本文为整合个人理解与网络资料写成

V1 的两个缺点

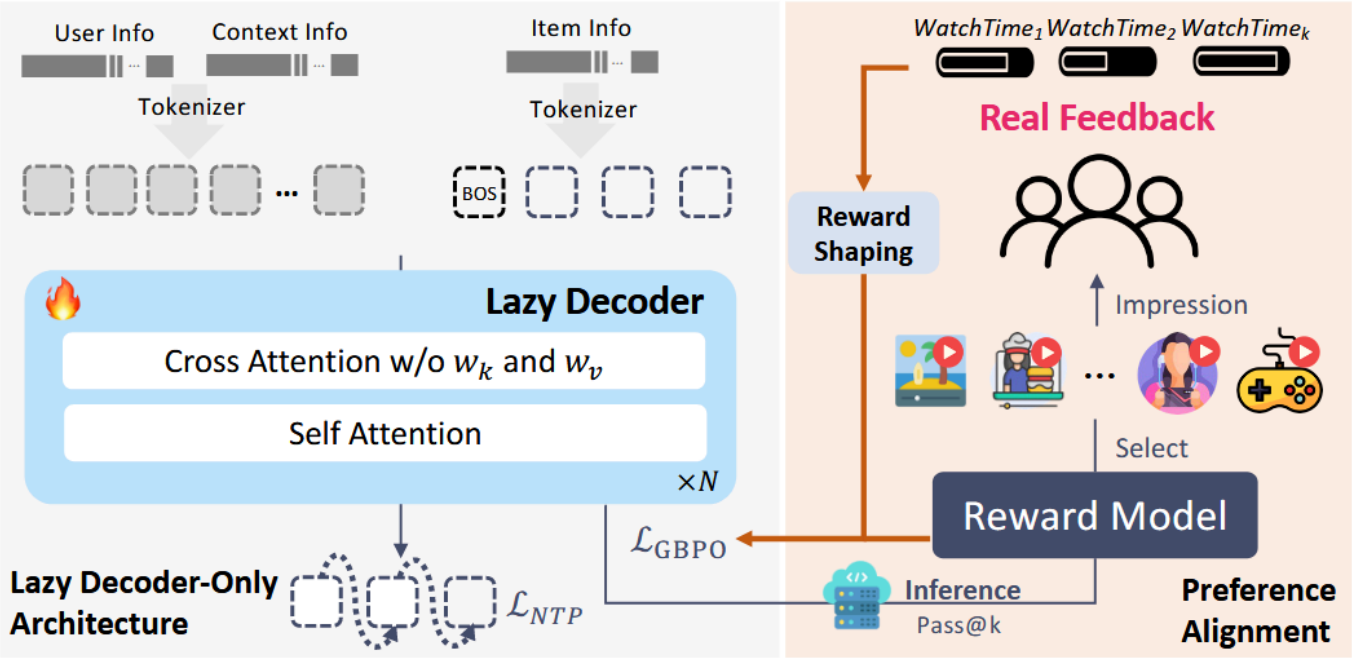
encoder占了资源消耗的97.66%，限制了模型的可扩展性
强化学习仅依赖奖励模型，使得模型容易钻reward的漏洞，模型可能会用不符合设计者预期的方式获取高额奖励，却没真正完成任务的行为（奖励黑客现象）

V2 的改进

Lazy Decoder-only框架，使用decode-only，极大减少了计算量
真实用户交互的偏好对齐：
时长感知的奖励塑造：用于缓解 “视频时长偏差”。例如，若系统仅根据视频播放时长给奖励，可能会倾向于推荐超长视频，而忽略用户对 “时长合适且内容优质” 视频的真实偏好。这一设计让奖励更合理地反映用户对不同时的接受度（比如结合完播率等
自适应比例裁剪：动态调整裁剪比例，限制极端值对策略更新的影响

V2 的整体框架

左侧说明了Lazy Decoder - Only架构
右侧描述了训练后的偏好对齐过程

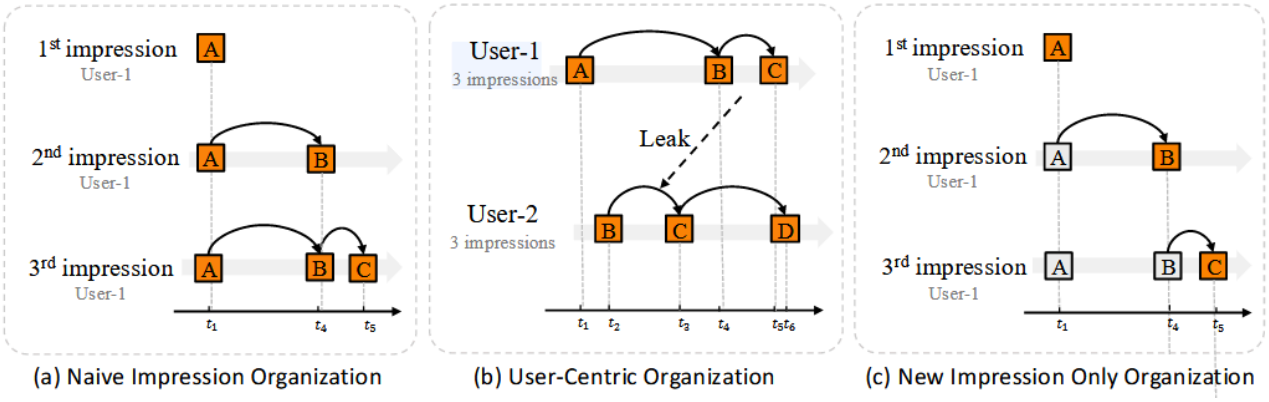


Lazy Decoder-Only 架构

注意 Lazy Decoder-Only 不是 传统的 Decoder-Only，有所改进的就是前面的 “Lazy”

样本组织方式

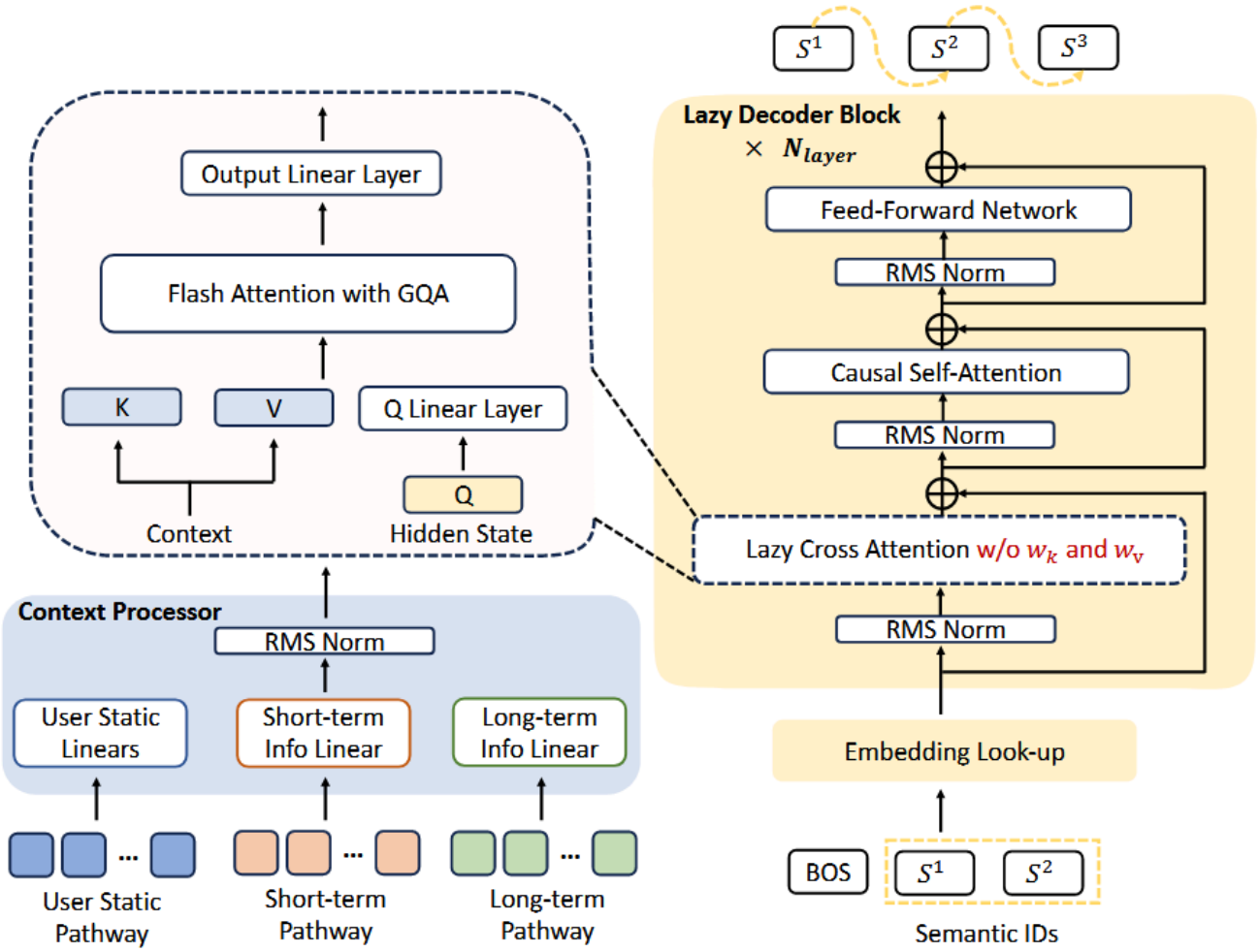
- 传统推荐系统的训练样本是按时间顺序的曝光记录来组织的。然而，当与标准的 “下一个 token 预测” 目标结合时，会产生冗余问题：左图
- 避免这种冗余的一种方法是采用以用户为中心的组织方式，即每个训练样本包含一个用户的完整交互历史：中间图，但存在时间数据泄露风险
- onerec采用：右图，只对最新的item计算损失（灰色的块被排除在下一次的token预测外）



(a) Naive Impression Organization (b) User-Centric Organization (c) New Impression Only Organization

之后论文分析了encoder-decoder和decoder-only在上下文编码和目标解码上的计算量（都很大），为使计算仅集中在目标项目的语义标记上，从而在保持高效的同时支持更大规模模型的扩展， 我们提出了惰性解码器结构(Lazy Decoder-Only)

Lazy Decoder-Only架构图



Context Processor（上下文处理器）

- **输入路径：**
  - User Static Pathway：用户静态画像（年龄、性别、地区等）
  - Short-term Pathway：短期行为（最近点击、观看记录）
  - Long-term Pathway：长期兴趣轨迹
- **处理方式：**
  - 各路径先经过独立的 Linear 层 → 投影到统一维度
  - 拼接成一个统一的 “上下文序列”（Context）
  - 归一化后生成跨层共享的 key-value 对（K, V），供 Decoder 的 Cross-Attention 使用 **（所以每一层都共享同一组K、V，lazy就体现在这里）**

Lazy Decoder（惰性解码器）

输入：

- BOS token（序列起始符）
- 目标物品的语义 token（semantic IDs，例如代表商品的三个语义向量）
  结构：
  由多个 Lazy Decoder Block 堆叠组成，每个 block 包含：

1. **Lazy Cross-Attention（惰性交叉注意力）**
   - 不再进行 key/value 投影（去掉线性层），直接使用 Context Processor 生成的共享 K、V；
   - 支持 Grouped Query Attention (GQA) → 多个 query head 共用一组 K/V；
   - 显著减少显存占用与计算量
2. **Causal Self-Attention**
   - 捕捉语义 token 之间的因果依赖关系；
3. **RMSNorm 用于稳定训练**
4. **Feed-Forward Network**
   - 非线性特征变换；
   - 深层替换为 Mixture-of-Experts (MoE) 提升容量与效率；
     输出：
     最终隐表示通过线性层预测目标物品的语义 token（s1, s2, s3）。
     创新点：

- 不再使用 Encoder
- KV固定，不参与反向传播
- 多层共享同一组KV
- 深层 FFN 替换为稀疏专家路由MoE

与真实世界用户交互的偏好对齐

v2的pre-training预训练和v1相同，但是post-training后训练部分有变化，即本节标题：与真实世界交互的偏好对齐

- v1的后训练是强化学习阶段，完全依赖奖励模型，但奖励模型并不完全等价于真实用户行为，模型的优化倾向是讨好奖励模型而不是真实的用户
- 因此v2使用真实用户的反馈信号直接作为强化学习的奖励， 让模型从真实用户行为中自适应优化，实现「偏好对齐」
  OneRec-V2 的后训练阶段仍分为两步：SFT 、RL，其中SFT阶段也与V1相同，但是RL部分产生了变化。

时长奖励（Duration-Aware Reward Shaping持续时间感知奖励塑造）

用户观看时长（playtime）是最密集的反馈信号；但播放时长（duration）不同的视频，直接用观看时长比较是不公平的（长视频就是时长长），因此：
v2的基于用户反馈奖励是使用基于播放时长的简单但有效的奖励机制

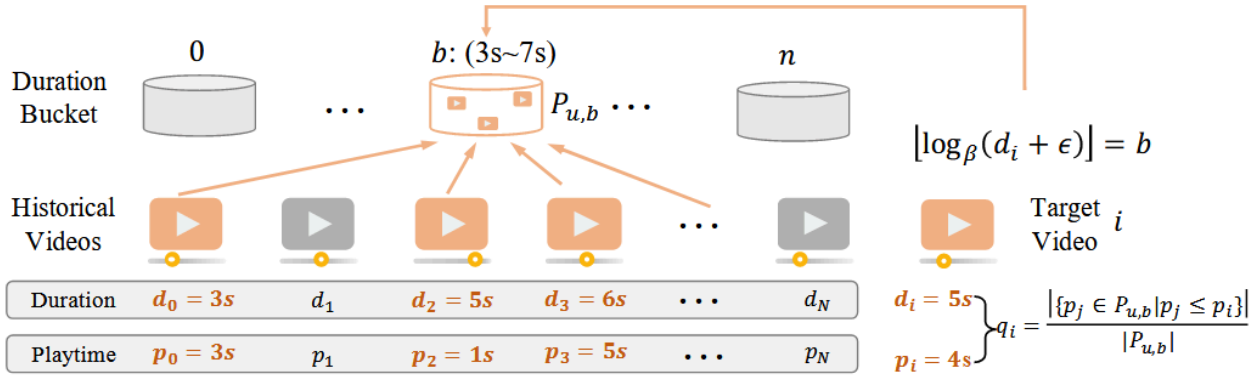


Figure 8 | Illustration of the Duration-Aware Reward Shaping. The videos in a user's watch history are bucketed according to the durations, and for a target video, the quantile of its playing time within the corresponding bucket is computed as the user's preference score.

将用户观看的历史长短视频分桶归一化，对于目标视频，计算在桶内的分位数来得出用户的倾向

RL：GBPO

GBPO (Gradient-Bounded Policy Optimization，梯度有界策略优化)
传统的PPO、GRPO存在的弊端：会丢弃大量的负样本，无法彻底防止梯度爆炸
GBPO使用二元交叉熵（BCE）损失的梯度上界来稳定 RL 的梯度，不会丢弃任何样本，而且会动态限制负样本的梯度大小，使其不会爆炸
如下图，GBPO 在负样本区域施加动态边界，使梯度随概率变化自动调节：

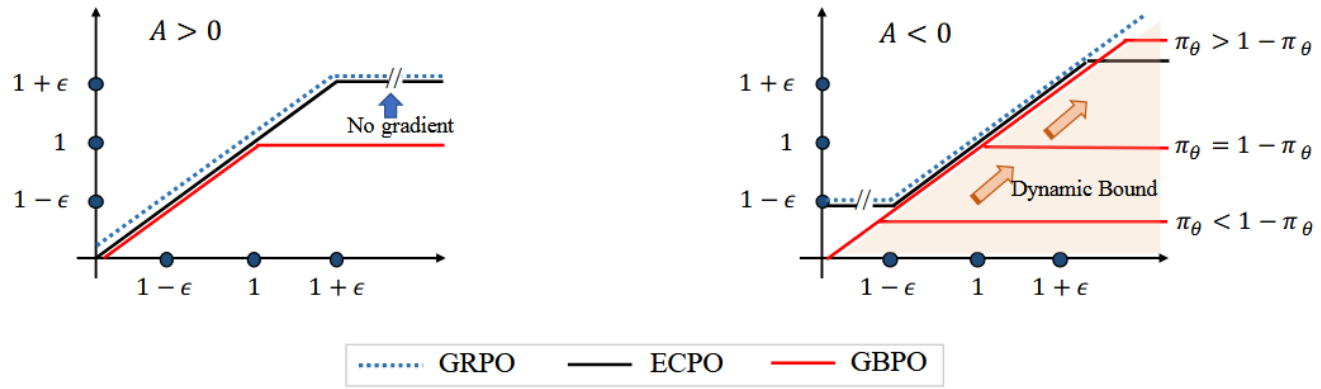


Figure 10 | Illustration of GBPO. The x-axis is $\pi_\theta/\pi_{\theta_{old}}$ and the y-axis is the clipped $\pi_\theta/\pi_{\theta_{old}}$. "//" means "No gradient". Compared with traditional ratio-clipping methods, the main differences of GBPO are: 1. It does not discard the gradients of any samples. 2. For negative samples, the bounding of the ratio is based on a dynamic bound related to π_θ.

创新点：
使用用户真实反馈标签
使用用户时长奖励
使用GBPO

AB测试

测试场景

- 部署平台：快手主站（Kuaishou Main Feed）和快手极速版（Kuaishou Lite Feed）
- 这两个场景是快手的最高流量场景，总共覆盖了 4 亿日活用户（DAU）。
- 实验流量：5% 的线上用户流量被实验
- 实验时长：持续一周（one-week observation period）

模型与推理配置
模型参数量：1B（10亿参数）
上下文长度：3000 tokens
Beam size：512
推理设备：L20 GPU 集群
平均延迟：36 毫秒（ms）
模型算力利用率（MFU）：62%
奖励系统版本：仅使用 用户反馈信号（User Feedback Signals），未使用奖励模型（Reward Model）以降低系统复杂度

目前的局限性

奖励系统（Reward System）

- 当前系统通过人为规则将短期指标（如观看时长）与长期价值挂钩；

- 但模型还**不能直接优化长期回报（long-term value）**；
- 未来目标： 让模型能够自我强化（self-reinforcement），直接学会最大化长期用户满意度。