Sim2Rec：一种基于模拟器的决策方法，用于优化真实世界序列推荐系统中的长期用户参与度

Xiong-Hui Chen[1,3], Bowei He[5], Yang Yu[1, 3, 4, *], Qingyang Li[2],Zhiwei Qin[6, +], Wenjie Shang[2],Jieping Ye[2],Chen Ma[51] 新软件技术国家重点实验室、南京大学，中国南京[2] 滴滴人工智能实验室，[3] *Polixir.ai*，[4] 鹏城实验室，中国深圳，[5] 香港城市大学，[6] *Lyft*chenxh@lamda.nju.edu.cn, boweihe2-c@my.cityu.edu.hk, yuy@nju.edu.cn, qingyangli@didiglobal.com, zq2107@caa.columbia.edu, shangwenjie@didiglobal.com, jieping@gmail.com, chenma@cityu.edu.hk

arXiv:2305.04832v1 [cs.IR] 3 May 2023

摘要——**研究表明，强化学习（RL）适用于序列推荐系统（SRS）中的长期用户参与度（LTE）优化，其通过寻找能够最大化长期奖励的策略来实现这一目标。然而，强化学习也存在不足，尤其是在探索过程中需要大量在线样本，这在现实世界应用中具有较高风险。规避这一风险的一种有效方法是构建模拟器，并在模拟器中学习最优推荐策略。在长期用户参与度优化中，模拟器需要模拟多个用户针对给定推荐结果所产生的日常反馈。然而，构建一个不存在现实差距的用户模拟器，即能够准确预测用户反馈的模拟器，是不切实际的，因为用户的反应模式十分复杂，而且每个用户的历史日志都十分有限，这可能会误导基于模拟器的推荐策略。本文提出了一种实用的基于模拟器的推荐策略训练方法——从模拟到推荐（Sim2Rec），以解决长期用户参与度优化中的现实差距问题。具体而言，Sim2Rec 引入了一个模拟器集合，用于生成用户行为模式的各种可能性，随后训练一个环境参数提取器，以识别模拟器中的用户行为模式。最后，基于推断出的环境参数，训练一种上下文感知策略，以针对用户的所有变体做出最优决策。由于该策略已经学会识别各种不同的用户行为模式，并根据推断出的环境参数做出正确决策，因此能够直接迁移到未见环境（例如现实世界）中。实验在合成环境以及现实世界的大规模网约车平台滴滴出行上进行。实验结果表明，Sim2Rec 实现了显著的性能提升，并在未见环境中生成了稳健的推荐结果。**

索引词—**强化学习，现实差距，推荐系统**

I. 引言

旨在为用户推荐潜在相关物品序列的序列推荐系统（SRS）已在各种互联网平台中发挥重要作用，例如网约车应用 [1, 2], 电子商务网站 [3, 4, 5, 6], 和视频网站 [7, 8]。提升长期参与度（LTE）通常代表用户留在平台上并保持活跃的意愿，是 SRS 中的一项关键目标 [7, 9]。近期研究表明，强化学习（RL）是一种优化 LTE 的有前景方法。这些研究将推荐过程视为用户与推荐智能体之间的序列交互 [10], ，然后使用强化学习寻找能够最大化用户从交互中获得的累积奖励的最优策略。

然而，强化学习方法依赖于在现实世界中进行大量试错采样，这阻碍了强化学习在这些风险敏感型平台中的进一步应用 [11, 12]。在模拟器中训练强化学习策略是避免试错成本的理想方式。在序列推荐系统场景中，模拟器用于模拟用户对给定推荐结果的响应。然而，构建一个准确的模拟器并不现实，因为用户行为通常十分复杂，而且历史日志有限 [9]。仿真与现实之间的差异，即所谓的现实差距，会导致基于标准强化学习范式学习到的策略在现实世界中的性能出现非预期下降 [13]。然而，在序列推荐系统中，基于存在现实差距的模拟器所构建的标准强化学习范式存在的问题却鲜少得到明确讨论。

本文聚焦于处理 LTE 优化中基于模拟器的强化学习所面临的现实差距问题。我们提出了基于 SRS 环境参数提取器的零样本策略迁移技术来解决这一问题。零样本策略迁移技术已广泛应用于应对具有挑战性的任务中物理模拟器的现实差距[14, 15, 16]。这些技术假设现实差距源于环境参数的差异（例如，机器人控制中不同的摩擦系数）。它们首先从环境参数空间中选取大量不同的环境参数，构建一个模拟器集合。基于该模拟器集合，它们学习一个提取器，以从交互轨迹中推断环境参数；同时学习一个上下文感知策略，以控制智能体根据推断出的参数执行适应性动作，从而实现最佳性能[14, 15]。部署后，提取器根据真实交互轨迹信息调整推断出的环境参数，从而自动使策略适应合适的动作。待策略收集到足够的样本且提取器确定环境参数后，策略迁移便告完成。如果环境参数空间

* 通讯作者+ 作者在滴滴人工智能实验室工作期间完成的研究

涵盖现实世界的环境参数，且模拟器集合已遍历该空间，我们便可以断定提取器能够推断出正确的参数，策略也将随之调整，以做出正确的决策。

然而，SRS 场景与零样本策略迁移的现有应用在以下方面有所不同：首先，在 SRS 场景中，一个策略需要同时服务多个地区的多位用户。环境参数提取器应识别每位用户的行为模式。此外，每个地区都有其自身的背景，因此不同地区的用户行为并不一致。例如，在网约车平台中，不同城市的司机可能具有不同的参与度（例如，在线时长），这与他们的用户画像无关，因为这些城市的乘客基数可能相差悬殊。本研究将不同地区间的行为差异称为组行为差异，这在现实世界中十分常见 [17]。在这种场景下，如果仅考虑单个用户的交互轨迹，就很难识别环境参数的表征。其次，在 SRS 场景中，用户模拟器很难通过“物理规则”建模，因此目前的实用算法通过神经网络从数据中学习模拟用户 [1, 18]。在这种场景下，环境参数空间就是神经网络的权重空间，其规模极其庞大且冗余。在这样的空间中开发提取器和策略来识别环境参数几乎是不切实际的。

在这项工作中，我们首先基于环境参数的概念对现实差距进行形式化，并分析现实差距带来的额外挑战。基于这些分析，我们构建了一种新的零样本策略迁移系统，称为**Sim**ulation- to-**Rec**ommendation（Sim2Rec），通过环境参数提取器来应对现实差距。为解决 SRS 中的环境参数提取问题，我们提出了一种分层环境参数提取器，其中包括**S**tate-**A**ction **D**istributional variational **A**uto**E**ncoder（SADAE）。该编码器基于对证据下界的理论分析，将用户组的状态-动作数据集嵌入为一个潜在向量。基于嵌入后的组信息向量，我们使用循环神经网络（RNN） [19] 来识别每个用户的参数；为应对数据驱动用户模拟器中环境参数空间极其庞大且冗余的问题，我们开发了若干使用模拟器和进行策略探索的技术，以确保该框架在真实世界 SRS 应用中的可行性。

总而言之，本文的主要贡献如下：

- 针对 SRS 中基于模拟器的强化学习方法所面临的现实差距问题，我们提出了一种零样本策略迁移方法 Sim2Rec。据我们所知，这是首项在 SRS 的策略优化中考虑模拟器现实差距的工作；
- 为在 SRS 场景中高效识别环境参数，我们提出了一种分层环境参数提取器架构，其中包含一种新的自编码器 SADAE，用于将用户组的状态 - 动作数据集嵌入为潜向量。我们还采用了多种技术将数据驱动模拟器的环境参数空间缩减到可行范围，以促进策略和提取器的学习；
- 我们在一个开源合成环境和真实网约车平台滴滴出行上开展了实验。在合成环境中的实验、离线测试和在线部署结果均证明了 Sim2Rec 的有效性。

II. 相关工作

在模拟器中训练强化学习策略，是避免在真实环境中付出高昂试错成本的理想方式 [20]。许多基于强化学习的 SRS 方法都将模拟器视为训练和测试的理想环境 [21, 22]。近期研究采用不同方法，专注于数据驱动的模拟器重构：[1, 23] 使用生成对抗框架学习模拟器，以生成与真实分布一致的数据分布； [24] 通过世界模型构建模拟器；Zhu 等人 [25] 通过因果结构模型提升世界模型的泛化能力。 [26] 使用逆倾向加权技术处理选择偏差问题，以构建去偏模拟器。Wu 等人 [27]使用真实数据集校正模拟器的表征和奖励函数，以提升其保真度。在实际应用中，重构后的模拟器不可避免地会存在现实差距，因为用户行为往往十分复杂。然而，当前研究在学习强化学习策略时并未考虑模拟器的现实差距，这可能导致不理想的真实世界表现 [13]。

另一方面，零样本策略迁移技术已广泛用于克服物理模拟器在具有挑战性的任务中的现实差距 [14, 15, 16, 28, 29, 30, 31, 32]。这些技术使用由人类专家依据物理定律构建的物理模拟器进行策略学习，并假设现实差距源于模拟器环境参数估计的误差（例如，机器人控制中的摩擦系数）。零样本策略迁移技术的范式可概括为两个阶段：（1）构建一个模拟器集合，其中包含从环境参数空间中选取的大量不同环境参数；（2）训练一种能够在该模拟器集合中采取合理动作的策略。如果环境参数空间涵盖了现实世界的环境参数，且模拟器集合遍历了该空间，那么我们可以认为，策略部署后能够像在模拟器中一样，在现实世界中做出合理决策。一种流行的策略学习方法是学习或构建在线系统辨识（OSI）模块 [14, 15, 31, 33, 34] ，从交互轨迹中推断环境参数，并构建上下文感知策略，根据推断出的参数控制智能体采取适应性行为，以获得最佳性能。部署后，OSI 模块会根据真实交互轨迹调整其推断出的环境参数，从而自动使策略适应合适的行为。 [33] 设计一种 EPI 策略来探索一些交互轨迹，并设计一个 EPI 轨迹嵌入网络，用于

环境参数表征，该网络能够预测相应模拟器的动态特性；还设计一种特定于任务的策略，根据各模拟器推断出的表征执行最佳行为。 [14, 15, 32, 34] 采用端到端架构进行环境参数表征和自适应策略学习。引入循环神经网络（ RNN），例如 LSTM [35]，，用于环境参数表征；随后，上下文感知策略根据 RNN 的输出和当前状态采取动作。在本研究中，我们沿用零样本策略迁移的基本思路以及与以往工作相同的端到端架构。我们阐述并分析了标准零样本策略迁移框架应用于 SRS 时面临的额外挑战，并提出了一种切实可行的解决方案来应对这些挑战。

III. 问题定义

我们首先对 SRS 的一般工作流程进行形式化描述。在 SRS 中，推荐系统服务于多个组 $u \in \mathcal{U}$ 中的多个用户 $g \in \mathcal{G}$、 $\mathcal{U}$ 和 $\mathcal{G}$ 分别表示用户空间和组空间。推荐策略 π 在推荐会话中以离散时间步 $t \in \{0, 1, ..., T\}$ 与这些用户进行交互，其中 T 是推荐会话的最大时间步数。在每个时间步 t，策略π 会向每个用户提供一个项目 $i \in \mathcal{I}$，并接收每个用户的反馈$y \in \mathcal{Y}$，其中 $\mathcal{I}$ 是项目空间， $\mathcal{Y}$是反馈空间。以网约车平台为例，该平台在多个城市（即组 g）提供服务，并与每个城市中的众多司机（即用户 u）进行交互。系统将设计若干项目i 供推荐。每个项目都包含一项供司机执行的任务，例如引导司机前往某些区域的派单任务。平台将接收司机的反馈y，例如完成一些订单或直接下线。

A. 马尔可夫决策过程建模

基于强化学习的推荐系统将推荐任务视为推荐系统（智能体）与用户（环境）之间的序列交互，并使用马尔可夫决策过程（MDP）对其建模 [9, 10, 36]。MDP 由五个元素组成的元组定义（$\mathcal{S}$， $\mathcal{A}$， R， P， γ， d_0），其中 $\mathcal{S}$ 和 $\mathcal{A}$ 分别是状态空间和动作空间， P: $\mathcal{S}\times\mathcal{A}\to\mathcal{S}$ 是转移函数， R: $\mathcal{S}\times\mathcal{A}\times\mathcal{S}\to\mathbb{R}$ 是均值奖励函数， $\gamma\in[0,1]$ 是折扣因子， d_0 是初始状态分布。推荐策略 π: $\mathcal{S}\to\mathcal{A}$。对于 LTE 优化，($\mathcal{S}$， $\mathcal{A}$， R， P) 设置如下：

- **状态空间** $\mathcal{S}$：状态由以下几个部分组成：用户画像特征 $s_{\text{用户}}$（例如，年龄、性别和位置）、用户的历史反馈 $s_{\text{历史}}$（例如，订单履行次数和在线时长）及其统计量 $s_{\text{统计}}$（例如，最近 7 天和 14 天的平均订单履行次数）、用户所在组的一些外部特征 $s_{\text{组}}$（例如，城市信息），以及一些与时间步相关的特征 $s_{\text{时间}}$（例如，天气）。

- **动作空间** $\mathcal{A}$：我们不再将 a 设为 $\mathcal{I}$ [7, 9]中物品的索引，而是将动作 a 定义为一组能够确定推荐物品的参数 i 来自 $\mathcal{I}$，这与 [1,3]等先前研究相同。具体而言，我们定义了一个基于规则的函数F： $\mathcal{A}\to\mathcal{I}$。例如，对于每个时间步 t， $\pi(a_t|s_t)$会确定每位司机的任务 a_t 难度系数，随后 $F(a_t)$会找出对应的项目条目i_t，并将其推荐给司机。

- **奖励函数** R：对于每个时间步 t，我们根据当前状态 s_t、采取的动作 a_t 以及用户反馈（在 s_{t+1}中），定义即时参与度指标 $R(s_t, a_t, s_{t+1})$。然后，我们将 LTE 指标定义为 $\sum_{t=0}^{T} R(s_t, a_t, s_{t+1})$，并忽略延迟指标 [9]以简化问题。

- **转移函数** P： $P(s_{t+1}|s_t, a_t)$ 定义了采取动作 a_t后状态从 s_t 转移到 s_{t+1} 的过程。

B. 基于模拟器的强化学习用于 LTE 优化

本文遵循基于模拟器的强化学习用于 LTE 优化的一般流程，如 [1, 3, 10]所示。我们首先定义一个用户模拟器 M： $\mathcal{S}\times\mathcal{A}\times\mathcal{S}\to\mathcal{Y}$。具体而言，用户模拟器的目标可形式化定义如下：给定一个状态-动作对 (s, a)，根据状态 s，模仿用户对推荐动作 a 的反馈（行为） y。对于每个时间步 t，给定预测值 $\hat{y}_{t+1}$，我们首先通过 $\hat{y}_{t+1}$更新 s^{hist}_{t+1} 和 s^{stat}_{t+1}，然后从日志数据集 $\mathcal{D}$中的真实轨迹载入 s^{user}_{t+1}、s^{group}_{t+1}、s^{time}_{t+1} 和 τ^{r}，其中 $\tau^{\text{r}} := [s^r_0, a^r_0, s^r_1, a^r_1, ..., s^r_T, a^r_T]$。最后，我们得到 $s_{t+1} = [s^{\text{hist}}_{t+1}, s^{\text{stat}}_{t+1}, s^{\text{user,r}}_{t+1}, s^{\text{group,r}}_{t+1}, s^{\text{time,r}}_{t+1}]$和奖励$r_t = R(s_t, a_t, s_{t+1})$。我们将记号 $P_{M,\tau^{\text{r}}}(s'|s, a)$ 定义为基于 M 和 τ^{r}的上述转移过程。注意，与其直接预测整个下一个状态 s_{t+1}，模拟器只预测过去的 y，并根据历史数据 τ^{r}构建其他状态。

基于模拟器的强化学习的总体目标是找到一个最优策略 $\hat{\pi}^*$，使所有用户的累积奖励（即 LTE）最大化。具体而言，目标函数写作：

$$\max_{\pi} \mathbb{E}_{g\sim p(g), u\sim p(u), \tau^{\text{r}}\sim\mathcal{D}(u,g)} \left[\mathbb{E}_{\tau\sim p(\tau|\pi, P_{M,\tau^{\text{r}}})} \left[\sum_{t=0}^{T} \gamma^t r_t \right] \right], \tag{1}$$

其中， $p(g)$ 和 $p(u)$ 分别是组和用户的先验分布， $\tau^{\text{r}} \sim \mathcal{D}(u, g)$ 表示从日志数据集 $\mathcal{D}$ 中采样组 g 内用户 u 的真实轨迹，而 $p(\tau|\pi, P_{M,\tau^{\text{r}}})$ 表示在策略 π 和状态转移函数 $P_{M,\tau^{\text{r}}}$ 下生成轨迹 τ: $= [s_0, a_0, r_0, ..., a_{T-1}, s_T, r_T]$ 的概率。具体而言，

$$p(\tau \mid \pi, P) := d_0(s_0) \prod_{t=0}^{T} P(s_{t+1} \mid s_t, a_t)\pi(a_t|s_t), \tag{2}$$

其中， $d_0(s_0)$ 是初始状态分布。

C. SRS 中基于模拟器的强化学习的现实差距

我们首先定义真实用户模型 $\mathcal{E}$，它会输出用户的真实反馈。由于用户 $u \in \mathcal{U}$ 有自己的行为模式，而且也取决于他/她所属的组 $g \in \mathcal{G}$，因此我们定义两个函数 $F_u(u)$ 和 $F_g(g)$，将这些个体映射到相应的行为参数

模式。然后，我们可以将真实用户模型构造为 $\mathcal{E}(y|s,\ a,\ F_u(u)\ F_g(g))$。真实最优策略 π^* 是使下式最大化的策略：

$$\max_{\pi} \mathbb{E}_{g\sim p(g),u\sim p(u)} \left[\mathbb{E}_{\tau\sim p(\tau|\pi,P_{\mathcal{E}})} \left[\sum_{t=0}^{T} \gamma^t r_t \right] \right]. \quad (3)$$

假设我们拥有正确的先验分布 $p(u)$和 $p(u)$，这是一个较弱的假设，因为部署时我们可以轻松控制所推荐用户的范围。那么，我们可以看到，现实差距源于 $P_{\mathcal{E}}$ 与$P_{M,\ \tau^{\mathrm{r}}}$之间的不匹配，这使得 $\pi^*\ \neq\ \hat{}\pi^*$。如果 $P_{\mathcal{E}}$ 与$P_{M,\ \tau^{\mathrm{r}}}$ 之间的转移差距较大，那么 π^* 与$\hat{}\pi^*$之间的性能差距也会很大 [37]。此外，单步预测误差会在多步滚动过程中累积，最终导致性能差距进一步扩大 [38]。

本文的主要符号总结于表 I。

表 I 主要符号。

π	推荐策略
ϕ	用户模型的环境参数提取器
z	使用 ϕ推断出的环境参数
F_g 和 F_u	将组 g 和用户 u 映射到对应行为模式参数的函数
$\mathcal{E}$	真实用户反馈模型
M_ω	由 ω参数化的用户模拟器
Ω	ω的参数空间
$\mathcal{H}$	用户-模拟器学习算法
X_t^g	组 g 在时刻 t 之前的状态-动作轨迹
ψ_t^g	生成该分布的参数
	状态-动作对在 X_t^g
θ	SADAE 中后验近似的参数
κ	SADAE 中推断过程的参数

IV. 从模拟到推荐

A. 零样本策略迁移框架

本节将零样本策略迁移技术引入 SRS。标准的零样本策略迁移技术已广泛用于克服具有挑战性的任务中物理模拟器的现实差距 [14, 15, 16]。这些技术假设现实差距源于环境参数之间的差异 ω。通常，它们首先从环境参数空间 Ω 构建一个模拟器集合，其中包含大量环境参数各不相同的模拟器ω 。基于该模拟器集合，它们学习一个提取器 ϕ 来推断环境参数，并学习一个上下文感知策略 π 来控制智能体，使其根据推断出的参数采取适应性行为，从而实现最佳性能 [14, 15]。部署后，提取器会利用真实交互轨迹信息调整推断出的环境参数，从而自动使策略适应合适的行为。

在 SRS 中，由于用户的行为通常难以通过物理规则建模，许多实际应用会采用数据驱动技术来学习预测这些行为 [1, 3, 10, 26]。在这里，我们假设用户模拟器 M_ω 由 ω 参数化，而 ω 是通过用户模拟器学习算法 $\mathcal{H}$学习得到的。然后，$\mathcal{E}$、 F_u 和 F_g 基于 $\mathcal{H}$由 ω 隐式表示。

现在，我们将标准的零样本策略迁移框架应用于 SRS [14, 15, 16, 29, 39, 40]。形式上，我们提出以下目标函数来处理现实差距问题：

$$\max_{\pi} \mathbb{E}_{\omega\sim p(\Omega)} \left[\mathbb{E}_{\tau^{\mathrm{r}}\sim p(\tau^{\mathrm{r}}),\tau\sim p(\tau|\pi,\phi,P_{M_\omega,\tau^{\mathrm{r}}})} \left[\sum_{t=0}^{T} \gamma^t r_t \right] \right],$$

其中，Ω 是 M 的参数空间， $p(\Omega)$ 是用于生成模型参数的采样策略， $p(\tau^{\mathrm{r}})$ 是对过程 $\tau^{\mathrm{r}}\ \sim\ \mathcal{D}(u,g)\ u\ \sim\ p(u)$ $g\ \sim\ p(g)$（见式 (1)）的简化， $p(\tau|\pi,\phi,P_{M_\omega,\tau^{\mathrm{r}}})$ 表示基于上下文感知策略 π 和环境参数提取器 ϕ的 rollout 过程：对于每个时间步 t，我们首先推断当前用户模型 M_ω 的环境参数 $z\ =\ \phi(M_\omega)$（稍后将讨论 ϕ 的具体输入），其中 z 是用户模型 M_ω 的表征；随后，上下文感知策略 $\pi(a|s,z)$ 将根据推断出的表征z采取动作。上下文感知策略 π 经过训练，可在所有满足 $\omega\ \in\ \Omega$ 的模型 M_ω 中做出最优决策。部署时，我们使用同一个提取器推断真实世界 $z_r\ =\ \phi(\mathcal{E})$ 的表征，然后上下文感知策略 ϕ 根据 z_r 做出决策： $a\ \sim\ \hat{}\pi^*(a|s,z_r)$。如果 $\mathcal{E}$ 可以由 ω 表示，即 $\exists\omega\ \in\ \Omega\ M_\omega\ \approx\ \mathcal{E}$，并且 ϕ 能够正确识别参数的表征，那么我们有 $\hat{}\pi^*(a|s,z_r)\ \approx\pi^*(a|s)\ \forall s\in\mathcal{S}$ [14, 15, 39]。

然而，由于以下两个方面的原因，上述解决方案在实践中不可行：

(1) Ω 的**参数空间极其庞大**：在以往零样本策略迁移技术的应用中 [14, 15,39, 40, 41]，P 通过物理原理构建，并使用具有明确含义的参数 ω，例如摩擦系数或机械臂的长度。因此，Ω 的空间对于ϕ 和 π 学习而言较为紧凑。在 SRS 中，模拟器通过数据驱动技术构建，因此 ω 较为复杂，例如神经网络的权重。因此，Ω 的空间庞大且冗余。目前，几乎不可能直接在如此庞大的空间中开发 ϕ 和 π 来识别 ω 。要为 SRS 开发实用的零样本策略迁移技术，我们首先应将 Ω 缩减至可行的规模；

（2）**较高的 ϕ 识别正确表征**：在以往的应用中，策略用于操作单个机器人（例如四足机器人 [41], 机械臂 [14], 或机械手 [15]）。它们只需要识别所部署机器人的参数。因此，一些实际的在线搜索方法可以通过某些在线交互样本 [40]直接搜索正确的参数，这在实践中是可行的。在 SRS 中，策略同时服务于多个地区的众多用户。因此，为所有用户搜索参数的计算成本将非常高，在大型互联网平台上将难以接受。另一种范式是表征学习：它们训练环境参数提取器 ϕ，将智能体的历史交互样本嵌入为隐藏变量 z。循环神经网络（RNN）通常用于将序列信息嵌入环境参数向量$z_t\ =\ \phi$（s_t， a_{t-1}， z_{t-1}）。理论上，在嵌入足够多的交互样本后，目标环境是可识别的。该

流程更适合 SRS 场景，因为部署时端到端推理模块 ϕ 的计算成本更低。然而，在 SRS 中，用户的反馈不仅取决于用户画像（F_u 见公式（3）），还取决于用户所在地区（F_g 见公式（3））。如果只考虑单个用户的交互，则需要更多时间步的交互来进行识别 z，这会导致部署时产生额外的决策风险，因为策略需要经过更多步骤的探索，才能为每个组中的每位用户识别出最优策略 [42]。

如上所述，ϕ 的表征学习是一种有望解决 SRS 中现实差距问题的范式。本文遵循这一范式，并提出了几种实用技术来应对上述挑战。形式上，为了找到适用于 ϕ^* 的最优提取器和策略 π^*，标准目标函数 [14, 15] 为：

$$\max_{\pi,\phi} \mathbb{E}_{\omega\sim p(\Omega)} \left[\mathbb{E}_{\tau^{\mathrm{r}}\sim p(\tau^{\mathrm{r}}),\tau\sim p(\tau|\pi,\phi,P_{M_\omega,\tau^{\mathrm{r}}})} \left[\sum_{t=0}^{T} \gamma^t r_t \right] \right], \tag{4}$$

其中，$p(\Omega)$ 表示从模拟器参数集 Ω 中抽取转移函数 M_ω 的采样策略，满足 $P[\omega] > 0,\ \forall \omega \in \Omega$。以下分析中我们采用均匀采样策略。对于每个时间步 t，我们首先通过 $z_t = \phi(s_t, a_{t-1}, z_{t-1})$ 推断环境参数，其中 s_t 是 $P_{M_\omega}(s|s_{t-1}, a_{t-1})$ 的一个样本；随后，上下文感知策略 $\pi(a|s_t, z_t)$ 将基于推断得到的表征 z_t 采取动作。ϕ 和 π 通过公式 (4) 共同优化。请注意，如果不同模拟器中的最优策略不一致，但具有相同的 π，则梯度将从 z 反向传播至 z，从而 ϕ 的参数会自动更新，以识别 Ω 中的参数。

在本文接下来的部分中，我们首先在第 IV-B 节提出一种新的 ϕ 环境参数提取器架构，以便在 SRS 场景中更高效地进行参数识别。在第 IV-C 节中，我们提出了若干技术，将 Ω 缩减至适用于 π 和 ϕ 学习的规模。

B. 分层环境参数提取器

在 SRS 中，环境参数取决于用户和群组信息 u 和 g。若已知 u 和 g的真实特征，我们可以将其输入 ϕ：$z_t = \phi(s_t, a_{t-1}, z_{t-1}, g, u)$，以解决表征识别问题。然而，$\mathcal{G}$和 $\mathcal{U}$ 的一些特征难以建模，这种情况不可避免。因此，除了通过与 $\mathcal{G}$ 和 $\mathcal{U}$相关的特征工程来构建静态状态，即 s^{user} 和 s^{group}之外，我们还设计了一种分层架构，用于

用于建模用户和群组信息的提取器。直观而言，我们应将群组轨迹 $S_0^g, A_0^g, S_1^g, A_1^g, ..., S_t^g$ 添加到输入中，即 $z_t = \phi(s_t, a_{t-1}, S_t^g, A_{t-1}^g, z_{t-1})$，其中 $(S_t^g, A_{t-1}^g) := \{(s_t^{(i)}, a_{t-1}^{(i)})\}_{i=1}^N$（在本文其余部分，为简洁起见，我们使用 $X_t^g := (S_t^g, A_{t-1}^g)$），其中包含每个时间步的 N状态-动作对 t。

然而，用户数量 N 可能很大。直接将 X_t^g 输入神经网络并不实际。我们更倾向于将 X_t^g 嵌入为一个低维向量 υ，再输入 ϕ。计算 X_t^g 的统计量（例如，均值和标准差）是一种直接的方法，但会限制 υ的表征容量。注意力机制等热门模块 [43] 是可行的选择，但这些模块的计算成本较高。

在这项工作中，我们提出了一种简单的方法，在给定 X_t^g 的情况下推断潜在嵌入 υ，称为受变分自编码器（VAE）[44] 启发的 **状**态-**动**作 **分**布式变分 **自**动**编**码器（SADAE）。我们首先基于以下假设构建数据生成过程：首先，X_t^g 中的状态-动作对是从分布 $p_{\psi_t^g}(s, a)$ 中独立同分布采样得到的，该分布由每个时间步 t 和组 g 对应的 ψ_t^g 参数化。其次，分布的参数 ψ 由分布 $p_\theta(\psi \mid \upsilon)$ 生成，该分布由 θ 参数化。它涉及一个潜在连续随机变量 υ，该变量由先验分布 $p(\upsilon)$ 生成。X 的生成包含三个步骤：(1) 从 $p(\upsilon)$ 中采样 υ；(2) 从分布 $p_\theta(\psi \mid \upsilon)$ 中采样 ψ；(3) 重复采样 $p_\psi(s, a)$ 以生成 X。图1展示了其与有向图模型上的 VAE 的比较。

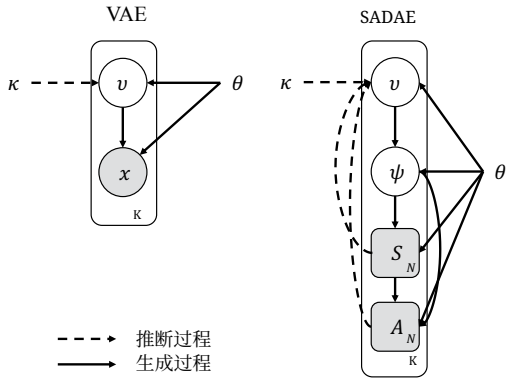


图1. 通过有向图模型比较 SADAE 和原始变分自编码器。圆形表示变量节点。圆角矩形表示数据集节点，其中角落处的符号表示数据集的数量。θ 表示生成模型的近似参数，κ表示变分近似模型的参数，K 表示 $\mathcal{D}$中 S 和 A 的样本数量，N 表示 S 和 A中 s 和 a 的样本数量。

形式上，我们的目标是学习一个由 κ 参数化的嵌入模型$q_\kappa(\upsilon \mid X)$，使其与后验近似 $p_\theta(\upsilon \mid X)$ 保持一致。使用库尔贝克-莱布勒散度（KLD）进行度量，目标函数可写如下：

$$\min_{\kappa,\theta} \mathbb{E}_{X\sim\mathcal{D}} \left[KLD\left(q_\kappa(\upsilon \mid X) \,||\, p_\theta(\upsilon \mid X)\right) \right], \tag{5}$$

其中，数据集 $\mathcal{D}$ 被重塑为 $\{X_t^g :\ g \in \mathcal{G}, 0 < t \le T\}$包含所有时间步和组中的状态-动作对，而后验分布 $p_\theta(\upsilon \mid X)$ 是 $q_\kappa(\upsilon \mid X)$ 的目标分布。为简洁起见，我们分别用 θ 和 κ 表示后验近似和推断的所有参数。

我们首先在引理4.1中给出式（5）的证据下界（ELBO）。

引理*4.1*：状态-动作分布变分推断目标函数（公式（5））的证据下界为：

$$\max_{\kappa,\theta} \mathbb{E}_{X\sim\mathcal{D}} \left[\mathbb{E}_{q_\kappa(\upsilon|X)} \left[\log p_\theta(X \mid \upsilon) \right] - KLD\left(q_\kappa(\upsilon \mid X) \| p_\theta(\upsilon)\right) \right].$$

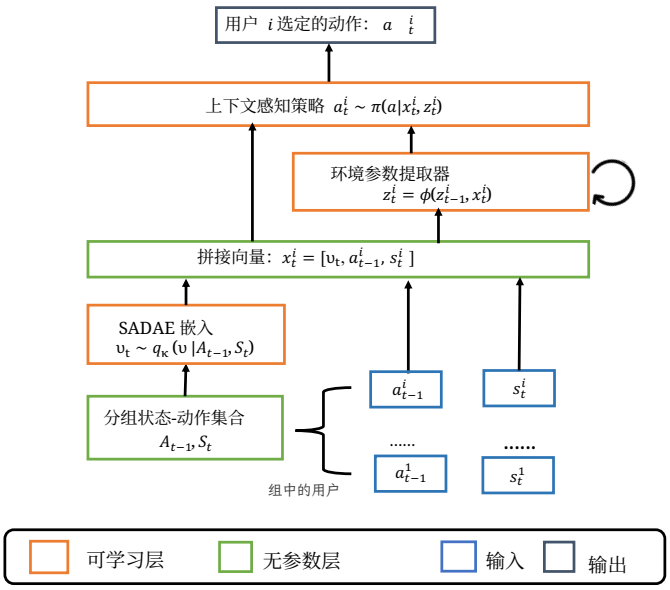


图 2. Sim2Rec 的整体架构。

在 υ 下，X q_κ $\upsilon \mid X$ p_θ $X \mid$ 独立同分布的假设下，（）和（）可以通过似然进行估计：

$$q_\kappa(\upsilon \mid X) = \prod_{i=1}^{N} q_\kappa\left(\upsilon \mid s^{(i)} a^{(i)}\right), \tag{6}$$

$$p_\theta(X \mid \upsilon) = \prod_{i=1}^{N} p_\psi\left(s^{(i)}, a^{(i)}\right) p_\theta(\psi \mid \upsilon), \tag{7}$$

其中，ψ 表示分布 p_ψ 的参数。我们在定理 4.1 中给出了可处理的证据下界（ELBO）定理。证明见附录 A。

定理 *4.1*：状态-动作分布变分推断的可计算证据下界为：

$$\mathbb{E}_{X\sim\mathcal{D}, q_\kappa(\upsilon|X)}\left[\sum_{i=1}^{N} \log p_\theta\left(s^{(i)} \mid \upsilon\right) + \log p_\theta\left(a^{(i)} \mid \upsilon, s^{(i)}\right)\right]$$

$$-KLD\left(q_\kappa(\upsilon \mid X) \,||\, p_\theta(\upsilon)\right). \tag{8}$$

定理 4.1 为我们提供了一个三步流程，用于最小化公式 (5) 中的目标函数：(1) 从数据集 $\mathcal{D}$ 中采样一批 X；(2) 根据公式 (6) 推断潜在编码 υ；(3) 根据公式 (7) 计算状态-动作对的重构对数概率，并计算 υ 的后验分布与先验分布之间的 KL 散度，然后将梯度应用于 κ 和 θ。最后，提取器 ϕ 使用 s、a 和 υ 推断环境参数：$z_t = \phi(s_t, a_{t-1}, \upsilon_t, z_{t-1})$，其中 $\upsilon_t \sim q_\kappa(\upsilon \mid X_t)$。随后，上下文感知策略 $\pi(a_t \mid s_t, z_t)$ 根据 z_t 采样一个动作。q_κ 也根据公式 (4) 更新。梯度将从 ϕ 反向传播至 υ，以更新 κ。整体架构如图 2 所示。

C. 可行参数空间Ω 构建

考虑一个基于神经网络的数据驱动用户模拟器，Ω 的原始参数空间将是神经网络的权重空间，其规模会极其庞大且复杂。

然而，Ω 中的许多权重 $\omega \in$根本无法模仿用户的反馈。没有必要让 ϕ 和 π了解 Ω 中的所有权重。从这一角度来看，用户模拟器模仿算法 $\mathcal{H}$ [1, 3, 10] 可被视为一种实用方法，用于采样一个 ω，使其接近真实世界的参数 ω^*。具体而言，我们有 $\omega = \mathcal{H}(\mathcal{D}, \lambda)$，其中 $\mathcal{D}$ 是用于用户模拟器学习的数据集，λ 是学习算法 $\mathcal{H}$的超参数（例如随机种子和学习率）。使用不同的 $\mathcal{D}$ 和 λ，$\mathcal{H}$ 将生成一个权重向量 ω。

受集成技术的启发，在本研究中，我们构建了一个收缩参数空间 $\Omega' := \{\omega: \mathcal{H}(\mathcal{D}', \lambda)\ \lambda \in \Lambda,\ \mathcal{D}' \subseteq \mathcal{D}\}$，其中 $\mathcal{D}'$ 是 $\mathcal{D}$ 的一个子集，而 Λ 是为 $\mathcal{H}$ 学习所选定的超参数空间。通过这种方式，我们可以生成一个权重集合，其中 $\omega \in \Omega'$ 与 ω^* 大致接近，并具有合适的 $\mathcal{D}'$ 和 λ。

然而，我们仍然无法断言 $\omega^* \in \Omega'$，因为用户行为过于复杂，无法被精确预测。所有ω 都可能在某些状态和动作下存在预测误差。一般而言，学习误差包括以下两个方面：（1）近似误差：数据集$\mathcal{D}$ 中的近似误差受到神经网络模型容量和学习工具的限制；此外，在序贯环境中，即使单步预测误差较小，近似误差也会在每个步骤中不可避免地累积，导致模拟轨迹出现较大偏差 [37]；（2）外推误差：由于这些模型被用作策略训练的模拟器，我们期望模型在使用数据收集动作以外的其他动作进行查询时，能够给出无偏预测。这使得模型在数据分布中的学习和使用违反了独立同分布（独立同分布）假设，并导致外推误差。因此，模型在未见动作（即反事实动作）下的预测可能出现灾难性失效，完全无法为策略学习提供正确的指导 [11, 23]。

虽然我们无法直接使 $\omega \in \Omega'$ 成立，但可以干预强化学习的探索过程，避免策略在 M_{ω^*} 与 M_ω差距较大的区域进行学习。为提高通用性，本文设计了几种不依赖特定模型学习技术的后处理方法，以解决上述问题：

避免策略利用预测误差较大的区域：为避免智能体进入很可能产生错误预测的区域，我们在每一步根据模型不确定性 $U(s_t, a_t)$ [37]计算惩罚，并将其加入奖励中。模型不确定性U 衡量已学习的转移模型在 (s_t, a_t) 处预测结果的不一致程度；为减轻模型的累积误差，我们从日志数据集中随机抽取一个状态作为初始状态，并将最大展开长度限制为固定值 T_c。上述解决方案的灵感来自 [37, 42]，它们是在 MuJoCo [45] 中使用的离线基于模型的强化学习算法。

确保策略在外推误差较小的区域内进行优化：在 LTE 优化问题中，我们通常对用户反馈的趋势有先验知识

算法 1 Sim2Rec 伪代码

输入:
ϕ_φ 作为环境参数提取器，参数化为φ；上下文感知策略 π_ι 参数化为 ι；状态-动作分布嵌入 q_κ；日志数据集 $\mathcal{D}$；不确定性惩罚系数： α；截断式展开时域 T_c；模型不确定性函数 U；

流程:

1：构建参数集 $\mathbf{\Omega}' := \{\omega : \mathcal{H}(\mathcal{D}', \lambda)\ \lambda \in \Lambda, \mathcal{D}' \subseteq \mathcal{D}\}$；2：初始化一个空缓冲区 $\mathcal{D}_{\text{rollout}}$；3：对于 1、2、3、.....，执行以下操作：4：采样一个模拟器 M_ω，其中 $\omega \sim p(\mathbf{\Omega})$。5：选择一个组 $g \sim p(g)$。6：从组 g中采样真实轨迹 $\tau^r \sim p(\tau^R)$，并采样具有截断时域 T_c的模拟轨迹 $\tau \sim p(\tau|\pi, \phi, P_{M_\omega, \tau^r})$。7：将轨迹 τ 添加到 $\mathcal{D}_{\text{rollout}}$。8：将不确定性惩罚 $U(s,a)$添加到$\mathcal{D}_{\text{rollout}}$中的奖励中，即 $r \leftarrow r - \alpha U(s,a)$。9：通过 $\mathcal{D}_{\text{rollout}} \leftarrow F_{\text{trend}}(\mathcal{D}_{\text{rollout}})$ 过滤$\mathcal{D}_{\text{rollout}}$ 中的数据，并通过 $\mathcal{D}_{\text{rollout}} \leftarrow F_{\text{exec}}(\mathcal{D}_{\text{rollout}})$更新完成状态和奖励值。10：使用强化学习算法之一，通过式（4）更新 φ、 θ、 ι 和 κ，其中使用 $\mathcal{D}_{\text{rollout}}$，并通过式（8）更新 κ。11：结束循环。

针对特定应用中的动作变化。例如，在需求预测中，如果价格上涨，用户需求就会下降。利用关于弹性的先验知识，我们可以评估模型对反事实动作的预测，并删除其中用户模拟器 M 的预测与趋势先验不一致的轨迹$\mathcal{D}$ 。我们用 $\mathcal{D} \leftarrow F_{\text{trend}}(\mathcal{D})$ 表示这一筛选过程。此外，我们为每个状态定义可执行动作子空间，以避免策略采取偏离数据收集策略 π_e 过远的动作。例如，在我们的应用中，我们计算历史交互中针对用户 u 曾采取过的最小和最大动作值 $a^u_{\min}$, $a^u_{\max}$。如果策略 $a \notin (a^u_{\min}, a^u_{\max})$ 的输出，为避免策略采取高风险动作，可以将该状态设为终止状态，即 done $= \mathbb{I}[a \notin (a^u_{\min}, a^u_{\max})]$,，并将奖励设为 $\frac{R_{\min}}{1-\gamma}$，其中 $R_{\min}$ 是任务的最小奖励。我们用 $\mathcal{D} \leftarrow F_{\text{exec}}(\mathcal{D})$ 表示这一过程。

基于上述技术，我们在算法 1 中给出了 Sim2Rec 的伪代码。

V. 实验

本节首先在 Google RecSim 中一个名为长期满意度（LTS）的合成推荐环境中开展实验 [1] [22]。随后，我们将 Sim2Rec 应用于大型网约车平台滴滴出行的司机程序推荐（DPR）任务，以展示所提出方法在真实场景中的有效性。具体而言，我们主要关注以下问题：

- **RQ1:** SADAE 能否有效地重建群组信息?
- **RQ2:** 在具有预定义可行环境参数空间的合成环境中，IV-B 节提出的提取器架构能否更高效地识别环境?
- **RQ3:** IV-C 节提出的为数据驱动模拟器构建可行参数空间的技术是否适用于现实世界应用?
- **RQ4:** Sim2Rec 策略在真实数据任务的未见环境中，能否比基准推荐系统取得更好的性能?
- **RQ5:** 整个系统在大规模生产环境中的表现如何?

下文将在第 V-B3 节和第 V-C4 节回答**RQ1** ，在第 V-B4 节回答**RQ2** ，在第 V-C5 节回答**RQ3** ，在第 V-C6 节回答**RQ4** ，并在第 V-D 节回答**RQ5** 。

此外，我们还进行了消融实验，以验证第 IV-B 节提出的 SADAE 和第 IV-C 节提出的后处理方法的必要性。关于 SADAE 和后处理方法的实验结果分别在第 V-B4 节和第 V-C5 节进行了分析。

A. 实验设置

1) 实现细节： 我们使用近端策略优化（PPO） [46] 作为策略学习方法来优化公式（4）。环境上下文提取层采用单层LSTM网络 [35]建模。我们在SADAE q_κ 的嵌入与环境参数提取器 ϕ之间添加额外的全连接层 f。我们在两个实验中使用相同的网络结构和超参数，但神经网络的复杂度不同。表II报告了超参数。

表 II Sim2Rec 的超参数。

超参数	LTS	DPR
策略和提取器学习		
学习率	从 1e-4 到 1e-6	
优化器	Adam	
折扣因子 γ	0.99	0.9
预测时域 T	140	30
批次大小	30000	120000
额外的全连接层 f	[128, 128, 128,32]	[512, 512, 256]
LSTM 单元数（在 ϕ	64	256
上下文感知层 π	[128, 64]	[512, 256]
SADAE 学习		
嵌入层 $q_\kappa(\upsilon\|s,\ a)$	[512, 512]	
重构层 $p_\theta(\psi\|\upsilon)$	[512, 512]	
优化器	Adam	
学习率	2e-5	1e-6
L2 正则化权重	0.1	0.001
潜在编码单元数	5	200

[1]我们在 https://github.com/xionghuichen/Sim2Rec 发布了代码

2）基线方法：我们将我们的方法 Sim2Rec 与以下基线方法进行比较：

- **DR-OSI**：一种采用标准LSTM神经网络作为环境参数提取器，以实现零样本策略迁移的OSI算法 [15]；与Sim2Rec相比，DR-OSI在神经网络架构中不采用SADAE进行提取器学习。
- **DR-UNI**：一种用于学习统一策略的领域随机化技术 [29]。这是一种替代性的零样本策略迁移方法，从模拟器集合中学习保守策略。DR-UNI可视为一种与公式4具有相同目标函数的策略学习方法，但 ϕ 的输出为常数。
- **DIRECT**：一种不考虑模拟器现实差距的标准基于模拟器的策略学习方法 [1];
- **WideDeep**：一种用于推荐系统的监督学习模型，利用宽层和深层来平衡记忆与泛化 [47];
- **DeepFM**：同样是一种采用监督学习方法的推荐系统学习算法，引入因子分解机层来替代WideDeep中的宽层 [47]，并采用深度神经网络构建混合结构，以发挥低阶和高阶特征交互的优势 [48];
- **Sim2Rec-PE**：Sim2Rec 算法未使用处理预测误差的技术;
- **Sim2Rec-EE**：不使用这些技术的 Sim2Rec 算法，包括两个用于处理外推误差的滤波器 $F_{\text{趋势}}$ 和 $F_{\text{外推误差}}$。

3）评估指标：我们使用 KL 散度评估 SADAE 的重构数据分布与真实数据分布之间的距离，并使用标准指标长期奖励来评估所学习策略的性能。

KL 散度 (KLD)：由于状态-动作空间的维度较高，且分布复杂，尤其是在DPR任务中，我们使用核密度估计 (KDE) [49]来估计真实数据和重构数据的概率密度函数 (PDF)。然后基于此计算两个数据集的KLD。具体而言，

$$KLD(\mathcal{D}_a, \mathcal{D}_b) = \frac{1}{||\mathcal{D}_a||} \sum_{x \in \mathcal{D}_a} \log \frac{f_a(x)}{f_b(x)}, \tag{9}$$

其中，$||\mathcal{D}_a||$ 表示数据集中的样本数，f_a 和 f_b 分别表示通过 KDE 估计的真实数据和重构数据的概率密度函数。

奖励：长期奖励根据公式 3 计算。在 LTS任务中，我们为每组抽取 750 名用户来计算长期奖励。在 DPR任务中，我们选择每组的所有司机来计算长期奖励。

B. 合成环境中的实验

为了更好地量化 Sim2Rec 的适应性，我们首先在 syntheticLTS 模拟器中进行实验，其中环境参数 ω 可配置 [22]。

1) 合成模拟器：长期满意度（Choc/Kale）问题源自 Google RecSim 框架 [22] 中的一个合成环境。在该环境中，推荐系统向用户发送内容项，目标是在多个时间步内最大化用户的参与度。内容项通过标题党程度分数来表征。用户的参与度由内容的标题党程度分数和长期满意度分数共同决定。较高的标题党程度分数会直接带来更高的参与度，但会降低长期满意度；而较低的标题党程度分数则会提高满意度，但会直接导致较低的参与度。此外，长期满意度是一个用于重新缩放给定内容项参与度的系数。

形式化地说，用户 i 在时间步 t 的参与度值是从高斯分布 $\mathcal{N}\left(\mu_t^i, {\sigma_t^i}^2\right)$ 中采样得到的，该分布由 μ_t^i：$= \left(a_t^i \mu_c^i + \left(1 - a_t^i\right) \mu_k^i\right) SAT_t^i$ 和 σ_t^i：$= \left(a_t^i \sigma_c^i + \left(1 - a_t^i\right) \sigma_k^i\right)$ 参数化，其中 i 表示用户的索引，a_t^i 表示待推荐文档项的标题党程度分数。μ_c^i、μ_k^i、σ_c^i 和 σ_k^i 是用户 i 的隐藏状态。SAT_t^i 表示长期满意度分数，其通过 a_t 更新：

$$\begin{aligned} SAT_t^i &:= \text{sigmoid}\left(h_s^i \times NPE_t^i\right) \\ NPE_t^i &:= \gamma_n^i NPE_{t-1}^i - 2\left(a_t^i - 0.5\right), \end{aligned}$$

其中，NPE_t^i表示用户 i的净正向曝光分数，γ_n^i表示 NPE_t^i的记忆折扣，h_s^i表示 NPE对满意度的敏感度比率。γ_n^i和h_s^i 也是该环境中的状态。状态 μ_c^i、μ_k^i、σ_c^i、σ_k^i、h_s^i 和 γ_n^i 定义了环境参数。为了构建一个包含多个组和多个用户的环境，我们选择 μ_c^u 作为群组特征 g，该特征在模拟器中的所有用户之间保持相同。也就是说，对于所有用户 i，均有 $\mu_c^i = \mu_c$。σ_c^i、σ_k^i、h_s^i 和 γ_n^i 是用户特征。具体而言，用户特征 $u = [\sigma_c, \sigma_k, h_s, \gamma_n, \mu_k]$ 和群组特征 $g = [\mu_c]$。在初始化时，我们从均匀分布中为每个用户随机采样 h_s^i 和 γ_n^i，并使 σ_c^i、μ_k^i 和 σ_k^i 在所有用户和组之间保持相同。然而，每个用户的观测状态 s 仅包含SAT$_t^i$和 $o_i \sim \mathcal{N}$（μ_c，4），观测到的用户反馈 y 则定义为SAT$_{t+1}^i$。我们使用$\mathcal{E}$（$y|s$，a，u，g）表示上述过程。我们定义参数空间 ω：$= [\omega_u, \omega_g]$、$\mu_{c,r} = 14$、$\mu_{k,r} = 4$，以及两个映射函数 F_{ω_u}（u）$= [\sigma_c, \sigma_k, h_s, \gamma_n, \mu_{k,r} + \omega_u]$ 和F_{ω_g}（g）$= [\mu_{c,r} + \omega_g]$。然后，我们可以定义用户模拟器M_ω（$y|s$，a）：$= \mathcal{E}$（$y|s$，a，F_{ω_u}（u）F_{ω_g}（g）），并将 ω^*：$= [0, 0]$作为部署时的“真实”环境。

现在，我们可以通过直接选择ω_g 来构建训练模拟器集合，并控制训练集与目标环境之间的 ω_g 差异，以设计不同的任务。具体而言，我们使用 ω^* 构建目标模拟器，并通过在参数空间中等距采样ω_g 来选择训练模拟器集合，同时移除那些 $|\ \omega_g - \omega^*\ | < \alpha$。$\omega_g$控制群体行为。$\alpha$越大，训练集与目标环境之间的群体行为差异就越大。具体而言，我们构建以下任务：

- LTS1: $\Omega = \{\omega : | \omega_g | \geq 2 \wedge 6 \leq \mu_c + \omega_g < 22, \omega_g \in \mathbb{N}, \omega_u = 0\}$;
- LTS2: $\Omega = \{\omega : | \omega_g | \geq 3 \wedge 6 \leq \mu_c + \omega_g < 22, \omega_g \in \mathbb{N}, \omega_u = 0\}$;
- LTS3: $\Omega = \{\omega : | \omega_g | \geq 4 \wedge 6 \leq \mu_c + \omega_g < 22, \omega_g \in \mathbb{N}, \omega_u = 0\}$;
- LTS3-β: $\Omega = \{\omega : | \omega_g | \geq 4 \wedge 6 \leq \mu_c + \omega_g < 22, \omega_g \in \mathbb{N}, \omega_u \in [-\beta, \beta]\}$;

其中 $\sigma_c^i = \sigma_k^i = 1$ 适用于所有任务。为简化起见，在 LTS1 至 LTS3 中，我们仅考虑 ω_g的现实差距。

2）实现：在 LTS环境中，g 仅与群体状态信息 S相关。因此，我们训练 SADAE 来重构状态分布，而不是状态-动作分布。对于 LTS3 中的每个模拟器，我们抽取 1000 个用户，构建状态数据集 $\mathcal{D}$。$q_\kappa\left(\upsilon|s^{(i)}\right)$ 是一个神经网络，用于输出 υ的高斯分布参数。我们还使用神经网络对 $p_\theta\left(\psi_s|\upsilon\right)$进行建模，该网络输出高斯分布的参数。 υ 的先验分布设为标准正态分布，即 $p\left(\upsilon\right) = \mathcal{N}\left(0, 1\right)$。

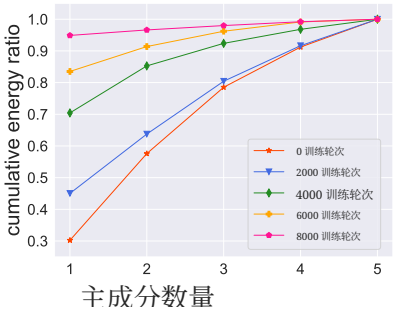
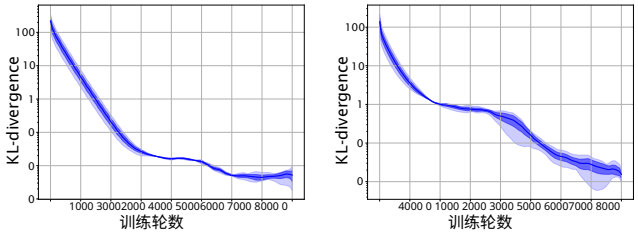


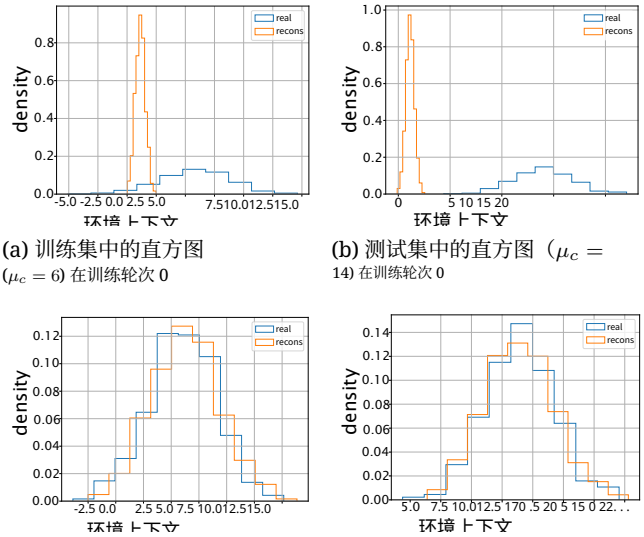
图 3. 关于υ的主成分数量的累积能量比示意图。能量由 υ的协方差矩阵的特征值表示。X 轴表示主成分的数量，Y 轴表示主成分的累积能量比。在附录 B 中展示了基于前两个主成分将 υ 投影为二维向量的可视化结果。



(a) 训练集中的 KLD　　(b) 测试集中的 KLD

图 4. LTS3 中训练集和测试集的 KL 散度示意图。实线表示三个随机种子的平均奖励。深色阴影表示标准误差，浅色阴影表示三个随机种子的最小—最大范围。

3）群组信息重构结果（***RQ1***）：我们使用 KLD 衡量重构性能。由于$p_\theta\left(\psi_s|\upsilon\right)$也会输出高斯分布的参数，因此我们直接利用高斯分布的解析表达式计算 $p_\theta\left(s|\upsilon\right)$ 与 $\mathcal{N}\left(\mu_c, \omega_c\right)$之间的 KLD。我们每 100 个训练轮次测试一次 KLD。图 4 显示，测试集中的 KLD 最终收敛至 0.01 到 0.02。图 5 显示，重构分布也存在相关性。



(a) 训练集中的直方图 ($\mu_c = 6$) 在训练轮次 0　　(b) 测试集中的直方图（$\mu_c = 14$) 在训练轮次 0

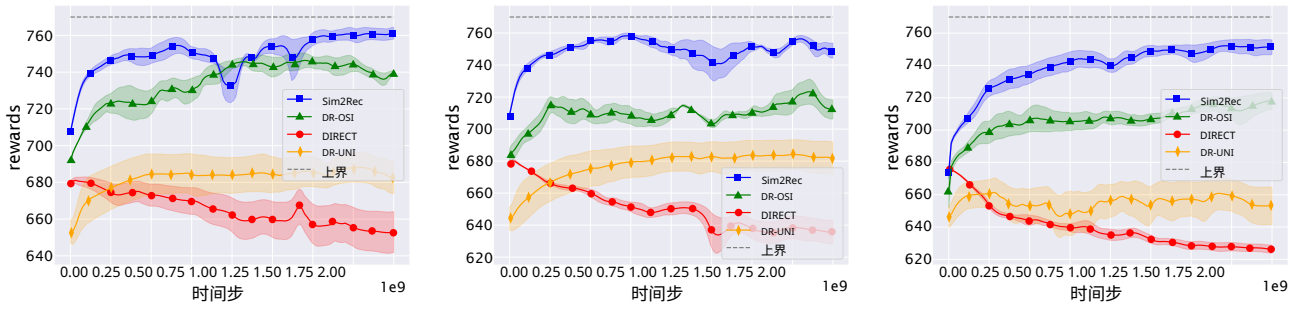
(c) 直方图　　在训练过程中　　set ($\mu_c = 6$) 在训练轮次 8000　　(d) 测试集中的直方图（$\mu_c = 14$) 在训练轮次 8000 时

图 5. 在 LTS3 任务中，o^i 的用户特征在重构数据和真实数据中的直方图示意图。

最后，我们通过主成分分析（PCA） [50]分析 SADAE 的嵌入性能。我们首先使用预先收集的数据集 $\mathcal{D}$ 训练q_κ ，并进行主成分分析。图 3 中 PCA 的累积能量比表明：经过 6000 个训练轮次后，潜在编码几乎可以由第一个主成分表示。将 υ 投影到由前两个主成分构成的二维向量空间，并与真实值 ω_g进行比较后，我们可以看到，ω_g 的值线性取决于第一个主成分（详情见附录 B）。

4) 策略性能结果（***RQ2***）：随后，我们测试了 Sim2Rec 在 SRS 中的适应性。结果如图 6 所示。首先，DIRECT 的结果表明，这些任务中的性能下降十分严重。由于未考虑训练与部署之间的差异，该策略会产生不可预测的行为。其次，所有考虑从多个动态模型中学习的算法都能提高其在未知环境中的鲁棒性。然而，采用环境表征的算法（Sim2Rec 和 DR-OSI）性能更好，因为它们试图在环境表征中寻找最优策略，而不是最大化训练集上的期望性能。此外，Sim2Rec 达到了近最优性能，并且在困难任务（例如 LTS3）上的表现优于 DR-OSI，这验证了第 IV-B 节提出的 SADAE 的必要性和有效性。在更困难的任务中，表征能力的局限性会限制上下文感知策略的性能。

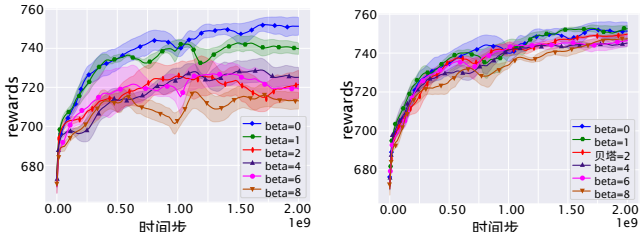
最后，我们分析模拟器集合的覆盖范围对 ω^*的影响。我们在 LTS3-β 中进行实验，该实验为模拟器中的每位用户注入参数差异。图 7



(a) LTS1 (b) LTS2 (c) LTS3

图 6. 合成环境中的性能示意图。实线曲线表示平均奖励，阴影表示三个随机种子的标准误差。“上界”表示直接在目标领域中训练的策略的性能。我们将其视为上界性能。

展示了 Sim2Rec 在此设置下的性能。从图 7(a) 可以看出，当差距水平增大时，训练集有限的 Sim2Rec 部署性能会下降，但仍优于对比方法。此外，从图 7(b) 可以发现，采样的模拟器数量充足时，Sim2Rec 能够很好地克服现实差距问题。总之，从经验上看，借助覆盖ω^*的模拟器集合，可以通过 Sim2Rec 克服现实差距问题。



(a) 500用户模拟器 (b) 无限用户模拟器

图 7. LTS3-β 任务中的性能示意图。实线表示平均奖励，阴影表示三个随机种子的标准误差。在 500 用户模拟器设置中，我们为每个模拟器和每个用户从 Uni$(-\beta, \beta)$ 中采样 ω_u。在无限用户设置中，我们在策略学习的每次迭代中为每个模拟器重新采样 ω_u。

C. 现实世界应用中的实验

1）滴滴出行的司机程序推荐（*DPR*）任务：滴滴出行的目标是平衡乘客的需求和司机的供给，即帮助司机完成更多订单，并满足更多乘客的出行需求。司机程序推荐（DPR）是网约车平台中 SRS 的典型任务。在 DPR 中，为了满足更多乘客的需求，我们希望通过推荐合理的程序物品序列，提升不同地区和城市中司机的长期参与度。参与度以每位司机完成的累计订单 数来衡量。平台每天向司机推送一次选定的程序。这些程序包含两个特征：（1）司机需要完成的任务，以连续变量建模。如果司机完成推荐的程序，其在平台上的参与度就会提高；（2）司机完成程序时平台产生的开支，也就是给司机的奖金。由于司机对不同地区的相同任务会有不同的反应，我们应根据司机的偏好及其所属的组，确定最佳的程序推荐。

DPR 可以建模为一个 MDP。为简化起见，我们假设可以忽略司机之间的影响。这是合理的，因为司机几乎不了解其他司机的任务。在 DPR 环境中，我们将每天视为一个时间步。在时间步 t，推荐系统策略 π根据观测到的特征 s_t^i，向司机 i 发送一个程序 $a_t^i = \pi(s_t^i)$。a_t^i 表示程序特征。

2) 实现：对于 SADAE，$q_\kappa\left(\upsilon|s^{(i)}, a^{(i)}\right)$ 输出 υ 的高斯分布参数。$p_\theta\left(\psi_a|\upsilon, s^{(i)}\right)$ 和 $p_\theta(\psi_s|\upsilon)$ 输出这些分布的参数。由于 DPR 中的动作是连续的，因此使用高斯分布对动作重构进行建模。然而，状态空间包含连续特征和离散特征。为简化起见，我们假设连续特征与离散特征相互独立。因此，我们分别使用多元高斯分布和分类分布对它们进行建模。υ 的先验分布设为标准正态分布，即 $p(\upsilon) = \mathcal{N}(0, 1)$。

我们通过 DEMER [1] 重构用户模拟器，该方法是网约车平台中最先进的用户模拟器学习技术之一。由于该模拟器通过数据驱动方法构建，我们采用第 IV-C 节中提出的技术进行可行参数空间构建。具体实现如下：(1) 我们基于 DEMER 使用不同的随机种子和不同城市的数据源训练 15 个模拟器，以构建 Ω'；(2) 对于每个时间步 t，奖励惩罚项$U(s_t, a_t) = \mathbb{E}[\|\mu_j(s_t, a_t) - \bar{\mu}(s_t, a_t)\|_2]$，其中 $\mu_j(s_t, a_t)$ 表示第i个模拟器在状态 s_t 和动作 a_t下所预测高斯分布的均值，$\bar{\mu}(s_t, a_t)$ 表示各模拟器预测结果的期望，$\|\cdot\|_2$ 表示 l2 范数；(3) 在 DPR 的所有实验中，T_c 均设为 5；(4) F_{trend}：我们进行干预测试，实验设置如图10所示，并移除所有模拟器中反应斜率为负或零的司机；(5) F_{exec}：我们计算每个组中每位司机过去 14 天的最小和最大动作值，将其作为可执行动作子空间，并直接采用 F_{exec}。

最后，对于每个时间步，奖励设定为：

$$\text{order} - \text{cost} \times \alpha_1 - 0.01 \times U(s, a),$$

其中，order 是司机已完成的订单，成本是司机的支出，可根据订单和

a, α_1 是一个权衡系数，设为平台的平均每单商品交易总额。

3) 离线测试设置：为进行离线测试，我们使用 Ω' 中的 12 个模拟器和数据集中的 80% 数据进行策略学习，并使用剩余的模拟器和数据进行测试。通过选择 Ω' 中的 3 个模拟器（名为 SimA、SimB 和 SimC）作为部署环境，我们构建了 3 个测试任务。这些任务使用相同的训练集和测试集。

4) 真实数据中的群体信息重构（***RQ1***）：我们在训练集上训练 SADAE，并在未见环境中测试重构后的数据分布。训练数据集 $\mathcal{D}$ 来自训练集中的人类专家数据。

我们每 100 个训练轮次测试一次 KLD。图 9(a) 显示，真实数据 X 与重构分布 $p_\theta(X|\upsilon)$ 之间的 KLD 稳步收敛至 0.6，这表明重构性能并非微不足道。图 8 展示了单个特征上真实数据和重构数据示例的直方图，它们之间也存在显著相关性。

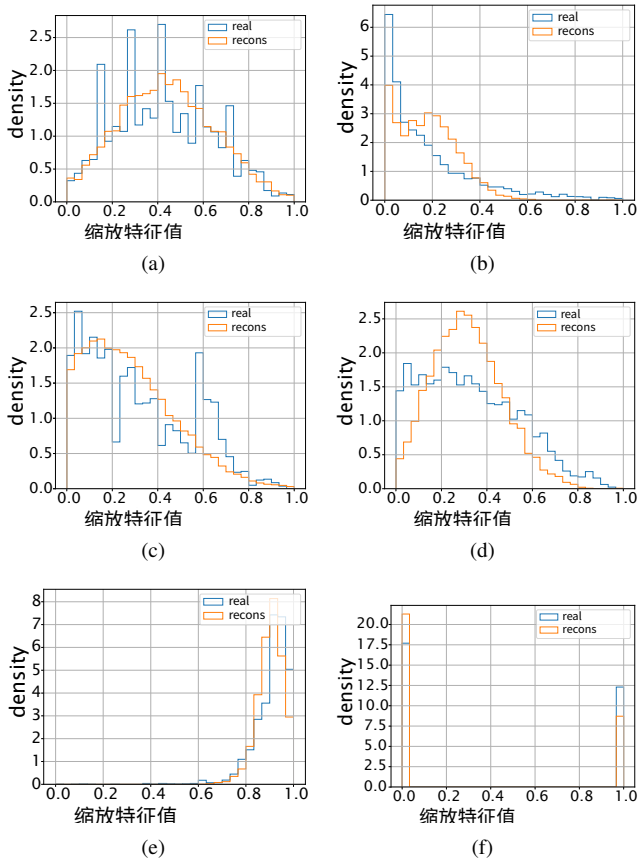
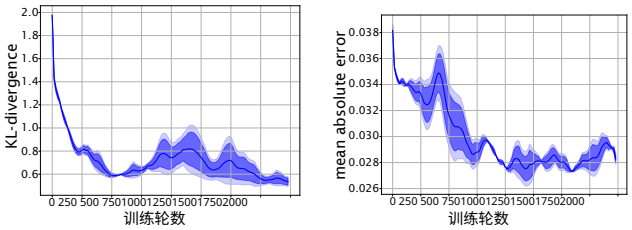


图 8. 部分状态的重构数据与真实数据直方图示意图。

为了评估 SADAE 的嵌入性能，我们进行了隐藏状态预测实验 [15]。我们使用另一个单层神经网络，根据两个数据对的嵌入变量（X_i，X_j，υ_i，υ_j）来预测它们的 KLD。该神经网络包含一个具有 32 个单元的隐藏层，以双曲正切函数作为激活函数，并连接到一个线性层，以预测通过公式 9 计算得到的 KLD。该神经网络每经过 SADAE 学习的 100 次迭代，就会对其进行初始化，并以相同的训练轮次重新训练。如果嵌入变量存储了有关分布的有用信息，那么任意两个数据集之间的 KLD 预测误差就会与训练轮次负相关。图 9(b) 展示了平均绝对误差（MAE）。与初始变量相比，MAE 提升了 26%，这表明嵌入变量有助于推断两个分布之间的关系。



(a) KL 散度　　(b) 平均绝对误差

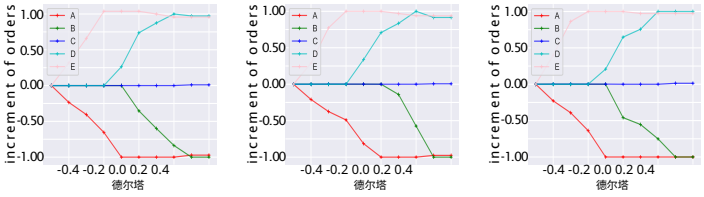
图 9. SADAE 上重构（左）和嵌入（右）性能的示意图。实线曲线表示均值。深色阴影表示标准误差，浅色阴影表示三个随机种子的最小—最大范围。

表 III 使用不同策略学习技术训练得到的策略的性能。该性能在 SiMA 中进行测试。

	订单（测试）	订单（训练）	成本（测试）	成本（训练）
Sim2Rec	2.0%	1.6%	0.9%	4.5%
Sim2Rec-PE	1.3%	2.3%	-8.0%	-4.0%
Sim2Rec-EE	8.1%	8.2%	-10.0%	-11.1%

5）可行参数空间构建的必要性（***RQ3***）：我们首先基于实际应用展示 SRS 的现实差距问题，并说明如果在训练过程中不采用第 IV-C 节提出的技术，现实差距会对策略训练产生何种影响。我们将 Sim2Rec 与 Sim2Rec-PE 和 Sim2Rec-EE 进行比较，并在表 III 中列出了训练集和测试集中相较于日志数据集中的行为策略 π_e 的订单和成本增幅百分比。如表 III 所示，在训练集中，Sim2Rec 实现的订单增幅低于 Sim2Rec-PE 和 Sim2Rec-EE。然而，在 Sim2Rec-PE 设置下，部署在该设置中训练得到的策略时，会出现较大的性能退化（43%），而 Sim2Rec 的性能在训练集和测试集之间则保持相近。这一现象表明，Sim2Rec-PE 的性能提升源于对模拟器预测误差的利用，而这种利用无法泛化到测试环境中，这也验证了第 IV-C 节提出的技术对于避免策略利用预测误差较大区域的必要性。另一方面，在 Sim2Rec-EE 设置下训练的策略，在训练集和测试集中均取得了优于 Sim2Rec 的性能，同时成本显著更低。然而，这种性能提升源于策略对模拟器外推误差的利用，而这种误差在这些集成模拟器中普遍存在。为了证明这一点，我们对模拟器进行了干预测试（图 10）。在干预测试中，我们选取每位司机的奖金 B（这是其中一个动作）作为研究对象。

在数据集中保留原始点，并以相同的偏差 ΔB 分配奖金¯：$B \leftarrow B + \Delta B$，然后我们根据原始状态¯特征和偏差奖金 B，记录司机的反馈预测 Y。对于每位司机，我们将Y 与不同的 ΔB 进行拼接，并通过 K-means 将响应向量分为 5 个不同的簇，如图 10 所示。在干预测试中，我们发现不同模拟器之间的反应模式相似，并且存在一些违反先验知识的模式（例如 A、B 和 C）。许多司机在不同模拟器中都表现出相同的模式。例如，根据我们的统计数据，在所有模拟器中，始终处于簇 C 的司机占 15%。这种反应违背了事实，并会误导策略训练，使其获得不合理的高性能。对于处于模式 A 的司机，策略可以降低奖金以提高其参与度，这解释了为什么 Sim2Rec-EE 能够获得更多的订单和更低的成本。这也进一步证明了第 IV-C 节提出的方法的必要性，即确保策略在没有较大外推误差的区域内进行优化，从而减少策略受到误导的可能性。



(a) SimA (b) SimB (c) SimC

图10. 干预测试中订单增量的示意图。每幅图展示了模拟器中司机响应向量的聚类中心。每条线表示一个聚类中心。X轴表示 ΔB的值。每个点的订单增量是从对应聚类的 $\Delta B = -0.5$ 值中减去该点的值。

表IV 使用不同算法学习的策略的性能。此处我们使用司机的期望累计奖励作为性能指标。

	SimA	SimB	SimC
Sim2Rec	**0.470**	**0.483**	**0.479**
DIRECT	0.450	0.241	0.027
DeepFM	0.325	0.302	0.368
WideDeep	0.192	0.398	0.211

6）离线测试中的策略性能（***RQ4***）：在上述讨论中，我们验证了所提出的策略学习技术的必要性。现在，我们展示基于模拟器的 Sim2Rec 性能。我们将 Sim2Rec 与两种基于监督学习方法的推荐系统 DeepFM [48]、WideDeep [47]，以及 DIRECT [1]进行比较。结果列于表 IV。我们发现，DeepFM 的迁移性能下降并不显著。DeepFM 和 WideDeep 也能在一定程度上从日志数据集中获得奖励。我们推测，强化学习风格的算法（例如 DIRECT）更容易对模拟器过拟合，
导致部署时行为不可靠 [51]。然而，在三项任务中，Sim2Rec 始终取得最优性能。

D. 生产环境中的 *A/B* 测试（***RQ5***）

我们最终将由 Sim2Rec 训练的策略部署到现实世界中，并测试其 7 天的性能。基线方法是一种基于模拟器的方法 DR-UNI，它使用与 Sim2Rec 相同的模拟器集合和强化学习算法 [46]，但不包含提取器和上下文感知策略。结果如图11所示。我们将司机分为对照组和实验组，并在某月第22天至第28天将该策略部署到实验组。在部署之前，司机接受相同的人工作策略推荐。我们发现，基线策略的性能提升为0.1%，与 AB 测试前的性能相近；而 Sim2Rec 的性能提升为6.9%，显著优于人工作策略和基线策略。

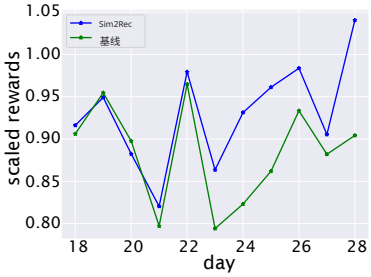


图11. 在线测试示意图。X 轴表示日期，Y 轴表示平均每日奖励。

VI. 讨论与未来工作

在本文中，我们研究了 SRS 中基于模拟器的强化学习用于 LTE 优化时所面临的现实差距问题。我们首先基于零样本策略迁移框架对该问题进行形式化，并提出了在 SRS 中解决现实差距问题所面临的额外挑战。我们构建了一种实用的从模拟到推荐（Sim2Rec）算法，以应对上述挑战，从而在现实世界中获得可靠的策略。实验在合成环境和现实世界应用中开展。我们使用合成环境量化所提出的环境参数提取器带来的性能提升。在现实世界应用中，我们验证了所提出技术的必要性，以及所提方法在生产环境中的有效性。

基于模拟器的强化学习是避免在现实世界序列推荐系统中通过试错学习策略所产生的成本的一种有前景的方法。我们希望 Sim2Rec 的良好性能能够激励研究人员通过处理现实差距来开发更强大的推荐系统。当前 Sim2Rec 的局限性主要源于第 IV-C 节中所提出技术的实现，这些技术仅基于经验方法进行设计。我们认为，还可以进一步探讨解决这些问题的更多理论方法，例如不确定性评估和外推误差评估；这将成为我们未来的工作。

致谢

本研究得到国家重点研发计划项目（2020年AAA0107200）、国家自然科学基金（61921006）和鹏城实验室重大重点项目（PCL2021年A12）的支持。

参考文献

[1] W. Shang、Y. Yu、Q. Li、Z. Qin、Y. Meng 和 J. Ye，《面向基于强化学习的推荐的含隐藏混杂因素的环境重构》，载于第*25*届*ACM SIGKDD*知识发现与数据挖掘国际会议论文集，2019年，第566–576页。[2] W. Shang、Q. Li、Z. Qin、Y. Yu、Y. Meng 和 J. Ye，《通过增益推断对基于强化学习的推荐进行部分可观测环境估计》，机器学习，第110卷，第9期，第2603–2640页，2021年。[3] J. Shi、Y. Yu、Q. Da、S. Chen 和 A. Zeng，《虚拟淘宝：为强化学习构建真实世界在线零售环境的虚拟化模型》，载于第*33*届*AAAI*人工智能大会（*AAAI 2019*）。夏威夷州檀香山：AAAI出版社，2019年，第4902–4909页。[4]Y. Gu、Z. Ding、S. Wang 和 D. Yin，《面向电子商务推荐系统的层次化用户画像》，载于第*13*届网络搜索与数据挖掘国际会议论文集，2020年，第223–231页。[5] G. Linden、B. Smith 和 J. York，《Amazon.com推荐：基于物品的协同过滤》，*IEEE Internet Computing*，第7卷，第1期，第76–80页，2003年。[6] G. Zhou、X. Zhu、C. Song、Y. Fan、H. Zhu、X. Ma、Y.Yan、J. Jin、H. Li 和 K. Gai，《用于点击率预测的深度兴趣网络》，载于第*24*届*ACM SIGKDD*知识发现与数据挖掘国际会议论文集，2018年，第1059–1068页。[7] Q. Zhang、J. Liu、Y. Dai、Y. Qi、Y. Yuan、K. Zheng、F. Huang 和 X. Tan，《通过强化学习实现多任务融合，以提升推荐系统中的长期用户满意度》载于第*28*届*ACM SIGKDD*知识发现与数据挖掘国际会议论文集，2022年，第4510–4520页。[8] P. Covington、J. Adams 和 E. Sargin，《用于YouTube推荐的深度神经网络》，载于第*10*届*ACM*推荐系统会议论文集，2016年，第191–198页。[9] L. Zou、L. Xia、Z. Ding、J. Song、W. Liu 和 D. Yin，《利用强化学习优化推荐系统中的长期用户参与度》，载于第*25*届*ACM SIGKDD*知识发现与数据挖掘国际会议论文集。阿拉斯加州安克雷奇：ACM，2019年，第2810–2818页。[10] X. Zhao、L. Xia、L. Zou、H. Liu、D. Yin 和 J. Tang，《UserSim：通过监督式生成对抗网络实现用户模拟》，载于*2021*年万维网大会论文集。纽约州纽约市：计算机协会，2021年，第3582–3589页。[11]S. Levine、A. Kumar、G. Tucker 和 J. Fu，《离线强化学习：教程、综述及开放问题展望》，*CoRR*，第abs/2005.01643卷，2020年。[12]X. Chen 和 Y. Yu，《具有无导数探索的强化学习》，载于第*18*届自主智能体与多智能体系统国际会议，加拿大蒙特利尔，2019年。[13]F. Zhang、J. Leitner、M. Milford、B. Upcroft 和 P. I. Corke，“面向机器人运动控制的基于视觉的深度强化学习”，*CoRR*，第 abs/1511.03791 卷，2015年。[14]X. B. Peng、M. Andrychowicz、W. Zaremba 和 P. Abbeel，“通过动力学随机化实现机器人控制的仿真到现实迁移”，载于第*35*届 *IEEE* 机器人与自动化国际会议论文集，2018年，第1–8页。[15] I. Akkaya、M. Andrychowicz、M. Chociej、M. Litwin、B. McGrew、A. Petron、A. Paino、M. Plappert、G. Powell、R. Ribas、J. Schneider、N. Tezak、J. Tworek、P. Welinder、L. Weng、Q. Yuan、W. Zaremba 和 L. Zhang，“用机器人手解魔方”，*CoRR*，第 abs/1910.07113 卷，2019 年。[16] J. Tobin 、 R. Fong 、A. Ray、J. Schneider、W. Zaremba 和 P. Abbeel，“用于将深度神经网络从仿真迁移到现实世界的领域随机化”，载于第*29*届 *IEEE/RSJ* 智能机器人与系统国际会议论文集，2017年，第23–30页。[17] P. Ramdya、J. Schneider 和 J. D. Levine，“黑腹果蝇群体行为的神经遗传学”，《实验生物学杂志》，第220卷，第35–41页，2017年。[18] X. Chen、S. Li、H. Li、S. Jiang、Y. Qi 和 L. Song，“用于基于强化学习的推荐系统的生成对抗用户模型”，载于第*36*届国际机器学习会议论文集，2019年，第1052–1061页。[19] K. Cho、B. van Merrienboer、D. Bahdanau 和 Y. Bengio，“神经机器翻译的特性：编码器—解码器方法”，载于*SSST@EMNLP 2014*：第八届统计机器翻译句法、语义与结构研讨会论文集。卡塔尔多哈：计算语言学协会，2014年，第103–111页。[20]F.-M. Luo、T. Xu、H. Lai、X.-H. Chen、W. Zhang 和 Y. Yu，“基于模型的强化学习综述”，*CoRR*，第 abs/2206.09328 卷，2022年。[21] F. Liu、R. Tang、X. Li、Y. Ye、H. Chen、H. Guo 和 Y. Zhang，“基于深度强化学习的推荐：显式用户—物品交互建模”，*CoRR*，第 abs/1810.12027 卷，2018年。[22]E. Ie、C. Hsu、M. Mladenov、V. Jain、S. Narvekar、J. Wang、R. Wu 和 C. Boutilier，“RecSim：可配置的推荐系统仿真平台”，*CoRR*，第 abs/1909.04847 卷，2019年。[23] X. Chen、Y. Yu、Z. Zhu、Z. Yu、Z. Chen、C. Wang、Y. Wu、H. Wu、R. Qin、R. Ding 和 F. Huang，“对抗性反事实环境模型学习”，*CoRR*，第 abs/2206.04890 卷，2022年。[24]L. Zou、L. Xia、P. Du、Z. Zhang、T. Bai、W. Liu、J. Nie 和 D. Yin，“Pseudo dyna-q：交互式推荐的强化学习框架”，载于

第*13*届 *ACM* 网络搜索与数据挖掘国际会议论文集，2020年，第816–824页。[25] Z. Zhu、X. Chen、H. Tian、K. Zhang 和 Y. Yu，“基于因果结构世界模型的离线强化学习”，*CoRR*，第 abs/2206.01474 卷，2022年。[26] J. Huang、H. Oosterhuis、M. de Rijke 和 H. van Hoof，“消除仿真中的数据集偏差：用于基于强化学习的推荐系统的去偏模拟器”，载于第*14*届 *ACM* 推荐系统会议论文集，2020年，第190–199页。[27] J. Wu、Z. Xie、T. Yu、Q. Li 和 S. Li，“通过离动力学强化学习实现仿真到现实的交互式推荐”，2021年。[28] F. Sadeghi 和 S. Levine，“CAD2RL：无需任何真实图像即可实现基于真实单张图像的飞行”，载于第*13*届机器人：科学与系统会议论文集，麻省理工学院，2017年。[29]J. Tobin、L. Biewald、R. Duan、M. Andrychowicz、A. Handa、V. Kumar、B. McGrew、A. Ray、J. Schneider、P. Welinder、W. Zaremba 和 P. Abbeel，“用于机器人抓取的领域随机化与生成模型”，载于*2018*年 *IEEE/RSJ* 智能机器人与系统国际会议，2018年，第3482–3489页。[30] K. Lee、Y. Seo、S. Lee、H. Lee 和 J. Shin，“用于基于模型的强化学习泛化的上下文感知动力学模型”，*CoRR*，第 abs/2005.06800 卷，2020年。[31] K. Rakelly、A. Zhou、C. Finn、S. Levine 和 D. Quillen，“通过概率上下文变量实现高效的离策略元强化学习”，载于第*36*届国际机器学习会议论文集，加利福尼亚州长滩，2019年，第5331–5340页。[32] F. Luo、S. Jiang、Y. Yu、Z. Zhang 和 Y. Zhang，“通过学习上下文敏感策略适应环境突变”，载于第*36*届 *AAAI* 人工智能会议论文集，线上会议，2022年，第7637–7646页。[33] W. Zhou、L. Pinto 和 A. Gupta，“环境探测交互策略”，载于第七届国际学习表征会议，路易斯安那州新奥尔良，2019年，会议论文集。[34]A. Nagabandi、I. Clavera、S. Liu、R. S. Fearing、P. Abbeel、S. Levine 和 C. Finn，“通过元强化学习，在动态的现实世界环境中学习适应”，载于第七届国际学习表征会议论文集，2019年。[35]S. Hochreiter 和 J. Schmidhuber，“长短期记忆”，神经计算，第9卷，第8期，第1735–1780页，1997年。[36] R. S. Sutton 和 A. G. Barto，“强化学习：导论”，*IEEE* 神经网络汇刊，第9卷，第5期，第1054–1054页，1998年。[37] T. Yu、G. Thomas、L. Yu、S. Ermon、J. Zou、S. Levine、C. Finn 和 T. Ma，“MOPO：基于模型的离线策略优化”，*CoRR*，第 abs/2005.13239 卷，2020年。[38] T. Xu、Z. Li 和 Y. Yu，“模仿的误差界

策略与环境，”载于神经信息处理系统进展 *33*，线上会议，2020年。[39] W. Yu、J. Tan、C. K. Liu 和 G. Turk，“为未知情况做好准备：通过在线系统辨识学习通用策略”，载于机器人：科学与系统 *XIII*，2017年。[40] F. Muratore、C. Eilers、M. Gienger 和 J. Peters，“通过贝叶斯优化实现数据高效的领域随机化”，*IEEE Robotics Autom. Lett.*，第 6 卷，第 2 期，第 911–918 页，2021年。[41] T. Miki、J. Lee、J. Hwangbo、L. Wellhausen、V. Koltun 和 M. Hutter，“在野外为四足机器人学习稳健的感知运动”，*Sci. Robotics*，第 7 卷，第 62 期，2022年。[42] X.-H. Chen、Y. Yu、Q. Li、F.-M. Luo、Z. T. Qin、S. Wenjie 和 J. Ye，“离线基于模型的可适应策略学习”，载于神经信息处理系统进展 *34*，2021年。[43]A. Vaswani、N. Shazeer、N. Parmar、J. Uszkoreit、L. Jones、A. N. Gomez、L. Kaiser 和 I. Polosukhin，“Attention Is All You Need”，载于神经信息处理系统进展 *30*，2017年，第 5998–6008 页。[44] D. P. Kingma 和 M. Welling，“自编码变分贝叶斯”，载于第二届国际学习表征会议论文集，2014年。[45] E. Todorov、T. Erez 和 Y. Tassa，“MuJoCo：用于基于模型控制的物理引擎”，载于第 *24* 届 *IEEE/RSJ*智能机器人与系统国际会议论文集，2012年，第 5026–5033 页。[46] J. Schulman、F. Wolski、P. Dhariwal、A. Radford 和 O. Klimov，“近端策略优化算法”，*CoRR*，第 abs/1707.06347 卷，2017年。[47] H. Cheng、L. Koc、J. Harmsen、T. Shaked、T. Chandra、H. Aradhye、G. Anderson、G. Corrado、W. Chai、M. Ispir、R. Anil、Z. Haque、L. Hong、V. Jain、X. Liu 和 H. Shah，“用于推荐系统的宽深度学习”，载于第一届推荐系统深度学习研讨会论文集，2016年，第 7–10 页。[48] H. Guo、R. Tang、Y. Ye、Z. Li 和 X. He，“DeepFM：一种基于因子分解机的点击率预测神经网络”，载于第 *26* 届国际人工智能联合会议论文集，2017年。[49] M. Rosenblatt，“关于密度函数若干非参数估计的评论”，《数理统计年刊》，第 27 卷，第 3 期，第 832–837 页，1956年 9 月。[50] S. Wold、K. Esbensen 和 P. Geladi，“主成分分析”，《化学计量学与智能实验室系统》，第 2 卷，第 1–3 期，第 37–52 页，1987年。[51] C. Zhang、O. Vinyals、R. Munos 和 S. Bengio，“深度强化学习中的过拟合研究”，*CoRR*，第 abs/1804.06893 卷，2018年。[52] K. Rakelly、A. Zhou、C. Finn、S. Levine 和 D. Quillen，“通过概率上下文变量实现高效的离策略元强化学习”，载于第 *36* 届国际机器学习会议论文集，2019年，第 5331–5340 页。

A. 证明

我们给出引理 4.1 和定理 4.1 的证明：

a) 引理 *4.1* 的证明：式（5）的目标函数可重写为：

$$
\begin{aligned}
& KLD(q_\kappa(\upsilon \mid X) \| p_\theta(\upsilon \mid X)) \qquad (10)\\
=& \mathbb{E}_{q_\kappa(\upsilon|X)}\left[\log \frac{q_\kappa(\upsilon \mid X)}{p_\theta(\upsilon \mid X)}\right] \\
=& \mathbb{E}_{q_\kappa(\upsilon|X)}\left[\log q_\kappa(\upsilon \mid X) - \log p_\theta(\upsilon, X) + \log p_\theta(X)\right] \\
=& -L(\theta, \kappa; X) + \log p_\theta(X). \qquad (11)
\end{aligned}
$$

由于 $\log\ p_\theta(X)$ 与 $q_\kappa(\upsilon \mid X)$ 无关，最小化式（10）等价于最大化式（11）中的 $L(\theta,\ \kappa;\ X)$。根据贝叶斯定理，我们有：

$$
\begin{aligned}
L(\theta, \kappa; X) =& \mathbb{E}_{q_\kappa(\upsilon|X)}\left[-\log q_\kappa(\upsilon \mid X) + \log p_\theta(\upsilon, X)\right] \\
=& \mathbb{E}_{q_\kappa(\upsilon|X)}\left[-\log q_\kappa(\upsilon \mid X) + \log\left(p_\theta(X \mid \upsilon) p_\theta(\upsilon)\right)\right] \\
=& \mathbb{E}_{q_\kappa(\upsilon|X)}\left[\log \frac{p_\theta(\upsilon)}{q_\kappa(\upsilon \mid X)} + \log p_\theta(X \mid \upsilon)\right] \\
=& \mathbb{E}_{q_\kappa(\upsilon|X)}\left[\log p_\theta(X \mid \upsilon)\right] - KLD\left(q_\kappa(\upsilon \mid X) \| p_\theta(\upsilon)\right).
\end{aligned}
$$

假设 X 是从 $\mathcal{D}$ 中独立同分布地采样得到的，我们得到证据下界（ELBO）目标函数：

$$
\max_{\kappa,\theta} \mathbb{E}_{X\sim\mathcal{D}}\left[\mathbb{E}_{q_\kappa(\upsilon|X)}\left[\log p_\theta(X \mid \upsilon)\right] - KLD\left(q_\kappa(\upsilon \mid X) \| p(\upsilon)\right)\right].
$$

b) 定理 *4.1* 的证明：在强化学习场景中，动作是在给定状态的条件下采样的，因此后验分布 p_θ 可分解为：

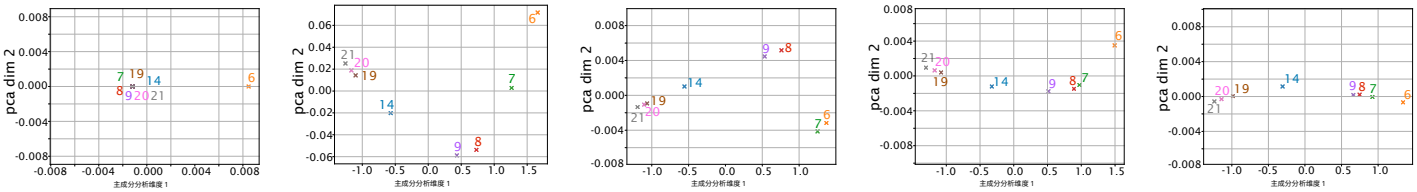
$$
\begin{aligned}
& p_\theta\left(s^{(i)}, a^{(i)} \mid \upsilon\right) \\
=& p_\theta\left(a^{(i)} \mid \upsilon, s^{(i)}\right) p_\theta\left(s^{(i)} \mid \upsilon\right) \\
=& p_{\psi_a}\left(a^{(i)}\right) p_\theta\left(\psi_a \mid \upsilon, s^{(i)}\right) p_{\psi_s}\left(s^{(i)}\right) p_\theta\left(\psi_s \mid \upsilon\right), \qquad (12)
\end{aligned}
$$

其中 ψ_s 和 ψ_a 表示该分布的解码参数。根据引理 4.1，式 5 的可计算目标函数可写为：

$$
\begin{aligned}
& \mathbb{E}_{X\sim\mathcal{D}, q_\kappa(\upsilon|X)}\left[\log p_\theta(X \mid \upsilon)\right] - KLD\left(q_\kappa(\upsilon \mid X) \| p(\upsilon)\right) \\
=& \mathbb{E}_{X\sim\mathcal{D}, q_\kappa(\upsilon|X)}\left[\sum_i^N \log p_\theta\left(x^{(i)} \mid \upsilon\right)\right] - KLD\left(q_\kappa(\upsilon \mid X) \| p(\upsilon)\right) \\
=& \mathbb{E}_{X\sim\mathcal{D}, q_\kappa(\upsilon|X)}\left[\sum_{i=1}^N \log p_\theta\left(s^{(i)} \mid \upsilon\right) + \log p_\theta\left(a^{(i)} \mid \upsilon, s^{(i)}\right)\right] \\
& - KLD\left(q_\kappa(\upsilon \mid X) \| p(\upsilon)\right).
\end{aligned}
$$

$q_\kappa\left(\upsilon \mid s^{(i)},\ a^{(i)}\right)$ 可以用高斯分布建模，因此结果 $q_\kappa\left(\upsilon \mid X\right)$ 也是高斯分布，并具有闭式解 [52]。对于任意可微的 p_{ψ_s} 和 p_{ψ_a}，通过重参数化技巧 [44]，证据下界目标函数是可计算的。

B. 主成分分析可视化



(a) 训练轮次 0　(b) 训练轮次 2000　(c) 训练轮次 4000　(d) 训练轮次 6000　(e) 训练轮次 8000

图12。 v 上可视化示意图。X 轴表示第一主成分，Y 轴表示第二主成分。每个交叉点表示状态分布的潜在编码的投影。与点颜色相同的数字表示真实值环境参数 ω_g。由于 $q_\kappa(v \mid X)$ 是高斯分布，为了便于阅读，我们只绘制该分布的均值。