arXiv:2303.04744v1 [cs.IR] 5 Feb 2023

面向设备端下一应用预测的联邦隐私保护协同过滤

Albert Sayapin*
斯科尔科沃科技学院，莫斯科，俄罗斯

Gleb Balitskiy
斯科尔科沃科技学院，莫斯科，俄罗斯

Daniel Bershatsky
斯科尔科沃科技学院，莫斯科，俄罗斯

Aleksandr Katrutsa
斯科尔科沃科技学院，莫斯科，俄罗斯

Evgeny Frolov
斯科尔科沃科技学院，莫斯科，俄罗斯

Alexey Frolov
斯科尔科沃科技学院，莫斯科，俄罗斯

Ivan Oseledets
斯科尔科沃科技学院，莫斯科，俄罗斯

Vitaliy Kharin一位
独立专家

摘要

在本研究中，我们提出了一种新颖的 SeqMF 模型，用于解决移动设备使用过程中预测下一应用启动的问题。尽管这一问题可以表示为经典的协同过滤问题，但由于数据具有序列性、用户反馈分布在各个设备上，而且向服务器汇总共同模式时必须防止用户数据泄露，因此需要对其进行适当改造。为满足这些要求，我们修改了经典矩阵分解模型的结构，并将训练过程更新为序列学习。由于有关用户体验的数据分布在各个设备上，我们采用联邦学习设置来训练所提出的序列矩阵分解模型。所提出方法的另一项要素是一种新的隐私机制，可确保用户发送至远程服务器的数据受到保护。为展示所提模型的有效性，我们使用了公开可用的移动用户行为数据，并将模型与序列规则以及基于应用启动频率的模型进行比较。比较在静态和动态环境中进行。静态环境用于评估我们的模型处理序列数据的能力，并与竞争方法进行比较。因此，我们采用标准的训练-验证-测试评估流程。动态环境则模拟真实场景：用户在设备上运行应用并生成新数据，同时评估模型在此情形下的表现。实验表明，在静态环境中，所提模型的性能与其他方法相当。然而，更重要的是，在动态环境中，我们的方法实现了比竞争方法更优的隐私-效用权衡，从而更准确地模拟了真实场景中的使用情况。

1. 引言

如今，智能手机会收集大量个人数据。移动操作系统可以根据收集到的数据，预测用户接下来会运行哪个应用。这项内部服务有助于更好地管理设备上的可用资源。例如，移动操作系统可以将部分应用从内存中移除，并在后台预加载其他应用，以节省计算资源或延长电池续航时间。这些收益对于改善用户使用设备时的体验至关重要。

本研究探讨下一应用预测问题，并提出一种解决该问题的新方法。该方法基于两个要素。第一部分是一种新的 SeqMF 模型，将现有的联邦矩阵分解方法 [3, 27] 扩展到序列学习场景。第二个要素则是一种新颖的隐私机制，用于对从设备发送到远程服务器的数据进行匿名化。虽然可以在每台设备上单独解决下一应用预测问题，从而避免将用户数据传输到远程服务器，但推荐系统领域的先前研究

*通讯作者（analystalb@gmail.com ）

[37]表明，用户的行为通常呈现共同模式。因此，利用集体行为数据可能有助于提升预测模型的质量。为此，需要与远程服务器交换信息，以实现协同过滤（CF）方案。

作为我们工作的基础，我们采用了一种针对矩阵分解模型改进的特殊联邦学习设置 [1] 。事实证明，这一思路适用于解决一般的协同过滤问题 [4] ，近期还加入了额外的隐私保护措施 [27]。我们工作的关键区别在于，重点关注设备端的下一应用预测问题，这与一般的协同过滤问题有显著差异。因此，模型训练和评估需要使用序列数据。此外，我们重新审视了隐私保护联邦学习问题，并提出了一种改进的 QHarmony 机制，用于对传输的数据进行匿名化。
因此，我们工作的主要贡献有两点：

- 我们针对下一应用预测问题提出了一种计算效率高的协同过滤模型 SeqMF。该模型基于序列感知矩阵分解技术，并在联邦学习设置下进行训练（第 4.1 节）。
- 我们为 SeqMF 模型配备了一种基于改进版 Harmony 机制的新型本地差分隐私（LDP）算法（第 5.2 节）。我们证明，与竞争方法相比，该算法具有更好的隐私-效用权衡（见第 6.5 节）。

论文组织。 本文的结构安排如下：我们首先在第2节介绍推荐系统如何用于下一应用预测，以及相关的隐私机制。随后，在第3节讨论一般的问题陈述和下一应用预测问题的特点。接着，在第4节详细介绍新的序列感知矩阵分解模型及其用于下一应用预测的联邦学习方法。我们在第5.2节介绍新颖的 ϵ-LDP QHarmony隐私机制。在第6节描述计算实验，并在第6.5节展示实验结果。最后，在第7节总结所得结果。

2. 相关工作

过去十年中，人们提出了许多用于设备端应用预测的算法。例如，Kamisaka等人 [20] 和Shin等人 [35] 采用了朴素贝叶斯模型，该模型使用从移动日志数据中提取的手工构造特征，而这些数据并未公开。Zhang等人 [43] 构建了一个贝叶斯网络，在推理过程中依赖时间、日期、位置以及此前使用过的应用。已有研究表明，若干基于最近邻的方法能够为该任务提供很大的灵活性 [24, 40, 6]。Baeza-Yates等人 [5]测试了多种预测方法，包括树增强朴素贝叶斯和基于C4.5算法的决策树。然而，这些方法大多被设计为仅在设备本地运行，不与远程服务器交换数据。因此，它们无法支持协作式设置。
下一应用预测问题的协同过滤形式在 [16, 32, 25]中得到了提出和研究。Ammad-Ud-Din 等人 [4]近期探索了适用于协同过滤任务的联邦学习范式。作者提出了一种特殊的混合优化方案，用于在联邦学习设置中训练并兼顾个人数据保护的非序列式模型。随后，Minto 等人 [27]通过采用更复杂的本地差分隐私机制，进一步提升了该方法的数据保护能力。
此外，研究人员还为序列推荐任务开发了许多基于深度学习的模型。Fang 等人 [15]对这类模型进行了全面综述。然而，这些研究既未探索联邦学习设置，也未探讨与隐私相关的问题。因此，深度学习模型在真实场景中的适用性仍是一个悬而未决的问题。

在协同过滤设置中，为了保护传输过程中的敏感用户数据，采用本地差分隐私机制，因为与集中式差分隐私（中心化差分隐私，CDP）机制 [13,11, 12]不同，该机制不假设存在可信第三方（例如服务器）。通常，本地差分隐私机制会先在设备上加密敏感用户数据，然后才将其传输到远程服务器。因此，加密阶段必须兼顾计算效率和内存效率。

本地差分隐私机制最著名的应用是 RAPPOR算法 [14]。它基于随机响应机制 [39], , 该机制后来被应用于许多本地差分隐私算法中。然而，由于它处理的是类别数据，因此并不适用。对于实值数据，人们采用了拉普拉斯机制 [11] 。该机制能够保证隐私，但当传输数据的维度较高时，会导致效用大幅下降。随后，Duchi 等人 [10] 提出了对随机响应（RR）技术的改进。在 [29], 中，作者指出，在某些情况下，Duchi机制无法保护隐私。为了解决这些问题，他们提出了一种简单的Harmony机制。此外， [7] 还建议在应用

RR 之前使用Kashin表示和量化。目前，渐近地说，这种方法在效用、通信成本和隐私预算之间实现了最佳权衡。然而，由于计算复杂度较高，它在设备上的应用受到限制。

3. 问题陈述

令 M 为用户数量， N 为可供安装的应用总数。根据用户历史记录，我们可以构造交互二值矩阵 $\mathbf{A} = [a_{ij}]$ ，其大小为 $M \times N$ ，使得当第 i 个用户运行过第 j 个应用时， $a_{ij} = 1$ 。形式上，假设用户 u 的设备上已安装应用的索引集合为 A_u ，满足 $|A_u| \ll N$ ；并假设已使用应用的有序集合为 $\mathcal{H}_u^{(t)} = \{i_1, i_2, \ldots, i_p\}$ ，满足 $i_k \in A_u$，我们将其称为时刻 t 的已使用应用历史。由于矩阵 $\mathbf{A}$ 是根据这些历史数据构造的，因此它也取决于参数 t ，为简化记法，我们省略该参数。

我们的主要任务是预测特定用户的设备上接下来将运行哪个应用。在这里，我们看到给定数据的序列特性，这对训练模型的高性能至关重要。接下来，我们需要获得一个函数，将可用历史记录或其中一部分映射到下一个应用的索引 $i^* \in A_u$。遵循先前的研究 [18, 4]，我们构造相关性评分函数 r，该函数取决于应用的未知嵌入 $\mathbf{q}_i$，$i = 1, \ldots\ldots, N$以及用户的未知嵌入 $\mathbf{p}_u$， $u = 1, \ldots\ldots, M$。 $r(\mathbf{q}_i, \mathbf{p}_u)$ 的值越高，应用 i 对用户 u 就越相关。因此，当前用户 u 的下一个应用预测索引，是所有候选应用$i \in A_u$中 $r(\mathbf{q}_i, \mathbf{p}_u)$ 的最大元素所对应的索引。

然而，我们需要从矩阵 $\mathbf{A}$ 中收集的观测历史中获取适当的嵌入向量 $\mathbf{p}_u$ 和 $\mathbf{q}_i$ 。寻找这些嵌入的经典方法是求解以下优化问题 [18]:

$$\frac{1}{2}\sum_{u,i} c_{ui}(a_{ui} - r(\mathbf{q}_i, \mathbf{p}_u))^2 + \frac{\lambda}{2}\sum_i \|\mathbf{q}_i\|_2^2 + \frac{\lambda}{2}\sum_u \|\mathbf{p}_u\|_2^2 \to \min_{\{\mathbf{q}_i, \mathbf{p}_u\}}, \tag{1}$$

其中， c_{ui} 衡量观测到 a_{ij} 和 $r(\mathbf{q}_i,\mathbf{p}_u) = \mathbf{q}_i^\top \mathbf{p}_u$的置信度。

计算最优 嵌入 $\{\mathbf{q}_i^*,\mathbf{p}_u^*\}$后，我们可以按如下方式为特定用户推荐 n 个应用作为候选项 :

$$\text{toprec}(u; n) = \arg\max_{i \in A_u}^{n} \left(r(\mathbf{q}_i^*, \mathbf{p}_u^*)\right), \tag{2}$$

其中，arg $^n\max_{i \in A_u}(f(i)) = (i_1^*, i_2^*, \ldots, i_n^*)$ 是从 A_u 中选出的应用有序集合，满足：如果 $f(i_k^*) \geq f(i_l^*)$，则 $k < l$。

求解问题 (1) 的标准方法是交替最小二乘（ALS）方法，该方法会更新 $\mathbf{q}_i$ 和 $\mathbf{p}_u$ [18]。然而，由于每个用户都在设备上本地存储自己的交互历史 $\mathcal{H}_u^{(t)}$，且所有用户拥有的应用均来自预定义的可用应用集合，因此利用联邦学习范式 [4, 27] 来求解 (1) 是很自然的。在该范式中，用户嵌入$\mathbf{p}_u$ 在设备上以并行异步方式进行本地更新，而应用嵌入 $\mathbf{q}_i$ 则在远程服务器上进行全局更新。为了执行此类更新，我们必须支持用户设备与服务器之间的数据双向传输。由于将数据从用户发送至第三方服务器可能导致个人数据泄露，因此必须在训练过程中引入适当的隐私算法。此外，相关性评分函数 r 必须经过构造，以便处理与应用交互历史相对应的序列数据。考虑到这些要求，我们将在下一节中给出对 (1) 的适当修改。

4. 序列感知矩阵分解模型的联邦学习

在本节中，我们设计了一个新的目标函数，将用户序列行为的相关知识融入目标函数 (1) 的框架中。随后，我们介绍通过联邦学习方法获得最优用户嵌入和应用嵌入的具体细节。该方法结合了用于在本地更新用户嵌入的 ALS 步骤，以及在远程服务器上更新应用嵌入的基于梯度的更新。

4.1 用于下一应用预测的序列感知矩阵分解

4.1.1 序列数据的目标函数

设 $\mathbf{P} \in \mathbb{R}^{M \times d}$ 和 $\mathbf{Q} \in \mathbb{R}^{N \times d}$ 分别是由用户嵌入和应用嵌入逐行组成的矩阵。由于用户历史记录分布在各设备上，我们将（1）改写为以用户为中心的形式:

$$\mathcal{L}(\mathbf{P}, \mathbf{Q}) = \frac{1}{2}\sum_u \|\mathbf{r}(\mathbf{Q}, \mathbf{p}_u) - \mathbf{a}_u\|_{\mathbf{C}_u}^2 + \frac{\lambda}{2}\left(\sum_u \|\mathbf{p}_u\|_2^2 + \|\mathbf{Q}\|_F^2\right) \to \min_{\{\mathbf{Q}, \mathbf{p}_u\}}, \tag{3}$$

其中，$\mathbf{a}_u$ 是一个长度为 N 的二元指示向量，在第 i个位置上的值为 a_{ui}；矩阵 $\mathbf{C}_u \in \mathbb{R}^{N\times N}$ 是对角矩阵，其对角线上第 i个元素分别等于 c_{ui}。此外，我们使用以下记号：$\|\mathbf{x}\|^2_{\mathbf{W}} = \mathbf{x}^\top\mathbf{W}\mathbf{x}$，并且 $\|\mathbf{Q}\|^2_F$ 表示矩阵 $\mathbf{Q}$的弗罗贝尼乌斯范数的平方。

标准矩阵分解模型 [3] 将物品相关性得分计算为 $\mathbf{r}_{MF}(\mathbf{Q}, \mathbf{p}_u) = \mathbf{Q}\mathbf{p}_u$，但它无法捕捉现有数据的序列特性。为捕捉用户的序列模式，我们引入以下相关性评分函数$\mathbf{r}$，其灵感来自 [25] 中的研究进展：

$$\mathbf{r}(\mathbf{Q}, \mathbf{p}_u) = \mathbf{Q}\mathbf{p}_u + \mathbf{h}_u(\mathbf{Q}) = \mathbf{Q}\mathbf{p}_u + \text{diag}\left(\mathbf{S}_u\mathbf{Q}\mathbf{Q}^\top\right), \tag{4}$$

其中，设计矩阵$\mathbf{S}_u$ 编码了相对频率权重，这些权重对应于用户 u 的历史记录中从某个先前应用 j 转移到应用 i 的情况。式（4）中的第二项仅取决于局部转移统计，因此能够捕捉序列行为。

$\mathbf{S}_u$ 的元素根据用户 u 交互历史 $\mathcal{H}_u^{(t)}$，按照以下公式计算：

$$(\mathbf{S}_u)_{ij} = \frac{\sum_{k=2}^{|\mathcal{H}_u^{(t)}|} \mathbb{1}[j \to i]\mathbb{1}[i_{k-1}^u = j]\mathbb{1}[i_k^u = i]}{\sum_{k=2}^{|\mathcal{H}_u^{(t)}|} \mathbb{1}[i_{k-1}^u = i]}, \mathbb{1}[m=n] = \begin{cases}1, & m=n\\ 0, & \text{otherwise},\end{cases} \quad \mathbb{1}[j\to i] = \begin{cases}1, & (j,i)\in\mathcal{H}_u^{(t)}\\ 0, & \text{otherwise},\end{cases} \tag{5}$$

其中，$(j,\ i) \in \mathcal{H}_u^{(t)}$表示 i 和 j 均属于历史记录 $\mathcal{H}_u^{(t)}$，且 i 紧接在 j之后。例如，假设用户 u 的交互历史如下：$\mathcal{H}_u^{(t)} = (a,\ b,\ c,\ a,\ a,\ b,\ a,\ c)$。那么，所得矩阵 $\mathbf{S}_u$ 的大小为 3×3，其形式如下：

$$\mathbf{S}_u = \begin{array}{c} \\ a \\ b \\ c \end{array}\!\!\begin{array}{c} \begin{array}{ccc} a & b & c \end{array} \\ \begin{pmatrix} \frac{1}{4} & \frac{2}{4} & \frac{1}{4} \\ \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{1} & 0 & 0 \end{pmatrix} \end{array} \tag{6}$$

例如，考虑该矩阵的第 (a,b) 个元素等于 $2/4$，因为历史记录中有 2 次 (a,b) 的序列共现，以及 4 次 a。

$\mathbf{h}_u^{(t)}(\mathbf{Q})$ 的第 i 个元素表示目标应用 i 与其他应用之间的平均相似度，并根据矩阵$\mathbf{S}_u$ 中相应元素进行加权；该矩阵编码了序列历史 [16]:

$$\left(\mathbf{h}_u^{(t)}\right)_i = \sum_{j=1}^{N}\left(\mathbf{S}_u^{(t)}\right)_{ij}\mathbf{q}_i^\top\mathbf{q}_j. \tag{7}$$

这里我们明确使用上标 (t)，以强调所考虑时段的持续时间十分重要。为了更好地捕捉应用使用频率中的全局动态，需要限制时段的持续时间。可以使用一个时间窗口来选取最近发生的事件，例如数周或数月，具体取决于可用数据。较小的 t 能提升计算性能，而较大的 t 则可能有助于发现彼此并非紧接着发生的事件之间的长程相关性。

因此，我们提出了适用于序列数据的新型相关性评分函数（4），并修改了矩阵分解模型中的经典损失函数。我们将由此得到的目标函数和相关性评分函数称为序列矩阵分解模型，或**SeqMF**。

4.1.2 SeqMF 模型中的推理

相关性评分函数 (4) 旨在帮助从用户操作历史中正确学习应用共现统计。然而，预测规则 (4) 的计算成本过高，在推理步骤中难以承受。因此，在推理步骤中，我们针对用户 u 和用户设备上安装的第 i 个应用，提出以下更轻量级的相关性评分函数：

$$\hat{r}(\mathbf{q}_i, \mathbf{p}_u; s_u) = \mathbf{q}_i^\top\mathbf{p}_u + \mathbf{q}_i^\top\mathbf{q}^{(s)}, \qquad \mathbf{q}^{(s)} = \sum_{i_k\in s_u}\mathbf{q}_{i_k}, \tag{8}$$

其中，$s_u = (i_1,\ \ldots\ldots,\ i_l)$ $i_k \in A_u$是用户最近交互过的 l 个应用的集合。第一项 $\mathbf{q}_i^\top\mathbf{p}_u$等于式（4）中对应第一项的第$i$个元素，第二项$\mathbf{q}_i^\top\mathbf{q}^{(s)}$ 则是式（4）中第二项的近似值，用于捕捉用户行为的局部模式。因此，我们在保留相关性评分函数主要特性的同时，降低了推理步骤的计算成本。

4.1.3 基于置信度的加权系数

基于置信度的加权方法旨在模型训练过程中，为观测交互和未观测交互的预测值与真实值之间的差异分配适当的重要性权重。这样便有望学到更好的嵌入，并更准确地捕捉潜在的行为模式。常见的加权方案 [18] 会为未观测数据分配较小的统一权重，为已观测条目分配较高的、依赖于数据的权重。这类方案已被证明在标准协同过滤任务 [18, 22], 中行之有效，因为在这类任务中，任何可用条目都可以推荐给任何用户。然而，初步实验表明，标准加权方案在我们的场景中并不奏效。可能的原因是，下一个应用只能从已安装应用中选择，而非从所有可能的应用中选择。

因此，遵循 基于频率的加权思路 [17], ，我们提出以下置信度 e
权重表达式:

$$c_{ui} = \frac{d_{ui}^{\gamma} + \alpha}{\sum_j d_{uj}^{\gamma} + \alpha N}, \tag{9}$$

其中， d_{ui} 是用户 u历史记录中项目 i 的启动频率，且 $\alpha \in [0;\ 1]$，$\gamma \in [0;\ 1]$ 是超参数。通过使用 α可加（拉普拉斯）平滑 [38] 对相对频率 d_{ui} 进行调整，我们使非活跃应用的置信度值成为一个较小的非零常数，这与传统的基于加权 ALS 的方法完全兼容。请注意，非零的 α 由于实现上的特殊性会导致计算成本增加。

4.2 SeqMF 模型的联邦学习

由于用户交互历史存储在用户设备本地，因此采用联邦学习方法来求解（3）是很自然的。针对协同过滤设置指定的联邦学习方法由两个主要的相互通信模块组成 [4]: 位于远程服务器上的应用嵌入 $\mathbf{Q}$，以及存储在用户设备本地的用户嵌入 $\mathbf{p}_u,\ \ u = 1, \ldots, M$（每台设备一个向量）。矩阵 $\mathbf{Q}$ 的最新版本会被分发到各台设备。随后，每台设备都可以利用新收集的行为数据更新相应的用户嵌入，并生成更准确的预测。按照一定的周期，用户设备中的部分数据会被发送回远程服务器，以更新矩阵$\mathbf{Q}$，然后重新开始这一过程。

该方案的关键在于更新用户嵌入和应用嵌入所采用的方法不同。如 [4], 所述，默认的 ALS 方案在更新矩阵 $\mathbf{Q}$ 时，需要将完整的用户交互历史发送到服务器，这会直接威胁数据隐私。为缓解这一问题，作者采用了一种混合优化方案，将用户嵌入的 ALS 更新与应用嵌入的基于梯度的更新相结合。

根据 ALS 方案，矩阵$\mathbf{P}$ 按行方式更新如下:

$$\mathbf{p}_u = \left(\mathbf{Q}^\top \mathbf{C}_u \mathbf{Q} + \lambda \mathbf{I}\right)^{-1} \mathbf{Q}^\top \mathbf{C}_u \left(\mathbf{a}_u - \mathbf{h}_u(\mathbf{Q})\right). \tag{10}$$

我们强调，每个用户嵌入$\mathbf{p}_u$ 都会在用户设备上私密地且并行地更新，以支持联邦学习的分布式特性。此外，请注意，在此类更新过程中，$\mathbf{h}_u(\mathbf{Q})$ 所表示的短期序列动态会抵消$\mathbf{a}_u$ 中编码的长期偏好的影响。这正是我们计划实现的效果。

相比之下，矩阵$\mathbf{Q}$ 会在远程服务器上使用基于梯度的优化器进行更新。例如，经典梯度下降的形式如下:

$$\mathbf{Q} := \mathbf{Q} - \beta \frac{\partial \mathcal{L}}{\partial \mathbf{Q}}, \tag{11}$$

其中， $\beta > 0$ 是学习率， $\frac{\partial \mathcal{L}}{\partial}\mathbf{Q}$ 是引入的损失函数 (3) 的完整梯度，计算如下:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{Q}} = \sum_{u=1}^{M} \mathbf{F}(u) + \lambda \mathbf{Q}, \qquad \mathbf{F}(u) = \mathbf{D}_u \mathbf{e}_N \mathbf{p}_u^\top + \left(\mathbf{D}_u \mathbf{S}_u + \mathbf{S}_u{}^\top \mathbf{D}_u\right) \mathbf{Q}, \tag{12}$$

其中，$\mathbf{D}_u = \mathrm{diag}\left(\mathbf{C}_u \left(\mathbf{r}(\mathbf{Q}, \mathbf{p}_u) - \mathbf{a}_u\right)\right)$，$\mathbf{e}_N$是一个大小为 N 的全 1 向量，$\mathrm{diag}(w)$ 是一个以向量 w 为对角线元素的对角矩阵。

要在服务器端计算 $\frac{\partial \mathcal{L}}{\partial}\mathbf{Q}$，我们只需要式 (12) 中的值 $\mathbf{F}(u)$。这些值可以从远程服务器上的客户端收集。然而，实际上，并非所有客户端都会根据请求发送 $\mathbf{F}(u)$。我们通过使用随机梯度估计器来处理这一问题:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{Q}} \approx \sum_{u \in U_b} \mathbf{F}(u) + \lambda \mathbf{Q}, \tag{13}$$

其中， U_b 是能够传输 $\mathbf{F}(u)$ 的用户集合。之后，可以使用带动量的随机梯度下降 [31] 或 Adam [21] 这类随机优化器，在远程服务器上更新矩阵 $\mathbf{Q}$。

尽管所述联邦学习混合方案需要将$\mathbf{F}(u)$从用户设备传输到远程服务器，但这些数据不包含明确的用户交互历史，因此敏感性较低。尽管如此，这些数据仍容易受到某些类型的攻击，因此我们会对这项传输进行隐私保护。我们在¯远程服务器上聚合匿名化数据，并使用扰动后的梯度估计$\mathbf{F}$ 来更新应用嵌入$\mathbf{Q}$。为更好地保护传输数据的隐私，我们提出了一种新的隐私保护机制，将在下一节中介绍。

5. SeqMF 模型中的隐私机制

5.1 本地差分隐私

将隐私机制纳入基于矩阵分解的模型中既简单又自然。图 1 展示了一个客户端与远程服务器之间的隐私通信方案。在该方案中，用户嵌入$\mathbf{p}_u$ 和用户行为历史 $\mathcal{H}_u^{(t)}$仅存储在设备上，以保护敏感信息的隐私。然而，为了更新矩阵$\mathbf{Q}$ 中的应用嵌入，用户会使用私有数据计算梯度$\mathbf{F}(u)$。将这些梯度传输到远程服务器可能导致私有数据泄露。因此，为了在整个学习过程中维护隐私，在传输梯度之前需要使用额外的隐私保护机制。

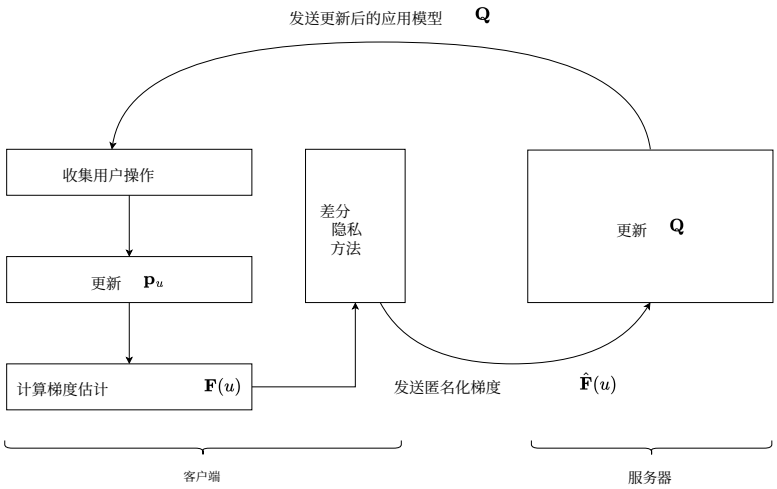

图 1：服务器与客户端之间的通信方案及所传输的数据。

通常，保护敏感数据有几种方法：安全多方计算（MPC） [41], 同态加密（HE） [33], 可信执行环境（TEE） [8, 23, 9], 差分隐私（DP） [13, 11, 12]。然而，TEE 在硬件层面施加了诸多限制。MPC 和 HE 会显著增加计算复杂度，或需要可信第三方 [19]。因此，接下来我们仅考虑本地差分隐私（ϵ-LDP）技术，该技术无需大幅增加计算复杂度和通信成本，即可提供强隐私保证 [19]。此外，与中心化差分隐私算法（CDP） [11] 不同， ϵ-LDP 机制不需要可信第三方。

定义 1. 一个随机 化机制 $\mathcal{A}$ 是 ϵ-本地差分隐私（ϵ-*LDP*），当且仅当对于任意两个输入 s v、 v' 以及任意输出 $y \in range(\mathcal{A})$ 满足以下条件：

$$\mathbb{P}(\mathcal{A}(v) = y) \leq e^{\epsilon}\mathbb{P}(\mathcal{A}(v') = y), \tag{14}$$

其中， ϵ称为隐私预算。隐私预算越小，对敏感数据的保护越好。

隐私保护矩阵分解方法已在相关研究中得到探讨。具体而言，为了在传输前扰动梯度，梯度聚合隐私保护算法 [27, 36] 中采用了 k-Harmony机制。下文将讨论该机制的缺点，并针对我们的场景提出其改进版本，称为*QHarmony*机制。此外，我们将在下一节讨论拉普拉斯机制 [11], Kashin机制 [7],及其缺点。

5.2 QHarmony机制

考虑用户 u 的梯度矩阵$\mathbf{F}(u) = [f_{ij}(u)] \in \mathbb{R}^{N\times d}$，其中 N 为应用数量， d 为嵌入维度。此外，我们假设 $f_{ij}(u) \in [-1,1],\ i \in 1,\ N,\ j \in 1,\ d,\ u \in 1,\ M$。由于我们可以对计算梯度进行归一化，因此该假设很容易成立。扰动梯度矩阵$\mathbf{F}(u)$由设备上的本地差分隐私机制 $\mathcal{A}$ 根据$\mathbf{F}(u)$计算得到。随后，用户将$\mathbf{F}(u)$发送至远程服务器，并获得隐私保证。在服务器端，接收到的扰动梯度$\mathbf{F}(u)$ $u \in U_b$ 会进行聚合；如有需要，还会对聚合后的更新执行额外变换。根据用于 $\bar{}$ $\bar{}$均值估计的标准聚合方案，$\mathbf{F} = [f_{ij}]$ 的计算方式如下：

$$\bar{\mathbf{F}} = \frac{1}{|U_b|}\sum_{u\in U_b}\hat{\mathbf{F}}(u). \tag{15}$$

这种梯度聚合方式旨在减少接收到的梯度 $\hat{}\mathbf{F}(u)$中的噪声，因此需要足够多的用户 M。然而，在我们的设置中，用户数量与梯度矩阵中的元素数量大致相同，因此，基线 ϵ-本地差分隐私机制并不适用于这种设置。具体而言，

- k-Harmony机制会使扰动梯度变得稀疏，因为每个用户只传输 k 个元素。 k 的选择基于低隐私预算 ϵ 和每个传输元素的高隐私预算 $\hat{}\frac{\epsilon}{k}$ 的要求。后者预算越高，扰动梯度元素 f_{ij} 中的噪声就越少。因此， k 应当较小，例如 $k < 10$。为了计算 $\bar{f}_{ij}$，只对大约 $\frac{kM}{Nd} \approx k$ 个值求和$\bar{}$；由于 k 较小，远程服务器无法足够准确地估计 f_{ij}。

- 根据拉普拉斯机制， [11], 梯度扰动的方差等于 $\frac{4N^2d^2}{\epsilon^2}$，对于较低的隐私预算而言，这是一个很大的数值，例如 $\epsilon < 3$。同时，聚合梯度元素_ $\bar{f}_{ij}$ 的标准差为 $\frac{2\sqrt{2}Nd}{\epsilon M} \approx \frac{2\sqrt{2}}{\epsilon}$。当隐私预算较低时，该标准差$\bar{}$ $\bar{}$与元素 f_{ij}相当。因此，使用拉普拉斯机制得到的估计值 $\mathbf{F}$ 同样不准确。

- 尽管与其他机制相比，Kashin机制对聚合梯度$\bar{}$ $\mathbf{F}$ 具有最低的渐近方差上界，但其计算成本过高。此外，如果$\bar{}$渐近估计中的隐藏常数较大，Kashin机制给出的估计值$\mathbf{F}$就不准确。

因此，在所考虑的设置中，基线隐私保护机制对平均梯度（15）的估计方差较大，导致性能不佳。为了提升整体性能并保持适中的计算$\bar{}$开销，我们对 k-Harmony机制作如下修改。由于该机制可视为对$\mathbf{F}$的量化，我们采用量化方法中的聚合方法。因此，我们的隐私机制包含以下步骤。首先，用户均匀采样 k 个不同的索引对（i_l, j_l），其中$i_l \in 1, N$、 $j_l \in 1, d$和 $l \in 1, k$。然后，用户将$\hat{}\mathbf{F}(u)$中除$\hat{}$ $\overline{f_{i_l j_l}(u)}$ $l \in \overline{1,k}$之外的所有$\bar{}$元素置为零，这些元素按以下方式计算：

$$\mathbb{P}[\hat{f}_{i_l j_l}(u) = x] = \begin{cases} \frac{f_{i_l j_l}(u)(e^{\epsilon/k}-1)+e^{\epsilon/k}+1}{2(e^{\epsilon/k}+1)}, \text{ if } x = 1 \\ -\frac{f_{i_l j_l}(u)(e^{\epsilon/k}-1)+e^{\epsilon/k}+1}{2(e^{\epsilon/k}+1)}, \text{ if } x = -1. \end{cases} \tag{16}$$

$\hat{}$ 最后，用户将 k 个三元组（$f_{i_l j_l}(u)$ i_l, j_l）和 $f_{\max}(u) = \max_{i,j} f_{ij}(u)$发送至远程服务器。此外，$\bar{}$我们观察到，服务器上的聚合梯度$\mathbf{F}$ 是非隐私梯度估计（13）的量化表示。基于这一观察，我们用以下表达式替换基线聚合公式（15）：

$$\bar{\mathbf{F}} = \max_u f_{\max}(u)\sum_{u\in U_b}\frac{\hat{\mathbf{F}}(u)}{\max\limits_{ij}\sum\limits_{u\in U_b}\left[\hat{f}_{ij}(u) > 0\right]}, \tag{17}$$

其中，每位用户都会将 $f_{\max}(u) = \max_{ij} f_{ij}(u)$ 和 $f_{\max}(u)$ 传输到服务器。我们将这一新的隐私保护算法称为*QHarmony*，并分别在算法 1 和算法 2 中总结了用户端和服务器端的操作。

算法1QHarmony机制：客户端

要求： $\mathbf{F}(u)$ 是用户 u 的计算梯度设备，ϵ 是隐私预算，k 是传输的非零值数量。

确保： $\mathcal{I}(u)$ 是 k 个三元组的集合（$\hat{f}_{i_l j_l}(u)$ i_l，j_l）对应于对应扰动后每个非零位置的位置索引梯度 $\mathbf{F}(u)$ 在第 u 个用户设备上，$f_{\max}(u)$ 是 $\mathbf{F}(u)$ 的最大元素。

函数 QHARMONYC($\mathbf{F}(u)$, ϵ, k)
  计算 $f_{\max}(u) = \max_{ij} f_{ij}(u)$
  初始化：$\mathcal{I}(u) = \emptyset$
  初始化：$\mathcal{P} = \{i\}_{i=1}^{N} \times \{j\}_{j=1}^{d}$
  对 $l = 1, \ldots, k$ **执行**
    从 $\mathcal{P}$ 中均匀随机采样（i_l, j_l）
    采样一个随机数字 $b \sim Bernoulli\left(\frac{f_{i_l j_l}(u)(e^{\epsilon/k}-1)}{2 \cdot e^{\epsilon/k}+1} + \frac{1}{2}\right)$
    $\hat{f}_{j_l i_l}(u) = 2b - 1$
    $\mathcal{P} = \mathcal{P}(u) / \{(i_l, j_l)\}$
    $\hat{\mathcal{I}} = \mathcal{I}(u) \cup \{(f_{i_l j_l}(u)\ i_l,\ j_l)\}$
  结束循环
返回 $\mathcal{I}(u)$, $f_{\max}(u)$
结束函数

算法 2QHarmony机制：服务器端

要求： 对于每个用户 $u \in U_b$, $\mathcal{I}(u)$ 是一组 k 个三元组（$f_{i_l j_l}(u)$ i_l，j_l），对应于每个非零 $\hat{}$ 来自扰动梯度 $\hat{}\mathbf{F}(u)$ 中的零元素第 u个用户，$f_{\max}(u)$是 $\mathbf{F}(u)$的最大元素。

确保： $\bar{\mathbf{F}}$ 是重新生成的扰动梯度用于矩阵$\mathbf{Q}$ 更新的远程服务器。

函数 QHARMONYS($\mathcal{I}(u)$,$f_{\max}(u)$, $u \in U_b$)
  初始化：$\mathbf{S} = \{0\}^{N \times d}$,$\mathbf{Z} = \{0\}^{N \times d}$
  对于 $u \in U_b$，**执行**
    对于 $\hat{f}_{i_l j_l}(u)$ i_l, j_l) $\in \mathcal{I}(u)$ **执行**
      $s_{i_l j_l} = s_{i_l j_l} + \hat{f}_{i_l j_l}(u)$ ▷ 聚合梯度
      $\hat{z}_{i_l j_l} = z_{i_l j_l} + [f_{i_l j_l}(u) > 0]$
    结束循环
  结束循环
  $z_{\max} = \max_{ij} z_{ij}$
  $\bar{\mathbf{F}} = \frac{\max_u f_{\max}(u)}{z_{\max}} \mathbf{S}$
返回 $\bar{\mathbf{F}}$
结束函数

6. 实验

6.1 数据

我们在两个公开可用的移动应用使用数据集上进行了实验。LSApp 数据集 [2] 收集了跨应用移动搜索查询，作者采集了用户按时间顺序产生的应用使用事件。该数据集包含来自 $M = 292$ 位用户的 $N = 87$ 个不同应用。App Usage 数据集 [42], 的每条记录都包含匿名化的用户身份标识、HTTP 请求或响应的时间戳等信息。该数据集包含来自 $M = 1000$ 位用户的 $N = 2000$ 个不同应用。我们注意到，这两个数据集中都存在大量冗余：同一应用的启动事件可能在不到 3 秒内重复出现多次。我们认为这不符合真实用户行为，因此将此类情况合并为单个事件。

此外，我们根据连续启动应用之间的时间间隔，将用户操作划分为不同会话。如果当前会话中，距离上一次观测到的事件已超过预先设定的阈值（15 分钟），则开始一个新会话。LSApp 和 App Usage 中用户的平均会话数分别为 109 和 58。会话划分仅用于评估，不会影响训练过程。在训练期间，我们只考虑数据的序列特性，并将所有用户历史记录表示为一个长会话。

6.2 评估方法

6.2.1 Top-n 预测

主要目标是预测用户设备上可用的下一个应用，而不是整个应用商店中的应用。因此，我们将可能的预测范围限制为用户设备上已安装的应用。例如，如果用户历史记录中包含与 5 个不同应用的交互，我们会要求模型仅为这 5 个应用生成预测分数。然后，对于 $n = 3$ 的 top-n 推荐，我们从这 5 个应用中选出预测分数最高的 3 个应用。请注意，用户 u 的历史记录中的应用数量 $|A_u|$ 可能小于 n。因此，我们计算 top-min($|A_u|$, n) 个预测，并仍将结果计入 top-n 指标。

6.2.2 评估协议

在每个会话中，我们使用 迭代揭示方案 [26] 来评估预测质量。该方案意味着，对于特定用户 u 和会话 s，我们会依次遍历 s 中的应用，并结合之前的所有应用来预测下一个应用。因此，对于长度为 l 的会话，我们依次生成 $l-1$ 个预测。我们使用相关性指标和排名相关指标，例如命中率（HR@n） [44], 平均倒数排名（MRR@n） [30], 和归一化折损累计增益（NDCG@n） [30], ，其中 n 对应推荐项目的数量。由于我们的设定与标准设定略有不同，我们相应地调整了这些指标，同时保留其属性。具体而言，平均计算不仅按用户数量进行，也按会话数量进行。

6.3 用于比较的方法

矩阵分解。 这是 [4] 提出的原始非序列基于矩阵分解的方法。

序列规则。 我们使用一种序列规则（SR）预测器 [26] ，该预测器基于用户历史记录中连续出现的应用之间的共现频率。这是对基于马尔可夫链假设的标准项目间方法的直接修改。在收集此类成对序列共现的统计数据后，我们直接使用这些数据，不进行归一化，为下一个项目分配预测分数。需要注意的是，根据数据收集方式，SR 模型有两种可能的变体。模型既可以通过协作方式学习，也可以通过隔离方式（在设备上）学习。我们将前者记为 *SR* ，后者记为*SR-od*。

最近使用。 MRU 预测器根据当前会话中项目的新近程度分配分数。为遵循评估协议（见第 6.2 节），不属于请求项目的项目将获得零分。较旧项目的非零分数较低，较新项目的分数较高，从而在生成的预测中将最近的会话项目排在前面。

最常使用。 最常使用预测器会根据每位用户的事件频率，计算项目流行度分数。这些分数用于选取最常使用的前若干个项目进行预测。

随机。 最后，该预测器会为所有请求的项目分配随机分数。

值得一提的是，尽管存在更复杂的模型，例如 [45, 34] 本研究的主要目的是开发一种用于设备端下一应用预测的隐私保护方法，使其在真实场景中优于知名模型。因此，基于神经网络隐私保护联邦学习的方法不在本研究范围之内，将作为未来工作的研究主题。

6.4 实验方法

6.4.1 静态环境

静态环境 [16] 旨在评估 SeqMF 模型处理序列数据的能力，并将其与标准基线方法进行比较。具体而言，在此环境中，采用标准方法划分训练集、测试子集和验证子集，以训练和评估应用预测模型。验证子集用于调整超参数。具体来说，我们调整用户和应用嵌入的维度 d、优化器 $\beta > 0$ 在（11）中的学习率、正则化系数 $\lambda > 0$以及（9）中的平滑参数 α 和 γ。在 LSApp 数据集中，训练集为前 222 天，验证集为接下来的 7 天，测试子集为最后 14 天。对于 App Usage 数据，我们将前 6 天划分为训练子集，接下来 1 天划分为验证子集，最后 1 天划分为测试子集。

6.4.2 动态环境

动态环境对应于模型在真实场景中的行为：模型内置于移动操作系统中，并开始在新设备上运行。为了取得高性能，模型必须快速适应用户行为，并开始生成合理的预测。动态环境假设我们在设备上存储了预训练应用嵌入$\mathbf{Q}$ 以及随机初始化的用户嵌入$\mathbf{p}_u$。随后，嵌入会根据在线数据流进行更新。例如，考虑一个持续 5 天生成的数据流。在动态环境中，我们使用第一天的数据评估 SeqMF 模型的初始化情况。接着，使用这一天的数据更新嵌入，并使用第二天的数据评估模型，然后再用第二天的数据更新嵌入。此过程持续到第五天。最终，我们得到五个数值，用来根据所选指标展示模型质量的动态变化。

动态环境设定中的另一个重要因素是用户活动随时间变化的分布。如果这种分布不均衡，那么由于在少量数据上估计出的梯度噪声较大，学习过程会不稳定

。我们在 LSApp 数据集中观察到了这种情况，见图 2（左）。为了使所研究的数据集达到均衡，我们人为地重新排列用户在周期中的操作，使每个周期内的活跃用户数量几乎相同，见图 2（右）。根据用户活动情况，每个周期可能包含一天或数天。同时，在重新排列过程中保留了用户操作的顺序。为了获得预训练应用嵌入$\mathbf{Q}$，我们取第一个周期来训练 SeqMF 模型。后续周期构成动态环境中的数据流。

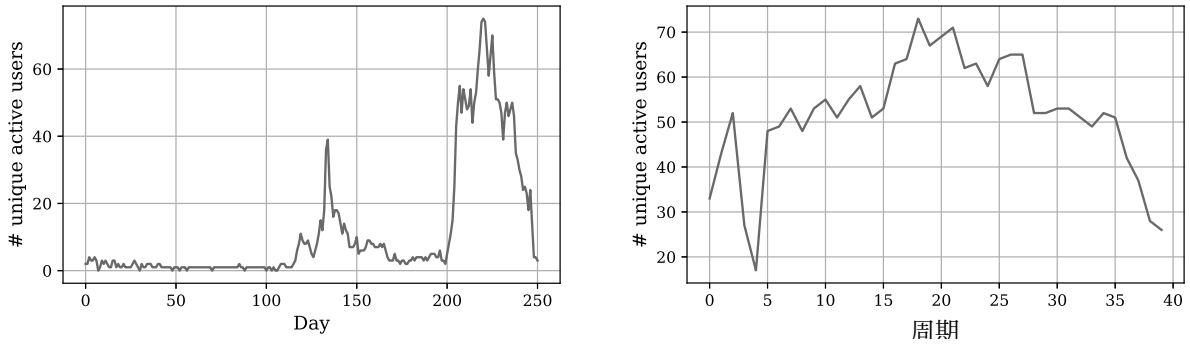

图 2：LSApp 数据集中原始统计数据（左）与重新排列为周期后的统计数据（右）的比较。

6.5 结果

6.5.1 静态环境

表 1 展示了所提出的 SeqMF 模型与上述竞争方法在静态环境下对 LSApp 数据集的比较结果，表 2 则展示了它们在 App Usage 数据上的比较结果。可以看出，SeqMF 在两个数据集上的表现都显著优于其非序列前身 MF，这符合数据的序列特性。然而，在所有指标上，它的表现都不如 LSApp 数据集上的简单 SR-od 模型，也不如 App Usage 数据上的 SR 模型。众所周知，SR 模型通常是表现强劲的基线方法，在各种数据集上都有良好表现 [26]。需要指出的是，造成这一结果至少有两个原因。首先，数据集中存在大量重复事件，包括由同一应用构成的长序列。因此，即使我们尝试过滤掉这些异常，它们仍会导致模型强烈偏向重复事件。另一个原因是，一旦收集到足够的统计数据，序列模式就会趋于平稳，也更容易被发现。因此，我们在第 6.5.2 节中展示了动态环境下的比较结果，该环境与真实场景相对应。

表 1：LSApp 数据集上的静态评估结果。

模型	HR@1	HR@3	HR@5	平均倒数排名@1	平均倒数排名@3	平均倒数排名@5	归一化折损累计增益@1	归一化折损累计增益@3	归一化折损累计增益@5
随机	0.089	0.239	0.365	0.089	0.153	0.181	0.089	0.175	0.227
MRU	**0.618**	0.823	0.849	**0.618**	**0.716**	**0.722**	**0.618**	**0.744**	0.754
MFU	0.403	0.669	0.777	0.403	0.516	0.541	0.403	0.555	0.600
SR	0.610	0.801	0.863	0.610	0.695	0.709	0.610	0.723	0.748
SR-od	0.612	**0.824**	**0.893**	0.612	0.707	**0.722**	0.612	0.737	**0.765**
MF	0.379	0.649	0.784	0.379	0.497	0.527	0.379	0.536	0.591
SeqMF	0.522	0.797	0.871	0.522	0.643	0.660	0.522	0.683	0.713

6.5.2 动态环境：定期更新用户嵌入至关重要

本节中，我们研究预测规则 (4) 中的长期组件和短期组件对模型性能的影响。我们采用第 6.4.2 节中的设置进行实验。

我们考虑 SeqMF 模型训练的三种机制：完整、稀有和全局。在这些机制中，对于 LSApp 数据集，应用嵌入$\mathbf{Q}$每 10 个周期更新一次；对于 App Usage 数据集，则每两个周期更新一次。在完整机制中，用户嵌入$\mathbf{p}_u$每个周期都会更新。在 Rare 机制中，用户嵌入$\mathbf{p}_u$仅在应用嵌入

表 2：App Usage 数据集上的静态评估结果。

模型	HR@1	HR@3	HR@5	平均倒数排名@1	平均倒数排名@3	平均倒数排名@5	归一化折损累计增益@1	归一化折损累计增益@3	归一化折损累计增益@5
随机	0.029	0.093	0.152	0.029	0.056	0.069	0.029	0.065	0.090
MRU	0.393	0.660	0.707	0.393	0.516	0.527	0.393	0.553	0.572
MFU	0.229	0.436	0.546	0.229	0.319	0.344	0.229	0.349	0.394
SR	**0.477**	**0.664**	**0.746**	**0.477**	**0.559**	**0.578**	**0.477**	**0.586**	**0.620**
SR-od	0.414	0.591	0.659	0.414	0.493	0.508	0.414	0.518	0.546
MF	0.227	0.426	0.520	0.227	0.314	0.336	0.227	0.343	0.382
SeqMF	0.402	0.615	0.686	0.402	0.497	0.513	0.402	0.527	0.557

更新时才会更新$\mathbf{Q}$。在全局机制中，用户嵌入$\mathbf{p}_u$从不更新，并在随机初始化后保持不变。SR-od 模型会在每个周期中利用此前周期收集的数据重新训练。

为了评估模型在动态环境中的性能，我们引入了一个额外指标，记为 δHR@5。该指标等于目标训练机制计算出的 HR@5 与在完整机制下训练的 SeqMF 模型计算出的 HR@5 之差，因为我们假设后者最具代表性。在下方的图中，我们给出了数据流中的累积 δHR@5，以说明用户嵌入更新计划对 SeqMF 模型性能的影响。我们还绘制了 SR-od 模型的性能，因为它在静态环境中表现出较高性能（见表 1 和表 2），且仅使用设备上的本地数据。后一特点使 SR-od 模型适用于我们所考虑的设置，该设置接近真实场景。

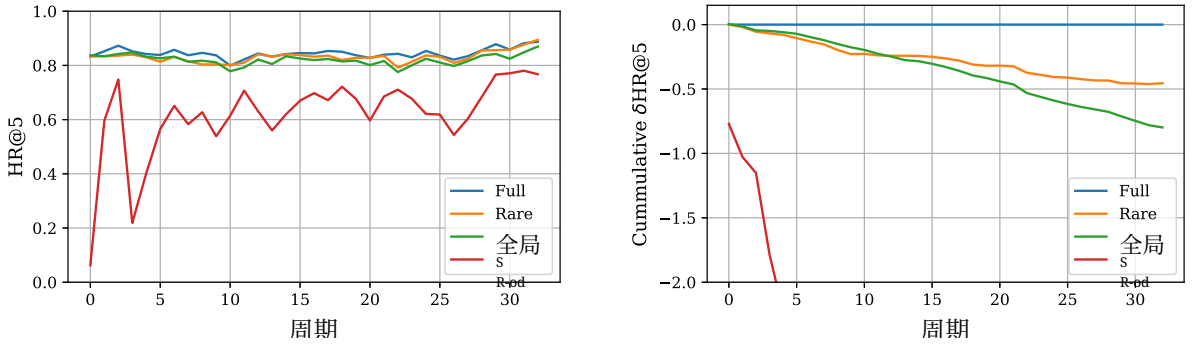

图3：*LSApp* 数据集。动态环境下，SeqMF 模型和 SR(od) 模型在所考虑的训练机制中的 HR@5 指标（左）和累积 δHR@5（右）。在动态环境中，SeqMF 模型在所考虑的训练机制下均优于 SR-od 模型。右图表明，定期更新用户嵌入可以提升性能。

图3展示了在动态环境中，SeqMF 模型在 *LSApp* 数据上优于 *SR-od* 模型。这一观察证实了我们对 SR-based 模型在静态环境中表现出色的原因所作的假设。正如我们所见，如果完全不在设备上更新用户嵌入（全局机制），SeqMF 模型的整体性能就会下降。然而，如果只是偶尔更新用户嵌入（Rare 机制），其性能相比全局机制下的 SeqMF 模型有所提升。因此，我们通过实验表明，更新用户嵌入$\mathbf{p}_u$对于保持 SeqMF 模型的高性能至关重要。

图4展示了 SeqMF 模型在动态环境中针对 App Usage 数据集的性能。从左图可以看出，SeqMF 模型在完整和 Rare 机制下的性能非常相似。这一趋势可能是因为第一个周期已包含足够多的用户行为模式信息。因此，对于这个特定数据集，用户嵌入的更新频率并不重要。

6.5.3 动态环境：QHarmony机制优于竞争方法

为分析隐私机制对 SeqMF 模型性能的影响，我们采用与上一节类似的动态环境和完整训练方案。我们比较不使用

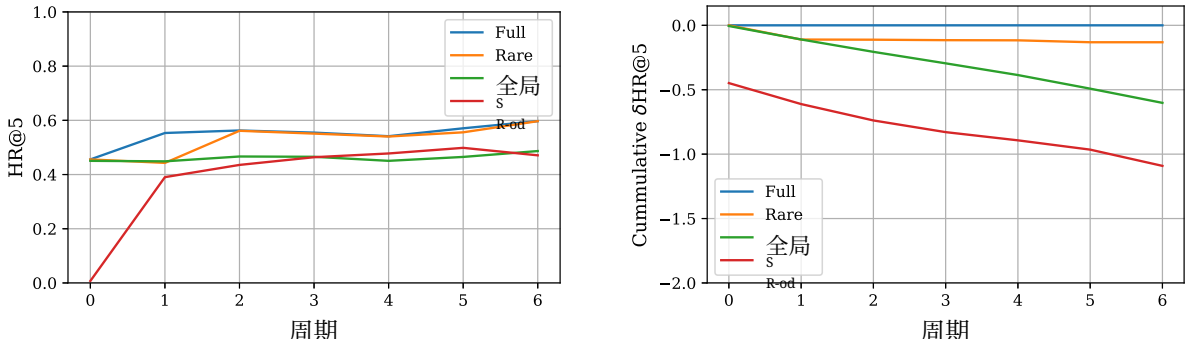

图4：*App Usage* 数据集。动态环境下所考察的 SeqMF 模型和 SR(od) 模型在各训练机制中的 HR@5 指标（左）和累积 δHR@5（右）。只有在更新用户嵌入时（完整和稀有机制），SeqMF 模型在所考察的训练机制中才会在动态环境下胜过 SR-od 模型。右图表明，定期更新用户嵌入可以提升性能。

隐私机制的 SeqMF 模型与结合所考虑基线方法及所提出的 QHarmony机制的同一模型的性能。

图 5 显示，与不采用任何隐私机制的 SeqMF 模型相比，采用所提出的 QHarmony 算法且隐私预算为 $\epsilon = 4.5$ 的 SeqMF 模型在 LSApp 数据集上的效用甚至更高。这种现象可能是由于深度学习研究中早已观察到的随机梯度具有众所周知的隐式正则化特性 [28]。同时，正如第 5.2 节所预期的那样，除 Kashin机制外，基线隐私机制的表现都较差。Kashin机制的隐私-效用权衡仍然较差，而且计算成本远高于所提出的 QHarmony机制。

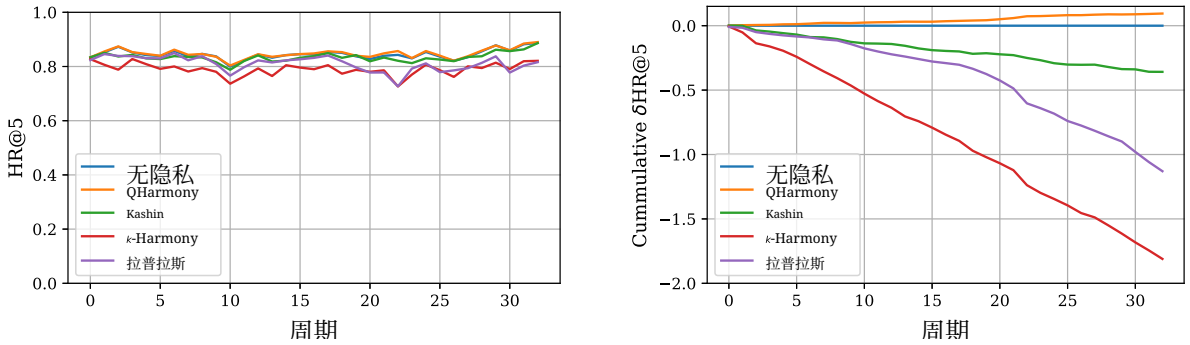

图 5：*LSApp* 数据集。**左图：**在 SeqMF 模型中纳入隐私预算$\epsilon = 4.5$ 后，HR@5 随隐私保护机制变化的情况。HR@5 越高，模型性能越好。因此，QHarmony机制能为敏感用户数据提供更强的保护。**右图：**展示 33 个周期内的累积 δHR@5，结果证实，与非隐私保护 SeqMF 模型相比，QHarmony机制不会导致性能下降。

图 6 显示，在 App Usage 数据上，与非隐私保护 SeqMF 模型相比，所提出的 QHarmony机制在预算 $\epsilon = 1.1$ 下几乎保持了相同水平的 HR@5。与 LSApp 数据的情况类似，与所提出的 QHarmony机制相比，基线隐私机制的隐私-效用权衡较差。这些实验验证了所提出的 QHarmony机制的有效性，并表明对于所考察的两个数据集，在引入 QHarmony机制后，SeqMF 模型的性能得以保持。

7. 结论

在本研究中，我们探讨了下一应用预测问题，并提出了一种新的序列感知矩阵分解模型，称为 SeqMF。该模型能够预测每位用户接下来会使用的应用，从而优化用户设备上的资源消耗。SeqMF 模型考虑了数据的序列特性，配备了新的 QHarmony 机制以提供更强的隐私保证，并采用联邦学习范式进行训练。

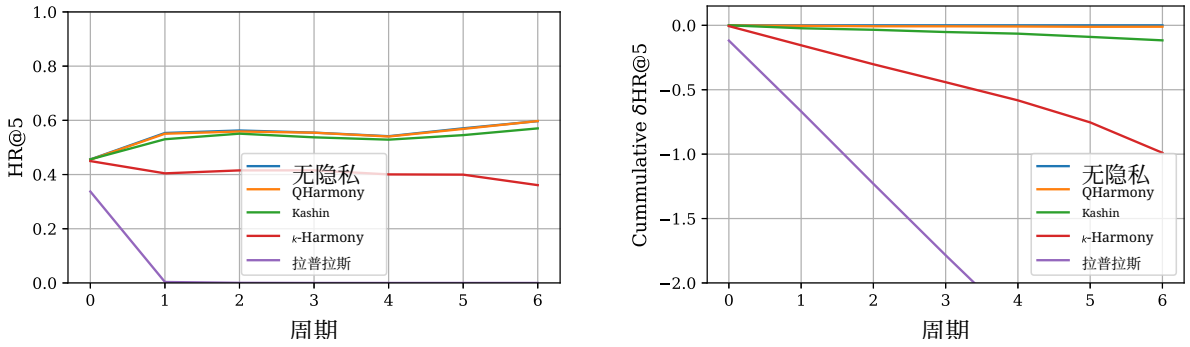

图 6：*App Usage* 数据集。**左：**将隐私预算 $\epsilon = 1.1$ 纳入 SeqMF 模型后，HR@5 随隐私保护机制的变化情况。HR@5 越高，模型性能越好。QHarmony机制为敏感用户数据提供了更强的保护。**右：**展示 7 个周期的累积 δHR@5，结果证实，与非隐私保护 SeqMF 模型相比，QHarmony机制不会导致性能下降。

这一范式意味着，用户嵌入存储在用户设备本地，而所有设备上的应用嵌入均相同，并在远程服务器上更新。这些特点使 SeqMF 模型能够抵御用户设备向远程服务器传输的数据可能遭到的泄露，并有助于捕捉多个用户共有的行为模式。为了展示模型的性能，我们在 LSApp 和 App Usage 数据集上进行了实验。在这些实验中，我们模拟了真实场景中的数据流，以便恰当地评估所研究的模型。实验结果表明，根据收集到的统计数据，SeqMF 模型优于标准矩阵分解模型及其他竞争方法。此外，与竞争方法相比，所提出的 QHarmony 隐私机制实现了最佳的隐私-效用权衡，且计算复杂度适中。因此，在实际应用中，所提出的 SeqMF 模型与 QHarmony 隐私机制相结合，相比此前针对下一应用预测任务提出的解决方案具有实际优势。

参考文献

[1] S. Abdulrahman、H. Tout、H. Ould-Slimane、A. Mourad、C. Talhi 和 M. Guizani。联邦学习综述：从集中式到分布式现场学习及更广阔领域的发展历程。*IEEE* 物联网期刊，8(7):5476–5497，2021年。doi: 10.1109/JIOT.2020.3030072。[2] M. Aliannejadi、H. Zamani、F. Crestani 和 W. B. Croft。用于增强个性化移动助手的上下文感知目标应用选择与推荐。载于*ACM* 信息系统汇刊 *(TOIS)*，2021年。[3] M. Ammad-ud-din、E. Ivannikova、S. A. Khan、W. Oyomno、Q. Fu、K. E. Tan 和 A. Flanagan。用于保护隐私的个性化推荐系统的联邦协同过滤。*CoRR*，abs/1901.09888，2019年。URL http://arxiv.org/abs/1901.09888。[4] M. Ammad-Ud-Din、E. Ivannikova、S. A. Khan、W. Oyo mno、Q. Fu、K. E. Tan 和 A. Flanagan。用于保护隐私的个性化推荐系统的联邦协同过滤。*arXiv* 预印本 *arXiv:1901.09888*，2019 年。[5] R. Baeza-Yates 、 D. Jiang 、 F. Silvestri 和 B. Harrison。预测你接下来将使用的应用。*WSDM 2015* - 第八届 *ACM* 国际网络搜索与数据挖掘会议论文集，第285–294页，2015年02月。doi: 10.1145/2684822.2685302。[6] B. M. Changmai、D. Nagaraju、D. P. Mohanty、K. Singh、K. Bansal 和 S. Moharana。用于上下文与序列感知推荐的设备端用户意图预测。*arXiv* 预印本 *arXiv:1909.12756*，2019年。.[7] W.-N. Chen、P. Kairouz 和 A. Ozg¨ur。打破通信—隐私—准确率三难困境。*ArXiv*，abs/2007.11707，2020年。[8] V. Costan 和 S. Devadas。英特尔 SGX 解析。*IACR* 密码学电子印刷档案，2016:86，2016年。[9] V. Costan、I. Lebedev 和 S. Devadas。Sanctum：用于强软件隔离的最小硬件扩展。载于*USENIX* 安全研讨会，2016年。[10] J. C. Duchi、M. Wainwright 和 M. I. Jordan。局部隐私估计的极小极大最优程序。美国统计学会期刊，113:182–201，2016年。

[11] C. Dwork 和 A. Roth。《差分隐私的算法基础》。2014年。[12] C. Dwork、K. Kenthapadi、F. McSherry、I. Mironov 和 M. Naor。我们的数据，我们自己：通过分布式噪声生成实现隐私保护。第4004卷，第486–503页，2006年05月。ISBN 978-3-540-34546-6。doi: 10.1007/11761679 29。[13] C. Dwork、F. McSherry、K. Nissim 和 A. Smith。在隐私数据分析中根据敏感度校准噪声。第3876卷，第265–284页，2006年01月。ISBN 978-3-540-32731-8。doi: 10.1007/11681878 14。[14] U. Erlingsson、V. Pihur 和 A. Korolova。Rappor：随机化、可聚合的隐私保护序数响应。载于2014年 ACM SIGSAC 计算机与通信安全会议论文集，CCS '14，纽约，NY，美国，2014年，第1054–1067页。计算机协会。ISBN 9781450329576。doi: 10.1145/2660267.2660348。URL https://doi.org/10.1145/2660267.2660348。[15] H. Fang、D. Zhang、Y. Shu 和 G. Guo。序列推荐的深度学习：算法、影响因素与评估。ACM 信息系统汇刊 (TOIS)，39(1):1–42，2020年。[16] R. He 和 J. McAuley。结合相似性模型与马尔可夫链，实现稀疏序列推荐。载于2016年 IEEE 第16届国际数据挖掘会议 (ICDM)，第191–200页，2016年。doi: 10.1109/ICDM. 2016.0030。[17] X. He、H. Zhang、M. Kan 和 T. Chua。基于隐式反馈的在线推荐快速矩阵分解。CoRR，abs/1708.05024，2017年。URL http://arxiv.org/abs/1708.05024。[18] Y. Hu、Y. Koren 和 C. Volinsky。针对隐式反馈数据集的协同过滤。载于2008年第八届 IEEE 国际数据挖掘会议，第263–272页，2008年。doi: 10.1109/ICDM.2008.22。[19] P. Kairouz、H. B. McMahan、B. Avent、A. Bellet、M. Bennis、A. N. Bhagoji、K. Bonawit、Z. Charles、G. Cormode、R. Cummings、R. G. L. D'Oliveira、H. Eichner、S. El Rouayheb、D. Evans、J. Gardner、Z. Garrett、A. Gasc´on、B. Ghazi、P. B. Gibbons、M. Gruteser、Z. Harchaoui、C. He、L. He、Z. Huo、B. Hutchinson、J. Hsu、M. Jaggi、T. Javidi、G. Joshi、M. Khodak、J. Konecn´y、A. Korolova、F. Koushanfar、S. Koyejo、T. Lepoint、Y. Liu、P. Mittal、¨M. Mohri、R. Nock、A. Ozg¨ur、R. Pagh、H. Qi、D. Ramage、R. Raskar、M. Raykova、D. Song、W. Song、S. U. Stich、Z. Sun、A. Theertha Suresh、F. Tram`er、P. Vepakomma、J. Wang、L. Xiong、Z. Xu、Q. Yang、F. X. Yu、H. Yu 和 S. Zhao。《联邦学习的进展与开放问题》。2021年。[20] D. Kamisaka、S. Muramatsu、H. Yokoyama 和 T. Iwamoto。移动电话上下文感知用户界面的操作预测。2009年第9届应用与互联网年度国际研讨会论文集（SAINT 2009），第16–22页，2009年07月。doi: 10.1109/SAINT.2009.12。[21] D. P. Kingma 和 J. Ba。Adam：一种随机优化方法，2014年。URL https://arxiv.org/abs/ 1412.6980。[22] H. Li、X. Diao、J. Cao 和 Q. Zheng。基于全加权矩阵分解和快速优化的协同过滤推荐。IEEE Access，6:25248–25260，2018年。doi: 10.1109/ACCESS.2018.2828401。[23] W. Li、Y. Xia 和 H. Chen。关于 ARM TrustZone 的研究。GetMobile Mob. Comput. Commun.，22:17–22，2019年。[24] Z.-X. Liao、S.-C. Li、W.-C. Peng 和 P. Yu。智能手机应用使用预测的特征发现。IEEE 国际数据挖掘会议论文集 (ICDM)，2013年09月。doi: 10.1109/ICDM.2013.130。[25] M. Ludewig 和 D. Jannach。基于会话的推荐算法评估。CoRR，abs/1803.09587，2018年。URL http://arxiv.org/abs/1803.09587。[26] M. Ludewig 和 D. Jannach。基于会话的推荐算法评估。CoRR，abs/1803.09587，2018年。URL http://arxiv.org/abs/1803.09587。[27] L. Minto、M. Haller、H. Haddadi 和 B. Livshits。通过隐式反馈增强联邦协同过滤中的隐私保护。CoRR，abs/2105.03941，2021年。URL https://arxiv.org/abs/2105.03941。[28] A. Neelakantan、L. Vilnis、Q. V. Le、I. Sutskever、L. Kaiser、K. Kurach 和 J. Martens。添加梯度噪声可改善超深层网络的学习，2015年。URL https://arxiv.org/abs/1511.06807。[29] T. Nguyen、X. Xiao、Y. Yang、S. C. Hui、H. Shin 和 J. Shin。使用本地差分隐私收集并分析智能设备用户数据。ArXiv，abs/1606.05053，2016年。[30] A. N. Nikolakopoulos、V. Kalantzis 和 J. D. Garofalakis。EIGENREC：用于 Top-N 推荐的高效、可扩展潜在因子族。CoRR，abs/1511.06033，2015年。URL http://arxiv.org/abs/1511.06033。[31] N. Qian。梯度下降学习算法中的动量项。神经网络，12(1):145–151，1999年。ISSN 0893-6080。doi: https://doi.org/10.1016/S0893-6080(98)00116-6。URL https://www.sciencedirect.com/science/article/pii/S0893608098001166。[32] S. Rendle、C. Freudenthaler 和 L. Schmidt-Thieme。通过分解个性化马尔可夫链实现下一购物篮推荐。载于第19届万维网国际会议论文集，第811–820页，2010年。

[33] L. Reyzin、A. D. Smith 和 S. Yakoubov。将仇恨转化为爱：面向可扩展安全多方计算的紧凑型同态临时门限加密。载于*CSCML*，2021年。[34] Z. Shen、K. Yang、Z. Xi、J. Zou 和 W. Du。Deepapp：用于移动应用使用预测的深度强化学习框架。*IEEE*移动计算汇刊，第 1–1 页，2021年。doi：10.1109/TMC。2021.3093619。[35] C. Shin、J.-H. Hong 和 A. Dey。理解并预测智能手机的移动应用使用情况。*UbiComp'12——2012*年*ACM*普适计算会议论文集，第 173–182 页，2012年9月。doi：10.1145/2370216.2370243。[36] H. Shin、S. Kim、J. Shin 和 X. Xiao。采用本地差分隐私的隐私增强矩阵分解推荐。*IEEE*知识与数据工程汇刊，30(9)：1770–1782，2018年。doi：10.1109/TKDE.2018.2805356。[37] X. Su 和 T. M. Khoshgoftaar。协同过滤技术综述。人工智能进展，2009年，2009年1月。ISSN 1687-7470。doi：10.1155/2009/421425。URL https://doi.org/10.1155/2009/421425。[38] D. Valcarce、J. Parapar 和 A. Barreiro。用于基于相关性的推荐系统语言建模的加性平滑。载于第*4*届西班牙信息检索会议论文集，CERI '16，美国纽约州纽约市，2016年。计算机协会。ISBN 9781450341417。doi：10.1145/2934732.2934737。URL https://doi.org/10.1145/2934732.2934737。[39] S. Warner。随机响应：一种消除回避回答偏差的调查技术。美国统计学会期刊，60 309：63–6，1965年。[40] Y. Xu、M. Lin、H. Lu、G. Cardone、N. Lane、Z. Chen、A. Campbell 和 T. Choudhury。偏好、情境与社区：一种预测智能手机应用使用模式的多面向方法。载于*2013*年国际可穿戴计算机研讨会论文集，ISWC '13，第 69–76 页，美国纽约州纽约市，2013年。ACM。ISBN 978-1-4503-2127-3。doi：10.1145/2493988.2494333。URL http://doi.acm.org/10.1145/2493988。2494333。[41] A. C.-C. Yao。如何生成和交换秘密。载于第*27*届计算机科学基础年度研讨会（*SFCS 1986*），第 162–167 页，1986年。doi：10.1109/SFCS.1986.25。[42] D. Yu、Y. Li、F. Xu、P. Zhang 和 V. Kostakos。使用兴趣点进行智能手机应用使用预测。*ACM*交互式、移动、可穿戴与普适技术论文集，1(4)：174，2018年。[43] C. Zhang、X. Ding、G. Chen、K. Huang、X. Ma 和 B. Yan。Nihao：预测性智能手机应用启动器。计算机科学、社会信息学与电信工程研究所讲义，110：294–313，2013年1月。doi：10.1007/978-3-642-36632-1 17。[44] S. Zhang、Y. Tay、L. Yao 和 A. Sun。采用自注意力的下一项目推荐。*CoRR*，abs/1808.06414，2018年。URL http://arxiv.org/abs/1808.06414。[45] S. Zhao、Z. Luo、Z. Jiang、H. Wang、F. Xu、S. Li、J. Yin 和 G. Pan。AppUsage2Vec：为预测建模智能手机应用使用情况。载于*2019*年*IEEE*第*35*届数据工程国际会议（*ICDE*），第 1322–1333 页，2019年。doi：10.1109/ICDE.2019.00120。