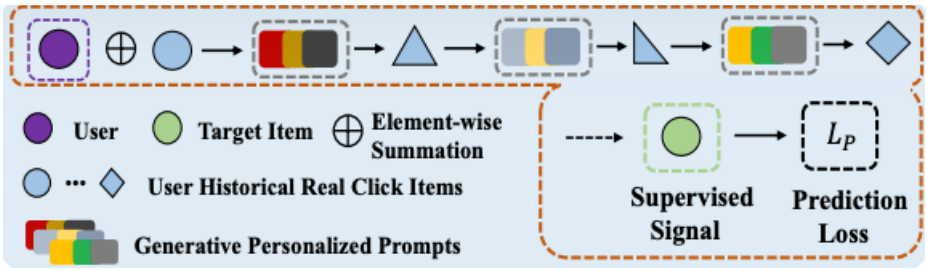


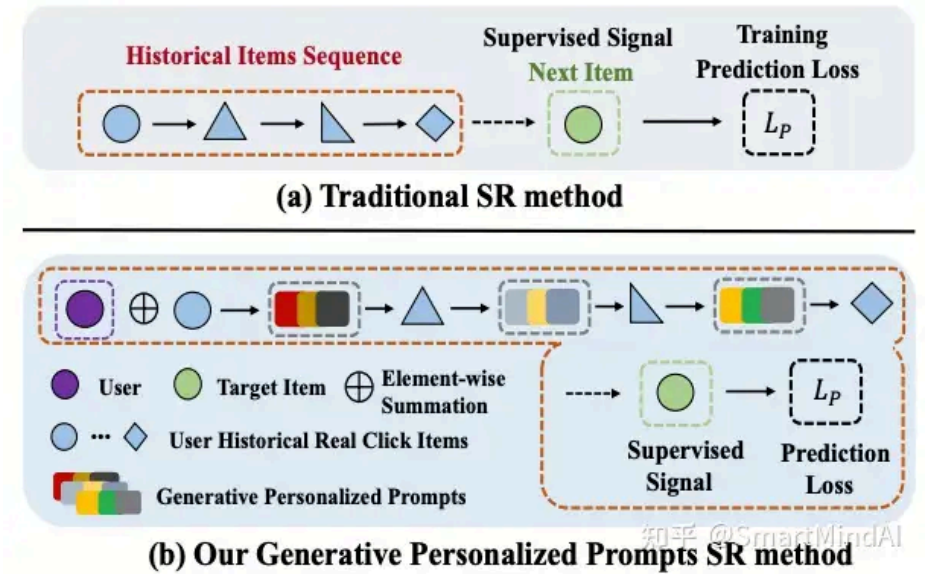
快手 2024：将ChatGPT智能模式融入短视频序列推荐

SmartMindAI
专注搜索、广告、推荐、大模型和人工智能最新技术，欢迎关注我
已关注

Introduction

在本文中，我们不依赖自然语言，而是将ChatGPT的模型结构和训练理念应用于商品序列预测。用户与ChatGPT的互动转化为对商品的提问和反馈，形成序列。在推荐系统+中，用户行为序列生成商品列表，反馈后再生成新的推荐。序列推荐的目标是预测用户未来喜好，这是许多场景的核心。

然而，现有方法主要聚焦于行为序列，忽视了其在全局交互空间中的稀疏性，且过于关注已点击商品，对未点击的潜在兴趣估计不足。为解决这一问题，我们在模型训练中引入未点击的个性化提示，模拟用户兴趣随时间的动态变化，以更全面地理解和预测用户偏好。



我们提出一种名为RecGPT的新方法，它结合ChatGPT的训练策略，通过生成型预训练模型改进了序列推荐。RecGPT通过在用户行为序列中生成个性化提示，提升推荐的个性化水平。

Preliminary

在这个假设下，我们面对用户集合$\mathcal{U}$和商品集合$\mathcal{V}$，每个用户s_u由其行为序列

$$s_u = \{v_{u,1}, v_{u,2}, \ldots, v_{u,j}, \ldots, v_{u,|s_u|}\}$$

其中$v_{u,j}$代表用户第j次的交互商品。

目标是通过学习用户历史，预测其未来最可能的购物行为，即第$s_u| + 1$个商品，用概率模型+表示为：

$$v_{u,|s_u|+1} = \max_{v \in \mathcal{V}} P(v|s_u)$$

$$\arg\max_{v_i \in \mathcal{V}} P(v_{|s_u|+1} = v_i | s_u),$$

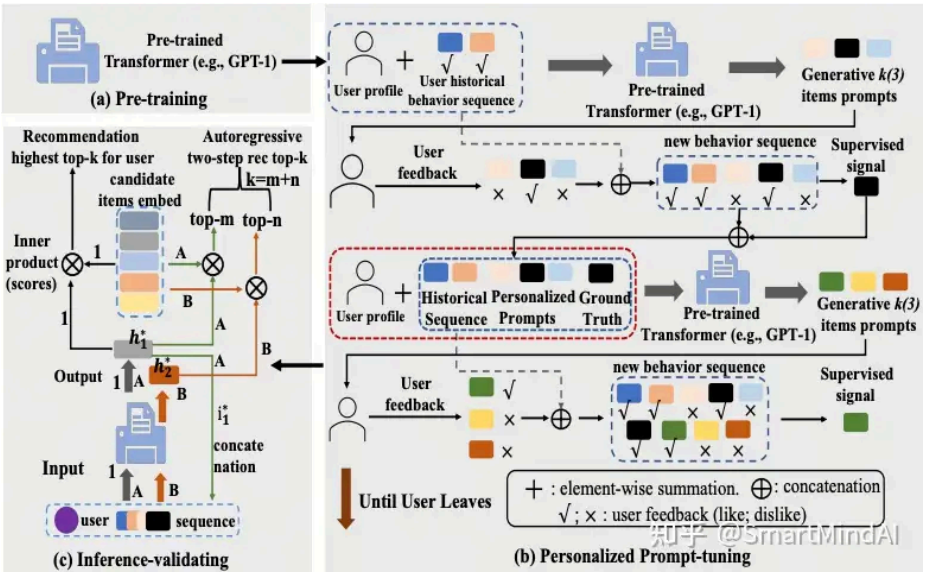
在用户集合$\mathcal{U}$和商品集合$\mathcal{V}$中，我们处理每个用户u的行为序列s_u，目标是预测用户下一次可能的购物行为，即s_u后的商品。通过计算商品v在s_u背景下的概率分布，找到概率最高的商品：

$$v_{u,|s_u|+1} = \max_{v \in \mathcal{V}} \frac{P(v|s_u)}{\sum_{v' \in \mathcal{V}} P(v'|s_u)}$$

这里$P(v|s_u)$通过分析用户行为模式、可能的预测得分（如协同过滤+或深度学习），估计用户对商品v的喜好。该过程填补了序列推荐领域处理多反馈序列的空白。

Methodology

Overall Framework



RecGPT框架的构建，包含三个关键步骤，如图所示。首先，进行个性化自回归生成模型+Transformer的预训练。然后，进行提示定制，通过与用户的交互生成特定的个性化提示。最后，通过推理阶段检验自回归回忆功能的有效性。

Pre-training auto-regressive generative model

首先，我们建立基础模型------Transformer解码器，用以学习序列数据的表示。Transformer因其在序列推荐中的强大表现，成为我们的首选。随后，我们利用经典的Transformer结构+进行预训练，如图所示

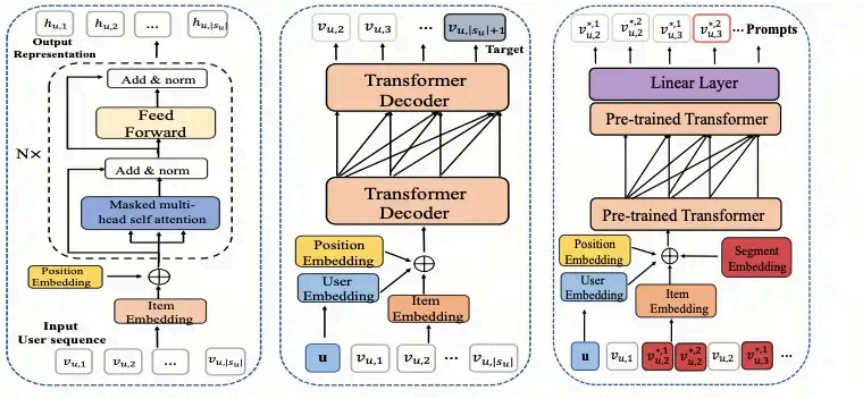


Fig. 3: (a) Model: Generative Pre-training Transformer (GPT). (b) Pre-training framework based on GPT. (c) Fine-tuning framework with personalized prompts.

其中l从1到n $|s_u|$是序列长度d是商品特征维度。每个层次应用一个名为'Masked Multi-head Self-Attention'（MMS）的模块，以及一个位置-wise的'Feed-Forward Network'（FFN）。

MMS通过m个注意力头对输入的

$$S_l \in \mathbb{R}^{|s_u| \times d}$$

进行处理，每个头在 $\mathbb{R}^{|s_u| \times \frac{d}{m}}$ 上通过点积计算注意力得分。具体步骤如下：

$$A_{l,h} = \text{softmax}\left(\frac{Q_{l,h} \cdot K_{l,h}^T}{\sqrt{d/m}}\right) \cdot V_{l,h}$$

其中$Q_{l,h}$、$K_{l,h}^T$和$V_{l,h}$分别是查询、键和值，分别维度翻转后与m维的值相乘，然后对结果取softmax并归一化，以生成注意力权重。接着，MMS的输出经过线性变换（$W_{l,1}$和$b_{l,1}$ $W_{l,2}$和$b_{l,2}$）得到最终的h_l：

$$h_l = \text{ReLU}(S_l W_{l,1} + b_{l,1}) W_{l,2} + b_{l,2}$$

这样，每一层的h_l都包含了对输入序列更深层次的理解，最终用于预测用户下一次可能的交互。

$$\begin{aligned}
&S_l = [A_{l,1}, \cdots, A_{l,m}] W_l^S, \\
&A_{l,h} = \text{softmax}(\frac{Q_{l,h} K_{l,h}^T}{\sqrt{d}} + M) V_{l,h}, \\
&S_l = h_{l-1} W_{l,h}^Q,\ h_{l-1} W_{l,h}^K,\ h_{l-1} W_{l,h}^V, \\
&M = \begin{cases} 0 & \text{Allow to attend} \\ -\infty & \text{Forbidden to attend} \end{cases}
\end{aligned}$$

在这个结构中

$$W_l^S, W_{l,h}^Q, W_{l,h}^K, W_{l,h}^V \in \mathbb{R}^{d \times \frac{d}{m}}$$

是各注意力头的投影矩阵，而 $M \in \mathbb{R}^{|s_u| \times |s_u|}$ 是个注意力掩蔽矩阵，控制着每个位置对其他位置的关注程度。MMS模型用于捕获序列间的联系，位置 wise FFN则生成预测序列的嵌入。

预训练时，计算预训练行为矩阵h_u^l的过程如下：

$$h_u^l = \text{ReLU}\left(W_l^S S_l + \sum_{h=1}^m (W_{l,h}^Q A_{l,h} (W_{l,h}^K)^T) + b^l\right)$$

其中，首先对MMS的输出S_l进行线性变换，接着对注意力得分$A_{l,h}$与键矩阵$K_{l,h}$进行点积加权求和，经过ReLU激活，最后加上偏置b^l。这一过程模拟了序列中的当前状态如何通过历史信息推断下一个项目。

$$\begin{aligned}
&h_u^0 = u W_u + s_u W_e + W_p, \\
&h_u^l = \text{Transformer}(h_u^{l-1}), \\
&h_{u,|s_u|} = f_{seq}(s_u|\theta) = h_{u,|s_u|}^N,
\end{aligned}$$

该公式表示损失函数L，用于评估模型预测下一个点击商品的准确性。具体地 $p(v_{u,|s_u|+1}|\theta; s_u)$ 是通过模型f_{seq}，利用参数θ根据用户u的历史行为序列s_u计算出的预测概率。损失函数计算的是实际下一个点击商品$v_{u,|s_u|+1}$与模型预测的负对数似然，通过$-\log$取对数来量化差距，目的是最小化误差。

$$\begin{aligned}
L_{SR} = -\sum_{u \in \mathcal{U}} \sum_{t \in [2, |s_u|]} \{&\log(\sigma([h_{u,t}]^T e_{u,t+1})) - \\
&\sum_{s_{u,neg} \in O_{u,t}^-} \log(1 - \sigma([h_{u,t}]^T e_{u,neg}))\},
\end{aligned}$$

计算得出，通过非线性激活函数δ处理。负样本集合$O_{u,t}^{-}$包含与$v_{u,t+1}$相对应的$s_{u,1}, s_{u,2}, \ldots, s_{u,neg}$

其中每个$s_{u,neg}$的嵌入向量记作$e_{u,neg}$。 预训练损失函数L通过对比学习构建，考虑了两个部分：实际下一个点击商品$v_{u,t+1}$的预测概率和对应的负样本$s_{u,neg}$不被选择的概率。具体表达式为：

$$L = -\sum_u \sum_{t=1}^{|s_u|} (\delta(\log(p(v_{u,t+1}|\theta; s_u)) + \log(1 - p(s_{u,neg}|\theta; s_u)))$$

其中$p(v_{u,t+1}|\theta; s_u)$是模型对$v_{u,t+1}$的预测 $1 - p(s_{u,neg}|\theta; s_u)$

是$s_{u,neg}$不被选的概率，两者之和的交叉熵用于调整模型参数θ，以最大化预测真实结果的正确性，同时避免错误预测负样本。

Personalized Prompt-tuning

在RecGPT模型经过预训练后，它采用基于用户交互的个性化提示进行后续推荐任务的微调。具体操作如图所示。生成个性化提示是个难题，因为推荐系统的提示词并非自然语言中的常规词汇，且需要针对每个用户定制。

为解决这一问题，RecGPT利用用户ID和行为序列的唯一项目ID，自动化地为所有用户生成个性化的提示。这个过程类似于单向语言模型的自回归生成，但目标是在用户行为序列$[v_{u,2}, v_{u,3}, \ldots]$的基础上，生成用户独有的个性化提示集$v_{u,j}^{i,*}$。 通过算法1，新生成的个性化提示项$id$通过设计的段嵌入矩阵$W_s$与原始行为序列项$id$区分开。然后，通过argmax选择所有可能项$\mathcal{V}$中概率最高的项作为最终提示，确保提示能有效地反映用户需求并促进个性化推荐。

Algorithm 1 Pseudocode to generate prompts sequence s_u^*

Input: User id u; Pre-trained "Transformer_block"; User behavior sequence $s_u = \{v_{u,1}, \cdots, v_{u,|s_u|}\}$; User embedding matrix : W_u; Item embedding matrix : W_e; Position embedding matrix : W_p; Segment embedding matrix : W_s; Linear output layer W_l; Whole items set $\mathcal{V}$; The number of personalized prompts generated: K.
Output: New sequence s_u^*.

Let $t = 1$; $s_u^* = \{v_{u,1}\}$
while $t < |s_u|$ **do**
  $k = 1$
  if $k < K$ **then**
    $h_u^0 = uW_u + s_u^* W_e + W_p + W_s$;
    $h_u^l = \text{Transformer\_block}(h_u^{l-1})$
    $P_{u,t+1} = \text{softmax}(h_{u,|s_u|}^N W_l^T)$
    $v_{u,t+2}^{*,k} = \text{argmax}\mathcal{V}(P_{u,t+1})$
    $s_u^* \leftarrow \text{Concatenation}(s_u^*, v_{u,t+2}^{*,k})$
    $k = k + 1$
  end if
  $s_u^* = [s_u^*, v_{u,t+2}]$
end while
return s_u^*

生成个性化提示后，RecGPT获得了增强的个性化序列s_u^*。这个步骤是通过对每个用户的行为序列应用算法1，结合用户ID和行为项的段嵌入，生成独特且针对用户需求的提示，以提升推荐的个性化程度。通过argmax选择概率最高的项，确保了微调后的提示能够更准确地指导模型进行下一步的推荐。

$$s_u^* = \{v_{u,1}, \underline{v_{u,2}^{*,1}, \cdots, v_{u,2}^{*,K}}, v_{u,2}, \underline{v_{u,3}^{*,1}, \cdots, v_{u,3}^{*,K}}, v_{u,3}, \cdots, \underline{v_{u,|s_u|-1}^{*,1}, \cdots, v_{u,|s_u|-1}^{*,K}}, v_{u,|s_u|}\}$$

在生成个性化的提示后，我们通过算法1对用户行为序列s_u进行了增强，得到增强序列s_u^*。这个过程利用用户ID和行为项的段嵌入，生成定制的提示，以提升推荐的个性化。通过选取概率最高的项（argmax），确保微调后的提示能精确引导模型预测下一个项目。此时，损失函数定义如下：

$$L = -\sum_u \sum_{t=1}^{|s_u^*|} \log(p(v_{u,t+1}|s_u^*; \theta) + \log(1 - p(s_{u,neg}|s_u^*; \theta))$$

$$L = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} -\log p(v_{u,|s_u|+1} | s^*_{u,1}, s^*_{u,2+K}, s^*_{u,3+2K}, \cdots, s^*_{u,|s_u|+(|s_u|-1)K}; \theta_{LP})$$

在这个场景中$v_{u,|s^*_u|+1}$代表用户u的实际下一个要预测的商品。K定义了生成个性化提示时考虑的最近行为项数量。参数集合$^+\theta_{LP}$包含了段嵌入矩阵W_s和附加的线性层参数W_l，它们共同参与训练，优化的是负对数似然损失函数L。 损失函数表达式更新为：

$$L = -\sum_{u=1}^{U} \sum_{t=1}^{K} [\log(p(v_{u,t+1} | s^*_u; \theta_{LP})) + \log(1 - p(s_{u,neg} | s^*_u; \theta_{LP}))]$$

这里，通过结合预训练的RecGPT模型和生成的个性化提示s^*_u，对用户行为序列进行预测，实现了个性化推荐，其中U表示所有用户。通过调整θ_{LP}，模型能够更准确地预测用户下一个可能的选择，同时避免非目标商品的误判。

Inference-validating auto-regressive recall

在推断阶段（如图2(c)中的路径1），现有方法主要通过用户历史轨迹来预测用户对商品i在时间步$(t+1)$的偏好，公式如下：

$$\text{Score}_{u,i}(t+1) = f_{rec}(h_{u,t}, v_i; \theta)$$

其中$h_{u,t}$是用户过去行为的表示，由序列模型f_{seq}以参数θ计算得出。f_{rec}是个用于预测的函数，它结合用户历史信息和物品特性(v_i)，依据θ来估计用户对该商品的兴趣程度。这个得分是预测用户在下一次行为中对物品i潜在兴趣的指标。

$$P(i_{t+1} = i | i_{1:t}) = e_i^{\mathrm{T}} \cdot F_t^L,$$

Experimental Setup

实验数据集选择包含三个来自亚马逊评论的子集（体育与户外、美容和玩具游戏），以及一个Yelp数据集。所有数据经过标准化处理，将评分视为正样本，未评分或无评论的视为负样本。特别指出，我们仅选取了在2019年1月1日之后的大宗交易记录，并且保留了满足'5核心'条件的用户数据，即每个用户至少拥有5件商品且每件商品至少有5次购买。这些数据来源确保了实验结果的现实性和有效性。

Table 1: Statistics of the datasets after preprocessing.

Dataset	#Users	#Items	#AveLen	Actions	Sparsity
Sports	35,598	18,357	8.3	296,337	99.95%
Beauty	22,363	12,101	8.9	198,502	99.73%
Toys	19,412	11,924	8.6	167,597	99.93%
Yelp	30,431	20,033	10.3	316,354	99.95%

在实现上，我们采用PyTorch在RecBole框架中对所有基线模型$^+$进行Python实现，特别注意保持公正性，所有模型在相同的数据预处理$^+$条件下运行，不使用RecBole默认的增强功能。对于预训练模型，仅使用AB→C的样本来训练，其他阶段采用留一法评估，确保一致性。

在预训练阶段，我们选择Transformer编码器配置，包括1个自注意力块、2个注意力头和嵌入维度64，最大序列长度为50。使用Adam优化器，学习率为0.001，批量大小256。对于RecGPT，我们调整生成提示窗口大小K的范围为0到6，两步自回归回忆的数量m和n从1到20。

实验中，我们以k=5和k=10的Top-k精确率（HR\@k）和Top-k归一化折现累积增益（NDCG\@k）作为主要指标，对整个项目集合进行排名，以确保公正的比较。

Performance Comparison (RQ1)

PopRec	0.0052	0.0028	0.0081	0.0037	0.0064	0.0030	0.0098	0.0040
BPR	0.0030	0.0019	0.0055	0.0027	0.0038	0.0026	0.0060	0.0033
STAMP	0.0079	0.0051	0.0119	0.0064	0.0077	0.0045	0.0114	0.0057
GRU4Rec	0.0113	0.0072	0.019	0.0097	0.0162	0.0097	0.0300	0.0141
NARM	0.0132	0.0085	0.0234	0.0118	0.0230	0.0142	0.0401	0.0197
S^3-Rec$_{IPS}$	0.0124	0.0086	0.0205	0.0111	0.0202	0.0119	0.0336	0.0163
BERT4Rec	0.0200	0.0130	0.0313	0.0166	0.0382	0.0210	0.0592	0.0319
SASRec	0.0203	0.0135	0.0324	0.0174	0.0398	0.0261	0.0614	0.0331
Pre-train	0.0203	0.0132	0.0324	0.0171	0.0418	0.0272	0.0610	0.0334
Fine-tuning	0.0212	0.0141	0.0328	0.0178	0.0421	0.0275	0.0618	0.0338
RecGPT$_1$	0.0213	0.0141	0.0333	0.0180	0.0426	0.0283	0.0651	0.0356
RecGPT	**0.0219**	**0.0143**	**0.0339**	**0.0181**	**0.0440**	**0.0289**	**0.0654**	**0.0357**
Improved	3.302%	1.418%	3.354%	1.685%	4.513%	5.091%	5.825%	5.621%
	Toys and Games				**Yelp**			
PopRec	0.0039	0.0021	0.0070	0.0031	0.0045	0.0023	0.0091	0.0038
BPR	0.0030	0.0019	0.0047	0.0024	0.0023	0.0015	0.0039	0.0021
STAMP	0.0033	0.0019	0.0061	0.0028	0.0041	0.0025	0.0071	0.0035
GRU4Rec	0.0157	0.0097	0.0274	0.0134	0.0122	0.0076	0.0219	0.0107
NARM	0.0257	0.0174	0.0405	0.0221	0.0152	0.0095	0.0256	0.0129
S^3-Rec$_{IPS}$	0.0227	0.0146	0.0397	0.0200	0.0159	0.0094	0.0269	0.0129
BERT4Rec	0.0455	0.0307	0.0683	0.0380	0.0156	0.0096	0.0262	0.0130
SASRec	0.0479	0.0334	0.0698	0.0405	0.0161	0.0100	0.0290	0.0142
Pre-train	0.0487	0.0342	0.0630	0.0388	0.0166	0.0102	0.0278	0.0138
Fine-tuning	0.0492	0.0335	0.0701	0.0402	0.0168	0.0103	0.0287	0.0141
RecGPT$_1$	0.0515	0.0355	0.0711	0.0418	0.0172	0.0105	0.0293	0.0144
RecGPT	**0.0529**	**0.0359**	**0.0721**	**0.0419**	**0.0177**	**0.0107**	**0.0297**	**0.0146**
Improved	7.520%	4.971%	2.853%	3.457%	5.357%	3.883%	3.484%	3.546%

1. PopRec和BPR非序列模型在推荐性能上落后于序列模型，证实了利用用户行为序列的有效性。
2. SASRec，一个基于Transformer的模型，通过多头自注意力显著超越了非Transformer的GRU4Rec、Caser、NARM和STAMP，表明其在序列推荐中的优越性。
3. 我们的方法（RecGPT和RecGPT$_1$）在大约88%和96%的情况下超越最佳基线，表明个性化提示能更有效地利用预训练模型*的上下文信息，证实了生成式方法在序列推荐中的优势。RecGPT在所有情况下都优于RecGPT$_1$，显示自回归回忆在捕捉未来偏好上的有效性。综上，我们提出的新训练范式ChatGPT在序列推荐任务中效果显著。

Parameter Analysis (RQ2)

我们研究了参数m_n，它控制生成个性化提示的数量。实验中，我们在$[0, 6]$这个区间内变化K，观察模型性能的变化。如图表明，随着k增大，模型性能提高，这归功于生成提示的个性化匹配能力。然而，最优表现出现在$[1, 3]$这个窗口，可能是因为过大的K导致生成过多不相关提示，减弱了原始序列的关联性。

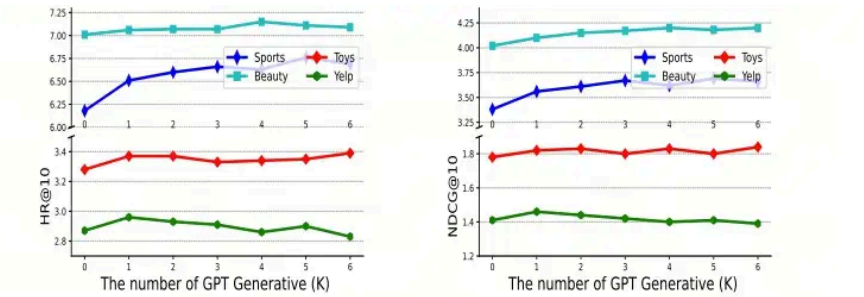


Fig. 4: Sensitive of the size of window K in term of HR@10 and NDCG@10.

Online A/B Testing (RQ1)

我们还在KuaiShou视频应用的推荐系统中进行了实际的A/B测试，以进一步验证我们的方法。这项实证研究通过对比不同版本（如Variant_1, Variant_2等）的应用效果，实际展示了RecGPT和两步自回归回忆组件在实际场景中的效能，强化了之前理论分析的结论。

在KuaiShou视频应用的推荐系统中，我们对RecGPT和两步自回归回忆策略进行了A/B测试替换现有基线（ComiRec）。实验中，当用户登录时，我们首先通过自回归*生成用户嵌入，然后以ANN检索作为基准。作为对照，我们采用了基于用户多兴趣学习的ComiRec，这是当前应用的主要召回方法。评估指标包括用户评论、分享、播放次数、关注度和观看时长等行为数据。

进验证了我们在RecGPT和个性化提示策略方面的有效性。

原文《RecGPT: Generative Personalized Prompts for Sequential Recommendation via ChatGPT Training Paradigm》

编辑于 2024-05-31 07:20 · IP 属地北京

快手 序列推荐 ChatGPT

理性发言，友善互动

2 条评论 默认 最新

zscs
推全了么？基线是什么？
05-11 · 北京 回复 喜欢

Neicul
这个指标收益看起来很小呀
05-09 · 美国 回复 喜欢

文章被以下专栏收录

电商序列模型
挖掘模型的真谛

推荐阅读

工作室如何挂快手
很简单，只需要准备一台电脑装上雷电模拟器，然后根据自己电脑配置选择开数，高配电脑可以十开甚至更多相当于是你有了10部手机，将下载好的快手直接安装进模拟器，打开快手就可以进行正常的...
770785839

auto js 抖音（快手）极速版自动刷视频脚本
上班快乐摸鱼，嘻 机器型号：Lenovo k5 note； 系统：Android 7 软件：autojs pro ；抖音（快手）极速版；可以在手机上coding，也可连接pc，看个人喜好能够实现自动刷新视频、点赞、...
鱼不肉

零基础玩抖音必须下载的5个手机软件
热心网民小柚子

抖音、快手账号播放上不需要一点运营技巧
杨怼怼 发表于