成与排序。*arXiv* 预印本 *arXiv:2603.02999*（2026）。[24] Jiawei Sun、Jun Yang、Ziyue Guo、Dongyue Xu、Jianan Yan、Lifang Deng 和 Xiaoyi Zeng。2026。UniSGR：语义ID生成与排序的统一框架。*arXiv* 预印本 *arXiv:2607.04068*（2026）。[25] Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Łukasz Kaiser 和 Illia Polosukhin。2017。Attention Is All You Need。神经信息处理系统进展 30（2017）。[26] Chunqi Wang、Bingchao Wu、Taotian Pang、Jiahao Wang、Jie Yang、Jia Liu、Hao Zhang、Hai Zhu、Lei Shen、Shizhun Wang 等。2026。SORT：面向工业规模推荐系统的系统化优化排序 Transformer。*arXiv* 预印本 *arXiv:2603.03988*（2026）。[27] Shen Wang、Yusheng Huang、Ruochen Yang、Shuang Wen、Pengbo Xu、Jiangxia Cao、Yueyang Liu、Kuo Cai、Chengcheng Guo、Shiyao Wang 等。2026。Onelive：面向直播推荐的动态统一生成式框架。载于第*32*届 *ACM SIGKDD*知识发现和数据挖掘会议论文集，第*2*卷。8155–8166。[28] Ziqi Wang、Xingzhou Lou、Meiqi Wu、Zhengqi Wen 和 Junge Zhang。2026。面向推理大语言模型的校准感知策略优化。载于第*64*届计算语言学协会年会论文集（第*1*卷：长篇论文）。18375–18390。[29] Zhipeng Wei、Kuo Cai、Junda She、Jie Chen、Minghao Chen、Yang Zeng、Qiang Luo、Wencong Zeng、Ruiming Tang、Kun Gai 等。2026。Oneloc：面向本地生活服务的地理感知生成式推荐系统。载于第十九届 *ACM* 国际网络搜索与数据挖掘会议论文集。735–744。[30] Ruochen Yang、Shuang Wen、Pengbo Xu、Yusheng Huang、Jiangxia Cao、Shuang Yang、Zhaojie Liu、Jiawei Sheng 和 Tingwen Liu。2026。在单一仅解码器序列中统一生成式召回与多目标排序。*arXiv* 预印本 *arXiv:2607.24439*（2026）。[31] Rex Ying、Ruining He、Kaifeng Chen、Pong Eksombatchai、William L Hamilton 和 Jure Leskovec。2018。面向互联网规模推荐系统的图卷积神经网络。载于第*24*届 *ACM SIGKDD*知识发现与数据挖掘国际会议论文集。974–983。[32] Jiaqi Zhai、Lucy Liao、Xing Liu、Yueming Wang、Rui Li、Xuan Cao、Leon Gao、Zhao- jie Gong、Fangda Gu、Michael He 等。2024。行动胜于言语：用于生成式推荐的万亿参数序列转换器。*arXiv* 预印本 *arXiv:2402.17152*（2024）。[33] Kun Zhang、Jingming Zhang、Wei Cheng、Yansong Cheng、Jiaqi Zhang、Hao Lu、Xu Zhang、Haixiang Gan、Jiangxia Cao、Tenglong Wang 等。2026。OneMall：一个模型，更多场景——快手电商端到端生成式推荐器家族。*arXiv* 电子预印本（2026），arXiv–2601。[34] Zhaoqi Zhang、Haolei Pei、Jun Guo、Tianyu Wang、Yufei Feng、Hui Sun、Shaowei Liu 和 Aixin Sun。2026。Onetrans：使用单一 Transformer 统一工业推荐系统中的特征交互与序列建模。载于*2026*年 *ACM Web*会议论文集。8162–8170。

[35] Guorui Zhou、Jiaxin Deng、Jinghao Zhang、Kuo Cai、Lejian Ren、Qiang Luo、Qianqian Wang、Qigen Hu、Rui Huang、Shiyao Wang 等。2025。Onerec 技术报告。*arXiv* 预印本 *arXiv:2506.13695*（2025）。[36] Jie Zhu、Zhifang Fan、Xiaoxie Zhu、Yuchen Jiang、Hangyu Wang、Xintian Han、Haoran Ding、Xinmin Wang、Wenlin Zhao、Zhen Gong 等。2025。Rankmixer：工业推荐系统中的排序模型扩展。在 第*34*届*ACM*信息与知识管理国际会议论文集。6309–6316。