IEEE INFOCOM 2024 - IEEE Conference on Computer Communications | 979-8-3503-8350-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/INFOCOM52122.2024.10621240

AeroRec: An Efficient On-Device Recommendation Framework using Federated Self-Supervised Knowledge Distillation

Tengxi Xia†‡,Ju Ren†§*,Wei Rao‡, Qin Zu‡, Wenjie Wang‡ Shuai Chen‡,Yaoxue Zhang†§
†Key Laboratory of Pervasive Computing, Ministry of Education, Department of
Computer Science and Technology, BNRist, Tsinghua University
‡Meituan Beijing,China
§Zhongguancun Laboratory, Beijing, 100094 China.
Email: {xtx19}@mails.tsinghua.edu.cn, {renju, zhangyx}@tsinghua.edu.cn,
{raowei,zuqin,WangWenjie22,chenshuai31}@meituan.com

[1] ***Abstract*—Modern recommendation systems operate entirely on the basis of central servers, requiring users to upload their behavior data from mobile devices to these servers. This practice has raised concerns about data privacy among users. Federated learning (FL), a machine learning technique designed to protect privacy, is becoming the standard solution. By combining federated learning with recommendation systems, Federated Recommendation Systems (FRS) allow users to collaboratively train a shared recommendation model without transmitting their original behavior data. However, existing federated learning solutions disregard the inherent limitations of resource-constrained mobile devices, which include limited storage space, computational overhead, and communication bandwidth. Although deploying a lightweight recommendation model can address the constraints of mobile devices, achieving satisfactory accuracy is difficult even with substantial communication overhead incurred during multiple rounds of federated learning training due to the limitations of the lightweight model's capabilities and the sparsity of recommendation data. To address this issue, we propose AeroRec, an efficient on-device Federated Recommendation Framework. In AeroRec, we use federated self-supervised distillation to enhance the global model after each parameter aggregation in each round. This approach not only accelerates the convergence rate but also surpasses the upper limit of the lightweight model's capability, thereby enabling a higher accuracy. We demonstrate that AeroRec outperforms several state-of-the-art FRS frameworks regarding recommendation accuracy and convergence speed through extensive experiments on three real-world datasets.**

***Index Terms*—Federated Recommendation, On-device Learning, Self-supervised Learning, Knowledge Distillation.**

I. Introduction

With the widespread use of smartphones, people can access recommendation services through their mobile devices anytime and anywhere. Recommendation systems have become an integral part of our daily lives, providing course recommendations on online learning platforms [1], [2], suggesting points of interest based on location-based services [3], [4], and offering micro-video recommendations on e-commerce platforms [5]. Balancing user preferences and privacy in recommendation systems presents a significant challenge [6]. To achieve satisfactory results with complex deep recommendation models on central servers, it is often necessary to collect personal data from millions of users for model training. This practice can lead to scalability issues, including user privacy concerns and significant storage resource requirements.

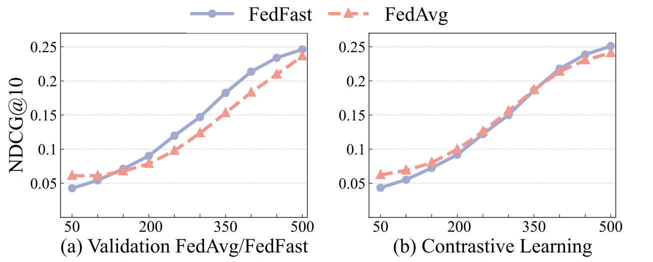

Fig. 1. (a) We evaluate the performance of the user model (Bert4Rec model) with FedAvg and FedFast algorithms on NDCG@10 and Recall@10 metrics. (b) We use contrastive learning to improve the performance of the client model.

Mobile device recommendation services face potential challenges in the current era marked by increasing concerns about data privacy. This is compounded by stringent regulations such as the General Data Protection Regulation (GDPR)[2] and the California Consumer Privacy Act (CCPA)[3]. These services often depend on data unrelated to the core operations of mobile commerce, such as item clicks [7] and page dwell times [8]. The GDPR stipulates that users grant clear and unambiguous consent for their data usage by recommendation systems, along with rights like data access, portability, and erasure. Consequently, many users, wary of privacy infringements, might withhold permission, impacting the efficacy of recommendation services. The safest way to protect user data privacy is by keeping their data locally and avoiding transmitting any intermediate results. Fortunately, federated learning offers a promising framework that allows model training without direct access to user data, ensuring enhanced

[1]*Corresponding author

[2]https://gdpr-info.eu/
[3]https://oag.ca.gov/privacy/ccpa/

privacy protection [9]. Some works have merged federated learning and recommendation systems to provide accurate recommendations to users while protecting their data privacy. For example, FedFast [10], FedRec [11], FedRecPP [12], FedRank [13], ContextFed [14], and FedSeq [15]. Even the most advanced federated recommendation systems have their limitations.

While they prioritize data privacy and personalization, they often overlook the impact of data sparsity on recommendation services. This issue is particularly significant in the context of mobile devices, which have inherent resource constraints such as limited storage space, computational overhead, and communication bandwidth. These constraints prevent the deployment of complex recommendation models on mobile devices, exacerbating the problem of data sparsity [16], [17]. To address this issue, we propose AeroRec, an efficient on-device federated recommendation framework. This framework, which does not involve uploading any raw data or intermediate results, ensures user privacy and security in compliance with GDPR. In addition, we draw inspiration from contrastive learning [16] and knowledge distillation [18] to propose the federated self-supervised knowledge distillation algorithm (Fed-SSKD), which addresses the impact of data sparsity on recommendation services in federated recommendation scenarios.

Initially, we constructed both a complex teacher recommendation model and a lightweight global recommendation model (student model) in the cloud. To train these models, we utilized some user interaction data collected before the implementation of GDPR. Following the introduction of GDPR, we deployed the lightweight recommendation model to each user's individual device. In the subsequent federated training process, users trained their recommendation models directly using the interaction behavior data from their devices. Once users completed their model training with local data, they transmitted the model parameters to the cloud for aggregation. Next, we applied a parameter enhancement technique, in conjunction with the teacher model, to augment the parameters of the global model. This augmented global model was then dispatched to individual devices, a process we termed "Federated Self-Supervised Knowledge Distillation (Fed-SSKD). Lastly, given the discrepancy between behavior data collected prior to the implementation of GDPR and current user behavior data, we have designed a personalized algorithm called Self-Contrastive Distillation (SCD). This algorithm enhances the model's personalization capabilities. Users can apply this algorithm to their local data, allowing them to fine-tune the global model they receive and achieve improved personalization.

By combining the above processes, we can train a lightweight recommendation model that ensures user data privacy, effectively tackles the challenge of data sparsity, and does not burden the user with additional communication overhead. To evaluate the performance of AeroRec, we conduct extensive experiments over three real-world publicly released user behaviour datasets, MovieLine-1M (ml-1m), MovieLine-20M (ml-20m) and Yelp datasets. Comprehensive experimental results verify that AeroRec achieves state-of-the-art performance compared with competitive methods. The contributions of this paper are summarized as follows:

1) We propose AeroRec, an efficient on-device federated recommendation framework, that enables to train high-quality on-device recommendation models for users without revealing user data privacy.
2) We propose the Fed-SSKD algorithm, specifically designed to counteract the effects of data sparsity on lightweight model performance, enhancing its generalization capability in federated learning contexts. Additionally, we introduce a Self-Contrastive Distillation (SCD) algorithm, which provides each user participating in the joint recommendation with a tailor-made recommendation service, without increasing communication overhead.
3) We conduct extensive experiments over three real-world publicly released user behavior datasets. Experimental results demonstrate that, compared to existing federated recommendation system algorithms, when using the Recall@5 metric, our method can enhance the model's generalization ability by up to 40% on the public dataset. Furthermore, compared to other methods, our approach requires 50% aggregation rounds to achieve the desired effect.

II. Background and Motivation

A. Recommendation in FL

Sequence Recommendation: Sequential Recommender systems (SRS) primarily model the sequential dependencies over user-item interactions (e.g., viewing or purchasing items on an online shopping platform) in a sequence, to suggest items that may be of interest to a user [19]–[22]. Consider the online-shopping events of Jack depicted in Fig.2 as an example. Before his tennis tournament, Jack purchased a tennis racket and sportswear, and also browsed sports watches. Given this sequence of actions, it's likely his next purchase will be sports shoes. Ideally, the shoes Jack chooses would be tailored for tennis and have a color that complements his sportswear. In this scenario, each of Jack's subsequent actions depends on the previous ones, establishing a sequential dependency across all four consumption actions. Likewise, we can see the sequential dependencies in Ada's case.

Generally, an SRS takes a sequence of user-item interactions as input and aims to predict subsequent user-item interactions that may occur in the near future. The sequential recommendation can be formulated as follows: given a sequence of user-item interactions, a recommendation list consisting of top-ranked candidate items is generated by maximizing the value of the utility function (e.g., the likelihood):

$$\mathcal{R} = \arg\max f(\mathcal{X}) \tag{1}$$

where f is a utility function to output a ranking score for the candidate items, and it could be of various forms, such as conditional probability [23], or an interaction score [24]. $\mathcal{X} = \{i_1, i_2, ..., i_{|\mathcal{X}|}\}$ is a sequence of user-item interactions where each interaction $i_j =< u, v, a >$ represents a triple

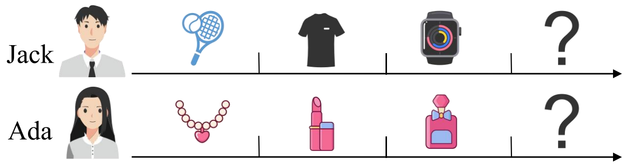

Fig. 2. Two examples of SRSs: (1) After Jack has viewed a tennis racket, a sportswear and a sports watch, what will be his next action? (2) After Ada has bought a necklace, a lipstick and a perfume, what would she buy next?

where u is a user, v is the corresponding item, and a is the user's action. In some cases, the user's actions can vary in type (e.g., click, add to cart, purchase) and occur under different contexts, such as time, location, or weather. The output $\mathcal{R}$ is a list of items ordered by the ranking score.

Knowledge Distillation: Knowledge Distillation (KD) provides a mechanism for transferring knowledge between models with different structures [25], [26]. Typical KD is used to transfer the knowledge of a pre-trained larger teacher model M_t to a smaller student model M_s relying on a shared dataset $\mathbb{D}$. The main idea is to allow M_s to mimic the behavior of M_t by encouraging M_s to approximate the output logits [4] of M_t. Generally, Kullback-Leibler (KL) divergence loss is used for student model training, which can be formulated as:

$$\min_{M_s} \mathbb{E}_{x_i \in \mathbb{D}} (D_{kl}(\psi(f(M_t; x_i)) || \psi(f(M_s; x_i)))) \quad (2)$$

where the $f(M_t; x_i)$, $f(M_s; x_i)$ are the output logits of teacher model M_t and students model M_s on sample x_i, respectively, $\psi(.)$ is a softmax function, D_{kl} is the KL divergence.

Self-Supervised Learning: Self-Supervised Learning: design a variety of pretext tasks, the labels for which can be derived directly from the data itself [27]. Solving these tasks makes the model learn to generate valuable representations [5]. These valuable representations are high-quality, meaningful features extracted from the data. Such features can be classified more accurately by the model, thereby enhancing its performance. Contrastive Learning (CL) is a technique rooted in self-supervised learning. The primary principle of CL is to use data augmentation methods to extract training signals from raw data, assisting in specific training tasks for model optimization. In the sequential recommendation task, data augmentation methods as shown in Fig.4, include Item Substituting, Item Shuffling, Item Masking, and Item Inserting [16]. Previous research has shown promising results using this method in the computer vision domain [28], [29], as well as in the field of recommendation systems [16], [30], [31]. The general pipeline of contrastive learning (CL) is depicted in Fig.3. Formally, self-supervised learning based on the contrastive method can be formulated as follows:"

$$\theta^{cl*}, \omega^{cl*} = arg \min_{\theta,\omega} \zeta_{con}(e_\omega(E_\theta(D^1)), e_\omega(E_\theta(D^2))) \quad (3)$$

where D^1 and D^2 are two generated data. $E_\theta(.)$ is the encoder to learn representations of samples in different data. $e_\omega(.)$ is

[4]Logits are the outputs of the last fully connected layer of a model.
[5]Representation is the input of the last fully connected layer of a model

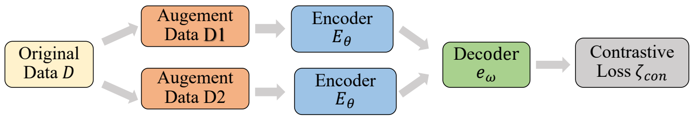

Fig. 3. Workflow of contrastive learning.

the decoder that estimates the agreement between two samples. ζ_{con} represents the contrastive loss.

Federated Learning: Federated Learning is a distributed machine learning approach in which clients collaboratively train a shared global model using their local data. This method mitigates privacy risks by eliminating the need for users to transmit their data to a central server. [32]. In classical FL, each user $k \in [\mathcal{K}]$ downloads the global model, trains the model with local dataset D_k, and uploads the model parameters w_u to the server. Then, central server performs parameters aggregation using the federated averaging (FedAvg) algorithm [9] to update the global model:

$$\mathbf{w} = \frac{\sum_{k=1}^{|\mathcal{K}|} |D_k| \mathbf{w}_k}{\sum_{k=1}^{|\mathcal{K}|} |D_k|} \quad (4)$$

In this way, model parameters as the carrier of knowledge are transferred between clients and the server, avoiding the privacy risk caused by the users transmitting data.

Recently, several studies have applied federated learning to recommender systems [11], [12], [33]. These approaches enhance recommendation accuracy by updating item and user embeddings via parameter aggregation. Some research utilizes data collected before the GDPR for pre-training on the cloud and employs the Automated Gradual Pruner (AGP) [34] to compress the size of the recommendation model [35].

B. Challenges in FRS

While updating user and item embeddings can accelerate convergence in Federated Recommendation Systems (FRS), achieving desirable recommendation effect in Sequence Recommendation with existing methods is nontrivial in real scenarios. On the one hand, complex recommendation models cannot be deployed on mobile devices due to the limitations of mobile device computing and storage resources. On the other hand, the sparsity of recommendation data significantly impacts the performance of sequential recommendation tasks, especially more pronounced in lightweight models. Specifically, we adopt the ml-1m dataset and deploy the Bert4Rec [24] model on the client. To adapt it to resource-constrained mobile devices, we set the embedding size to 32, reducing the model size to only 1.02M.

As illustrated in Fig.1(a), the sequence recommendation task displays a protracted convergence rate for the model. In the federated learning framework, each of the 500 rounds involves the participation of 1,000 users, resulting in a significant communication overhead for mobile devices. Notably, the NDCG@10 evaluation metric only reaches the value of 0.246. Therefore, augmenting the performance of lightweight models

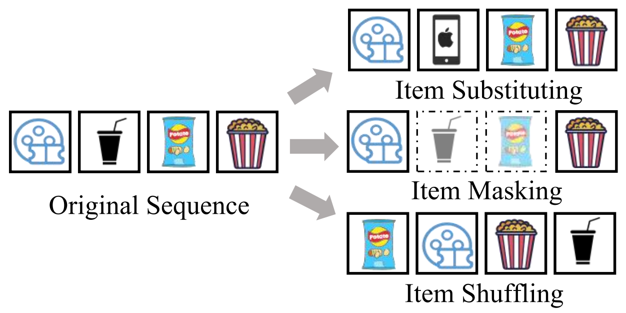

Fig. 4. A brief illustration of augmentation operations applied in our Fed-SSKD algorithm, including item Substituting, item mask, and item shuffling.

in federated sequence recommendation is the primary objective of the FedRec system. In a centralized training environment, contrastive learning has been widely recognized as an effective solution to the data sparsity issue, as discussed in [16], [36], [37]. Adapting this method to the Federated Recommendation Systems (FedRS) context, we facilitated each client to conduct evaluations using augmented data, maintaining experimental settings consistent with those documented in [16] and Fig.1(a). As depicted in Fig.1(b), the NDCG@10 evaluation metric merely attains a value of 0.251. Given the limitations posed by the data scale on clients and the inherent constraints of lightweight models, achieving the aspirational performance benchmarks for recommendation models in FedRS through contrastive learning proves to be a formidable challenge.

The existing literature has often overlooked the aforementioned challenges, leaving them unresolved. There is a pressing need for a robust Federated Recommendation framework. This framework should not only bolster the recommendation model's generalization capabilities but also cater to individual user requirements without amplifying their communication overhead.

III. System Overview

AeroRec Framework: To address the above challenges, we propose AeroRec, an efficient federated on-device recommendation framework as illustrated in Fig.5. Firstly, we train a teacher recommendation model M_t and a student recommendation model M_s (also known as the global model) using the existing interaction behavior data from all users, collected *before* the implementation of GDPR (D_h). Secondly, we push the global model to individual devices, allowing each user to train the model with their local data. Thirdly, users return their updated model parameters to the server, where an aggregation algorithm (i.e., FedAvg) aggregate these parameters. Then, we propose a Federated Self-Supervised Knowledge Distillation algorithm (Fed-SSKD) to enhance the global model's generalization ability. Specifically, we employ the pre-trained teacher model M_t and historical data D_h to augment the parameters of the aggregated model on the server, thereby improving its generalization performance. Finally, the server pushes the enhanced global model to each individual device, where users employ the Self-Contrastive Distillation algorithm (SCD) to adjust the model parameters according to their unique needs, return to the third step and repeats these steps.

IV. Design of AeroRec

In this section, we delve into the parameter enhancement mechanism of AeroRec. Subsequently, we detail a personalized algorithm tailored to fine-tune the model's parameters according to each user's distinctive requirements.

A. Parameters enhancement mechanism of AeroRec

As demonstrated in Fig.3, applying contrastive learning on the client side makes it challenging for FedRec to achieve satisfactory performance. Although Knowledge Distillation (KD) can boost the performance of student models with support from teacher models, implementing intricate teacher models on resource-limited mobile devices isn't practical.

Consequently, we introduce an advanced parameter enhancement algorithm that capitalizes on the server-side teacher model and historical data, enriching the student model's parameters. In the following, we will discuss our proposed parameter enhancement algorithm.

Our parameter enhancement algorithm draws inspiration from contrastive learning, known as Federated Self-Supervised Knowledge Distillation (Fed-SSKD). The critical difference between Fed-SSKD and KD is that the primary objective of Fed-SSKD is to enable the student model to generate valuable representations by emulating the teacher model. In contrast, the main goal of KD is to encourage the logits of the student model to approximate those of the teacher model.

B. Federated Self-Supervised Knowledge Distillation

The Fed-SSKD algorithm is executed after server parameter aggregation. We choose item shuffling and item substituting as the methods of data enhancement module. For any instance $x_u \in D_h$, the data augmentation module will generate two augmented instances, $x_u^{a_1}$ and $x_u^{a_2}$. Considering a mini-batch of $\mathbf{B}$ users $u_1, u_2, ..., u_{\mathbf{B}}$, then we utilize two augmentation metrics to each user's sequence and obtain augmented sequences $\mathbb{B} = \{x_{u_1}^{a_i}, x_{u_1}^{a_j}, x_{u_2}^{a_i}, x_{u_2}^{a_j}, ..., x_{u_{\mathbf{B}}}^{a_i}, x_{u_{\mathbf{B}}}^{a_j}\}$ and $|\mathbb{B}| = 2|\mathbf{B}|$.

Similarly to [16], for each user u, we consider $(x_u^{a_1}, x_u^{a_2})$ as the positive pair and treat the other $2\mathbf{B}$ augmented examples within the same mini-batch as negative samples, denoted as $\mathbf{B}^-$. With our data prepared, let p_s and p_t denote the predicted probability distributions (logits) of the teacher and student models, respectively, and z_s and z_t denote the representation of the teacher and student models. Then for each input x, we have:

$$\begin{aligned} p_s, z_s &= \omega_s(x) \\ p_t, z_t &= \omega_t(x) \end{aligned} \tag{5}$$

In the next, we will explain how to calculate the loss for Fed-SSKD.

Fed-SSKD Loss Function: Throughout the training process, the teacher model offers holistic guidance to steer the

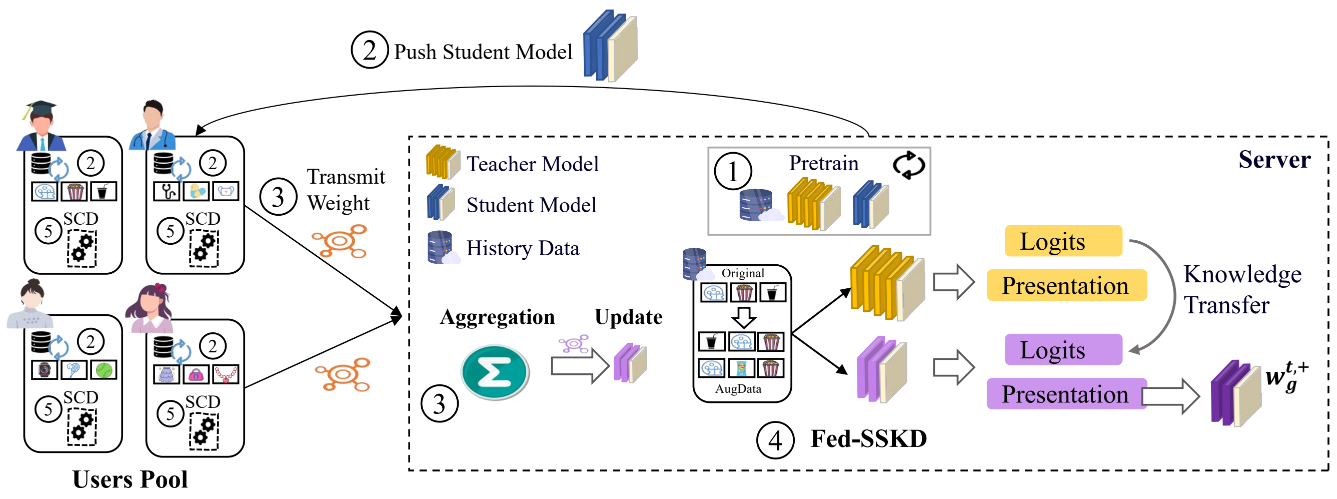

Fig. 5. The framework of AeroRec

development of the student model. Specifically, the loss function of Fed-SSKD $\mathcal{L}^+$ comprises three components, the main-task related loss, $\mathcal{L}_M$, the contrastive-distillation loss $\mathcal{L}_{cd}$ and the contrastive learning loss $\mathcal{L}_{cl}$.

$$\mathcal{L}^+ = \mathcal{L}_M + \mathcal{L}_{cd} + \mathcal{L}_{CL} \tag{6}$$

Main-task Loss: The first part is the main-task related loss $\mathcal{L}_M$, which is combined by student model's standard cross-entropy loss $\mathcal{L}_{ce}$ and the student model imitates the loss of the teacher model logits $\mathcal{L}_{cekd}$. The $\mathcal{L}_M$ can be formulated as follow:

$$\mathcal{L}_M = \mathcal{L}_{ce} + \mathcal{L}_{cekd} \tag{7}$$

the $\mathcal{L}_{ce}$ and $\mathcal{L}_{cekd}$ can be expressed as follow:

$$\mathcal{L}_{ce} = \frac{1}{|\mathcal{B}|} \sum_{x_u \in B} \mathcal{L}(p_s(x_u), y_u) \tag{8}$$

The $\mathcal{L}_{ce}$ is the standard cross-entropy loss, $|\mathcal{B}|$ is the size of a mini-batch, and $p_s(x) = \omega_s(x)$ is the logits of student mdoel input x.

$$\mathcal{L}_{cekd} = \frac{1}{|\mathcal{B}|} \sum_{x_u \in B} D_{kl}(p_s(x_u)||p_t(x_u))) \tag{9}$$

in $\mathcal{L}_{cekd}$, the ω_t is teacher model and ω_s is student model. $p_t(x) = \omega_t(x)$ denotes the logits of the teacher model given input x.

Contrastive Distillation Loss: The second part is the contrastive distillation loss $\mathcal{L}_{cd}$. This loss measures the discrepancy with the student model alongside the teacher model during the contrastive learning stage.

$$\mathcal{L}_{cd} = \mathcal{L}_{ce}^{CL} + \lambda_1 \mathcal{L}_{kd}^{CL} \tag{10}$$

This process comprises two parts. The first part is the standard cross-entropy loss of the student model during the contrastive learning phase $\mathcal{L}_{cd}^{CL}$, λ_1 is the weight of the student model imitating the teacher model, and the second part is the loss incurred when the student model imitates the logits of the teacher model $\mathcal{L}_{kd}^{CL}$. The $\mathcal{L}_{ce}^{CL}$ can be formulated as follow:

$$\mathcal{L}_{ce}^{CL} = \frac{1}{|\mathbb{B}|} \sum_{x_u \in \mathbf{B}} \sum_{i=1}^{|a_*|} \mathcal{L}(p_s(x_u^i), y_{u,i}^{cl}) \tag{11}$$

the $|\mathbb{B}|$ represents the size of the augmented sequences $\mathbb{B}$, x_u denotes a user in the mini-batch $\mathbf{B}$, $|a_*|$ is the number of methods used for data augmentation. Meanwhile, x_u^i refers to the data x_u augmented using the ith method, and $y_{u,i}^{cl}$ is the label generated by contrastive learning. The $\mathcal{L}_{kd}^{CL}$ can be shown as follow:

$$\mathcal{L}_{kd}^{CL} = \frac{1}{|\mathbb{B}|} \sum_{x_u \in \mathbf{B}} \sum_{i=1}^{|a_*|} D_{kl}(p_t(x_u^i)||p_s(x_u^i)) \tag{12}$$

the loss $\mathcal{L}_{kd}^{CL}$ is incurred when the student model ω_s imitates the logits from the teacher modelω_t during the contrastive learning phase.

Contrastive Loss: Finally, we introduce $\mathcal{L}_{CL}$, which distinguishes whether two representations originate from the same user's historical sequence. To achieve this, the contrastive loss minimizes the difference between differently augmented views of a single user's historical sequence and maximizes the difference between augmented sequences from different users.

$$\mathcal{L}_{CL} = \frac{1}{|\mathbf{B}|} \sum_{u \in \mathbf{B}} \phi(\mathcal{L}_{cl}(z_{u,s}^{a_i}, z_{u,s}^{a_j}), \mathcal{L}_{cl}(z_{u,t}^{a_i}, z_{u,t}^{a_j})) \tag{13}$$

we use $\mathcal{L}_{cl}(z_{u,*}^{a_i}, z_{u,*}^{a_j})$ to calculate he contrastive loss for each user. Here, $\phi(.)$ represents the cosine function. The representation from the student model is denoted by$z_{u,s}^* = \omega_s(x_{u,s}^*)$, while the representation from the teacher model is symbolized as $z_{u,t}^* = \omega_t(x_{u,t}^*)$. We use the dot product $sim(\mathbf{u}, \mathbf{v}) = \mathbf{u}^T\mathbf{v}$ to gauge the similarity between representations. Moreover, we introduce the function $es(z_u^{a_i}, z_u^{a_j}, \tau) = \exp(sim(z_u^{a_i}, z_u^{a_j})/\tau)$. Then, the loss function $\mathcal{L}_{cl}(z_u^{a_i}, z_u^{a_j})$ for a positive pair $(z_u^{a_i}, z_u^{a_j})$ can be defined as:

$$\mathcal{L}_{cl}(z_u^{a_i}, z_u^{a_j}) = \text{-}\log \frac{es(z_u^{a_i}, z_u^{a_j})}{es(z_u^{a_i}, z_u^{a_j}) + \sum_{x^- \in \mathbf{B}^-} es(z_u^{a_i}, z^-)}. \quad (14)$$

C. Self-Contrastive Distillation algorithm

To better cater to the distinct preferences of each user, we present the Self-Contrastive Distillation algorithm (SCD), a pioneering personalization algorithm designed for the client side. In Federated Recommender Systems (FRS), traditional methods frequently face challenges in swiftly adapting to the dynamic preferences of users [38]. Our method utilizes the local model weights from the $t-1$ round to guide the global model in the t, optimizing the embedding weights. This approach not only speeds up model training but also excels in precisely identifying users' preferences.

As illustrated in Fig.5 at step ⑤, during the second round, the user executes the SCD algorithm, incorporating our proposed SCD loss $\mathcal{L}_{scd}$. In the t-th round of FRS training, when user u engages in local training, they receive the global model ω^t from the server and deploy it into the local model as ω_u^t during the local training phase. For each input sequence x, we extract the representation of x from the global parameter ω^t, denoted as $z_g = \omega^t(x)$, the representation of x from the local model ω_u^t, given by $z_u^t = \omega_u^t(x)$, and the representation of x based on the previous round's local mode ω_u^{t-1}, expressed as $z_u^{t-1} = \omega_u^{t-1}(x)$. Drawing inspiration from MOON [39], we define the SCD loss as:

$$\mathcal{L}_{scd} = \text{-}\log \frac{es(z, z_u^{t-1})}{es(z, z_u^{t-1}) + es(z, z_g)} \quad (15)$$

where τ denotes a temperature parameter. The loss of each user is computed by

$$\mathcal{L}_u = \mathcal{L}_{ce} + \mu * \mathcal{L}_{scd} \quad (16)$$

where μ is a hyper-parameter to control the weight of scd loss. The comprehensive structure of the AeroRec framework is detailed in Algorithm 1.

Algorithm 1: The AeroRec framework

```
Input: number of communication rounds $T$, number
       of parties $K$ in each rounds, history data $D_h$,
       number of local epochs $E$, temperature $\tau$,
       learning rate $\eta$, teacher model $\omega_t$ and student
       model $\omega_s$.
Output: The final student model $\omega_s^{T,+}$
Server executes:
       Pretrain $\omega_t$ and $\omega_s$ with $D_h \rightarrow \omega_s^0$ and $\omega_t^*$.
foreach communication round $t = 0, 2, 3, .., T$-1 do
    foreach i=1,2,...,K in parallel do do
        if t=0 then
            $\omega_g^t \leftarrow \omega_s^0$
        else
            $\omega_g^t \leftarrow \omega_s^{t,+}$
        end
        send the global model $\omega_g^t$ to user $i$.
        $\omega_{s,i}^t \leftarrow$ LocalTraining($t$,$i$,$\omega_g^t$)
    end
    $\omega_s^{t+1} \leftarrow \sum_{i=1}^{K} \frac{|D^i|}{|D|} w_i^t$
    $\omega_s^{t+1,+} \leftarrow$ FedSSKD($\omega_t$,$\omega_s^{t+1}$)
end
return $\omega_s^{T,+}$
Functon LocalTraining($t$,$i$,$\omega_g^t$):
$\omega_{s,i}^t \leftarrow \omega_g^t$
foreach epoch i = 1,2,..,E do
    foreach each data $(x,y) \in D_i$ do
        if t=0 then
            $\ell \leftarrow \mathcal{L}_{ce}(x,y)$ (8)
        else
            $\ell \leftarrow \mathcal{L}_{ce}(x,y) + \mu * \mathcal{L}_{scd}(x,y)$ (16)
        end
        $\omega_{s,i}^t \leftarrow \omega_{s,i}^t - \eta * \Delta\ell$
    end
end
```

V. Experiments

In this section, we conduct extensive experiments to demonstrate the efficacy of AeroRec, which are compared with state-of-the-art baselines on three real-world tasks.

A. Experimental Setting

Datasets. We conduct experiments on three publicly accessible datasets: MovieLine-1M (ml-1m)[6], MovieLine-20M (ml-20m)[7] and Yelp[8]. All the datasets have been widely used in the recommender systems literature to evaluate sequence recommendation algorithms.

For dataset preprocessing, we follow the common practice in [10], [16], [40]. Initially, we excluded users with fewer than five interactions. All numeric ratings in a review were converted to "1", while others were set to "0". Duplicate interactions for each user were then eliminated. Afterward, we arranged the items from a user's history based on the time of interaction, proceeding in chronological order, to create a sequence of user interactions. Crucially, to ensure adequate interactions for each user/item, we adhered to the preprocessing methodology outlined in [41], [42], retaining only datasets with a "4-score". The statistics of the processed data are summarized in Table I.

Evaluation Settings. For performance evaluation, we use leave-one-out protocol, which is widely used [10], [43]. For each user, we reserved their most recent interaction for the test set. The item immediately preceding this was used for validation, while the remaining items constituted the training data. To streamline the often laborious process of ranking all items for each user during evaluation, we adopted a widely-used strategy as outlined in [10], [16], [44]. Specifically, this

[6]http://grouplens.org/datasets/movielens/1m/
[7]http://grouplens.org/datasets/movielens/20m/
[8]https://www.yelp.com/dataset/challenge

Algorithm 2: Federated Self-Supervised Knowledge Distillation

```
Input: $\omega_s^t$, $\omega_t^*$, number of Fed-SSKD epochs $E$ and $a^*$
        the number of data augmentation methods
Output: The final student model $\omega_s^{t,+}$
$\omega_s^{t,+} \leftarrow \omega_s^t$
foreach epoch $i$ = 1,2,..,$E$ do
    foreach each batch $B$={$x, y$} of $D_h$ do
        $x^*, y^*_{cl} \leftarrow$ DataAug($x, a^*$), Randomly select $a^*$
          methods to augment $x$
        $p_s^{a1}, z_s^{a1}, p_s^{a2}, z_s^{a2} \leftarrow \omega_s^t(x^{a1}), \omega_s^t(x^{a2})$
        $p_t^{a1}, z_t^{a1}, p_t^{a2}, z_t^{a2} \leftarrow \omega_t^*(x^{a1}), \omega_t^*(x^{a2})$
        $\mathcal{L}_M = \mathcal{L}_{ce}(x, y) + \mathcal{L}_{cekd}(x) \triangleright inEq.(7)$
        $\mathcal{L}_{cd} = \mathcal{L}_{ce}^{CL}(x^*) + \lambda_1 \mathcal{L}_{kd}^{CL}(x^*) \triangleright inEq.(10)$
        $\mathcal{L}_{CL}(z_s^{a*}, z_t^{a*}) \triangleright inEq.(13)$
        $\mathcal{L}^+ = \mathcal{L}_M + \mathcal{L}_{cd} + \mathcal{CL} \triangleright inEq.(6)$
        $\omega_s^{t,+} \leftarrow \omega_s^{t,+} - \eta\Delta\mathcal{L}^+$
    end
end
return $\omega_s^{t,+}$
```

TABLE I
STATISTICAL OVERVIEW OF THE EVALUATION DATASETS.

DataSet	Interaction	Item	User	Density
ml-1m	1,000,209	3706	6040	4.19%
ml-20m	20,000,263	27,278	138,493	0.53%
yelp	5,261,669	174,567	1,326,101	0.01%

approach entails randomly selecting 100 items with which the user hasn't previously interacted. The test item (the held-out item) is then ranked among this subset. This method offers an efficient way to gauge the model's performance without the need to benchmark against the complete item pool for every individual user. We employ Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) to evaluate the performance of each method, which are widely used in related works [40], [42]. HR focuses on the presence of the positive item, while NDCG further takes the rank position information into account. In this work, we report HR and NDCG with $k = [5, 10, 20]$.

Baselines. To verify the effectiveness of our method, we compare it with the following representative baselines:

- **FedSeq [15].** It is one of the representative Federated Recommendation System (FRS) baselines. It considers the congruence between the data distribution at the client level and the overall dataset. Moreover, it utilize low-rank tensor projections to capture users' long-standing preferences.
- **FedFast [10].** The approach serves as a foundational baseline in FRS. It focuses on clustering users based on their similarities, aiming to elevate the quality of recommendations.
- **DeepRec [35].** This is a deep recommendation model for end-cloud collaborative training. The approach employs AGP to compress the neural network model, enabling end-device deployment. Additionally, it effectively harnesses historical data from the period preceding the privacy protection regulations and integrates with FedAvg [9] for federated learning training.

Our evaluation target is to compare convergence speed and recommendation quality in a federated context, so we leave out comparison with other classical centralized algorithms because they are not easily amenable to work with federated learning.

B. Parameter Settings

For the recommender model, we employ bert4rec [24]. To simulate GDPR conditions, we randomly selected historical data D_h from 3,000, 5,000, and 50,000 users for the ml-1m, ml-20m, and Yelp datasets, respectively. This data served to pre-train both the student and teacher models. The data from the remaining users is exclusively reserved for on-device fine-tuning of the personal model, with any uploading to the cloud for global model training being strictly prohibited. We set the learning rate, τ, to 0.0001. The dimension of item embeddings in the student model was configured to 32, ensuring the recommender model's deployability on mobile devices. For the ml-1m and ml-20m datasets, the item embedding dimension in the teacher model was set to 512. In contrast, it was fixed at 256 for the Yelp dataset. In the centralized setting, we trained the student model using the entire dataset for 300 iterations. In contrast, within the federated context, we established communication rounds, T, at 300. During each communication round, the number of participating parties, K, in the federated learning training was capped at 50. Furthermore, we set the local training epoch for each user, denoted by E, at 3.

C. Overall performance evaluation

We evaluated AeroRec against the aforementioned baselines. The results on the three datasets are shown in Table II. From Table II, it's evident that AeroRec consistently outperforms other Federated baselines. We also observed that models trained using the standard federated framework often exhibit reduced performance compared to centralized learning.

The expectation is that the recommendation quality from a centrally trained Bert4Rec student model would be an upper-bound for AeroRec. Interestingly, AeroRec can compete with centrally trained student models in some cases. Unsurprisingly, AeroRec also performed better than FedAvg in this comparison, and in some cases, AeroRec even surpasses the efficiency of centralized training. One plausible explanation for AeroRec's enhanced performance over centralized training in particular situations can be attributed to the constraints imposed on the mobile device. Specifically, to facilitate the deployment of the recommendation model on mobile devices, the item embedding size in the student model is capped at 32. This size constraint significantly curtails the model's capacity to discern features. On top of that, AeroRec leverages the expertise of the teacher model to overcome these limitations, which makes our approach superior to the centralized training method in some cases.

TABLE II
RECOMMENDATION PERFORMANCE FOR AEROREC, FEDFAST, DEEPREC, FEDSEQ AND CENTRALIZED BASELINES. CONVERGENCE PLOTS FOR THESE RESULTS ARE PRESENTED IN FIGURE 8.

Dataset	Metric	AeroRec	FedFast	DeepRec	FedSeq	Centralised
ML-1M	NDCG@20	**0.5490**	0.4913	0.4990	0.4960	0.4998
	HR@20	**0.8400**	0.8152	0.8134	0.8230	0.8179
	NDCG@10	**0.5223**	0.4602	0.4703	0.4646	0.4702
	HR@10	**0.7351**	0.6921	0.7006	0.6987	0.7010
	NDCG@5	**0.4883**	0.4181	0.4288	0.4221	0.4299
	HR@5	**0.6301**	0.5628	0.5729	0.5682	0.5771
ML-20M	NDCG@20	**0.6945**	0.6569	0.6588	0.6369	**0.7513**
	HR@20	**0.9484**	0.9382	0.9251	0.9070	**0.9760**
	NDCG@10	**0.6799**	0.6403	0.6438	0.6224	**0.7420**
	HR@10	**0.8910**	0.8733	0.8465	0.8444	**0.9396**
	NDCG@5	**0.6521**	0.6069	0.6126	0.5923	**0.7189**
	HR@5	**0.8058**	0.7707	0.7710	0.7454	**0.8692**
Yelp	NDCG@20	**0.5591**	0.4519	0.5282	0.5138	0.5571
	HR@20	**0.9264**	0.8844	0.9125	0.8525	**0.9281**
	NDCG@10	**0.5294**	0.4035	0.4942	0.4813	0.5274
	HR@10	**0.8103**	0.6943	0.7795	0.7238	**0.8120**
	NDCG@5	**0.4775**	0.3392	0.4392	0.4372	0.4753
	HR@5	**0.6503**	0.4952	0.6099	0.5886	**0.6512**

D. *Personalization ability evaluation*

To assess the personalization effectiveness of AeroRec, we opted to evaluate users engaged in each federated learning training round rather than evaluating every user in every round. This evaluation approach computes the average recommendation performance for the selected test users. The reported results are an average drawn from five separate experiments. As illustrated in Fig.6, AeroRec consistently demonstrates remarkable personalization prowess. While both AeroRec and the other baselines are grounded on the same pre-trained model, AeroRec stands out by SCD algorithm. Across all datasets, AeroRec's performance sees a significant enhancement. When compared to DeepRec, AeroRec increases personalization by approximately 5%, and compared to FedFast, the improvement is around 12%.

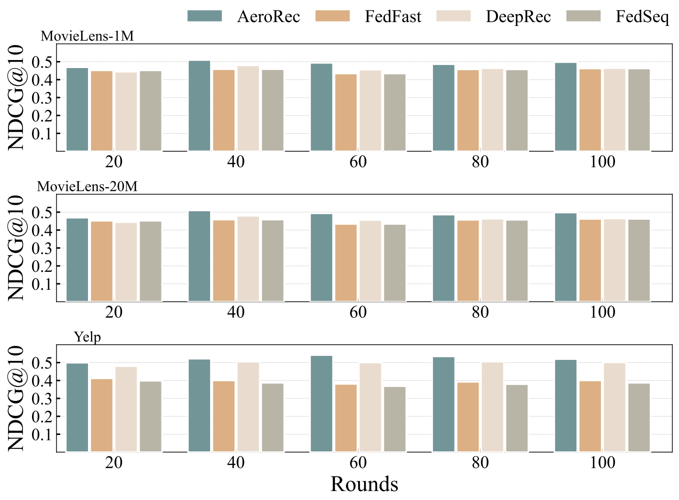

Fig. 6. Evaluating the Personalization ability of AeroRec and other baselines on the three datasets.

E. *Convergence Analysis*

As depicted in Fig.5, our AeroRec framework is uniquely designed to minimize extra communication overhead on the client's end. This experimentation further validates that, compared to other baselines, AeroRec accomplishes the intended accuracy with significantly reduced communication costs in a federated learning environment.

To gauge the system's efficacy across various rounds of federated learning, we've chosen NDCG@10 as our key evaluation metric. The performance benchmarks set for the datasets are as follows: 0.51 for ml-1M, 0.63 for ml-20M, and 0.53 for the Yelp dataset. We continued the federated learning training until the model achieved the targeted performance or arrive the ceiling of communication rounds. In our experiments ,we set the ceiling of communication rounds as 200. As highlighted in Fig.8, AeroRec meets the desired benchmarks in roughly 140 rounds, which marks a significant reduction in communication overhead compared to other baselines. It's worth noting that both FedFast and FedSeq, given their use of UserEmbedding, have model sizes that are slightly larger than those of AeroRec and DeepRec.

F. *Ablation experiment*

In this section, we take a closer look at the FedSSKD and SCD modules we've designed, gauging their individual impacts on recommendation performance. The experimental design and conditions adhere strictly to those detailed in Section V-D. In our experiments, 'w/oSCD' represents scenarios where the client forgoes the use of the SCD algorithm during local training. Conversely, 'w/oFedSSKD' indicates

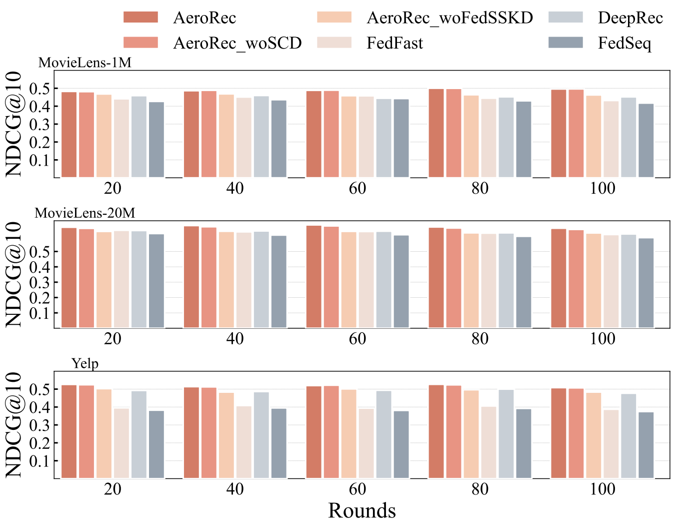

Fig. 7. Ablation experiment.

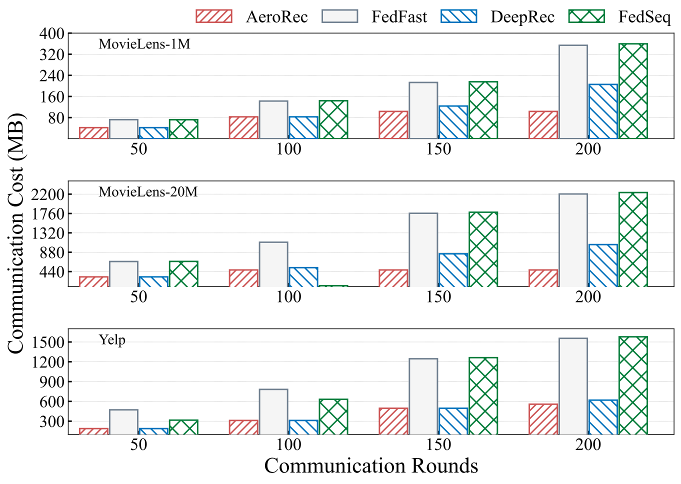

Fig. 8. Communication cost of AeroRec and other baselines on the evaluation dataset to achieve desired performance.

cases where the server opts out of refining the global model's parameters.

As depicted in Fig.7, our observations from the 'Aero_w/oFedSSKD' setup (excluding the cloud parameter enhancement algorithm) reveal a noticeable performance increase of 0.2% to 2% over DeepRec for the ml-1m dataset. Yet, for the ml-20m dataset, it lags behind DeepRec by a marginal 0.03%. Notably, on the Yelp dataset, 'Aero_w/oFedSSKD' achieves its peak performance, showcasing an improvement of up to 2%.

In contrast, the 'Aero_w/oSCD' setup, which eschews the client-side personalization algorithm, exhibits a notably robust performance. For the ml-1m dataset, it outperforms DeepRec by a significant 5% to 7%. On the ml-20m dataset, it boasts a lead of 2% to 5%. Finally, on the Yelp dataset, this configuration yields a 5% improvement over DeepRec.

VI. Related Work

Knowledge Distillation. Previous studies on KD mainly focus on the Computer Vision (CV) domain, they are usually in the transfer of knowledge from a large teacher model to a smaller student model [25]. Xv and Liu [27] observed that the teacher model assists the lightweight student model better distinguishing categorical features within contrastive learning. Tang and Wang [45] proposed a KD model to address the ranking problem, called rank distillation (RD). RD uses only a few items with the highest rankings in the label distribution learned from the teacher model. To address this, they introduced a rank-aware sampling method that facilitates cooperative training between the teacher and student models. However, direct application of these studies is challenged by the fact that client-side models rely on the client's localized private data, which is often both limited in scope and statistically diverse.

Federated Recommendation System. Recently, several researches explored combining FL with Recommendation system to allow training recommendation models with local data across clients to improve the learning performance of FL. Specially, in FedCF [33] consider how to apply Collaborative Filtering (CF) to federated learning scenarios. FedRec [11] and FedFast [10] explore various methodologies for updating item embeddings within a federated learning context. FedSeq [15] integrates low-rank tensor projections with federated sequence recommendation. Nevertheless, these works always with various limitations. FedSeq doesn't consider mobile device user. FedCF and FedRec cannot achieve satisfactory results in ranking 100 items. In contrast, AeroRec paves the way for lightweight recommendation models to deliver commendable results with a significantly reduced communication overhead. In contrast, our AeroRec framework paves the way for lightweight recommendation models to deliver commendable results with a significantly reduced communication overhead.

VII. Conclusion

In this paper, we proposed AeroRec, an efficient on-device recommendation framework that jointly considers the resource-constrained features of mobile devices and the sparsity of recommendation data. Accordingly, we introduce a self-supervised knowledge distillation algorithm, FedSSKD, tailored for federated learning scenarios, aiming to enhance the generalization capability of the global model. Additionally, we put forth a personalized algorithm, SCD, which utilizes the insights from the previous training round to guide the current round, thereby augmenting user-specific adaptability. Looking ahead, our intent is to expand the applicability of AeroRec to a broader spectrum of use cases. We also aspire to amplify its learning efficacy and resource optimization by contemplating clustering and bidirectional distillation mechanisms.

Acknowledgements

This research was supported in part by the National Key RD Program of China under Grant No. 2022YFF0604502, the National Natural Science Foundation of China under Grant No. 62122095, 62341201, 62072472, U19A2067, and by a grant from the Guoqiang Institute, Tsinghua University.

REFERENCES

[1] Dianshuang Wu, Jie Lu, and Guangquan Zhang. A fuzzy tree matching-based personalized e-learning recommender system. *IEEE TFS*, 23(6):2412–2426, 2015.

[2] Jibing Gong, Shen Wang, Jinlong Wang, Wenzheng Feng, Hao Peng, Jie Tang, and Philip S Yu. Attentional graph convolutional networks for knowledge concept recommendation in moocs in a heterogeneous view. In *ACM SIGIR*, pages 79–88, 2020.

[3] Yong Liu, Wei Wei, Aixin Sun, and Chunyan Miao. Exploiting geographical neighborhood characteristics for location recommendation. In *ACM CIKM*, pages 739–748, 2014.

[4] Peng Han, Zhongxiao Li, Yong Liu, Peilin Zhao, Jing Li, Hao Wang, and Shuo Shang. Contextualized point-of-interest recommendation. ACM IJCAI, 2020.

[5] Chenyi Lei, Yong Liu, Lingzi Zhang, Guoxin Wang, Haihong Tang, Houqiang Li, and Chunyan Miao. Semi: A sequential multi-modal information transfer network for e-commerce micro-video recommendations. In *ACM SIGKDD*, pages 3161–3171, 2021.

[6] Bo Zhang, Na Wang, and Hongxia Jin. Privacy concerns in online recommender systems: influences of control and user data input. In *USENIX SOUPS*, pages 159–173, 2014.

[7] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2015.

[8] Veronika Bogina and Tsvi Kuflik. Incorporating dwell time in session-based recommendations with recurrent neural networks. In *RecTemp@ RecSys*, pages 57–59, 2017.

[9] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282, 2017.

[10] Khalil Muhammad, Qinqin Wang, Diarmuid O'Reilly-Morgan, Elias Tragos, Barry Smyth, Neil Hurley, James Geraci, and Aonghus Lawlor. Fedfast: Going beyond average for faster training of federated recommender systems. In *ACM SIGKDD*, pages 1234–1242, 2020.

[11] Guanyu Lin, Feng Liang, Weike Pan, and Zhong Ming. wfedrec: Federated recommendation with explicit feedback. *IEEE IS*, 36(5):21–30, 2021.

[12] Feng Liang, Weike Pan, and Zhong Ming. Fedrec++: Lossless federated recommendation with explicit feedback. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 4224–4231, 2021.

[13] Vito Walter Anelli, Yashar Deldjoo, Tommaso Di Noia, Antonio Ferrara, and Fedelucio Narducci. Federank: User controlled feedback with federated recommender systems. In *Springer ECIR*, pages 32–47. Springer, 2021.

[14] Waqar Ali, Rajesh Kumar, Zhiyi Deng, Yansong Wang, and Jie Shao. A federated learning approach for privacy protection in context-aware recommender systems. *The Computer Journal*, 64(7):1016–1027, 2021.

[15] Li Li, Fan Lin, Jianbing Xiahou, Yuanguo Lin, Pengcheng Wu, and Yong Liu. Federated low-rank tensor projections for sequential recommendation. *ACM KBS*, 255:109483, 2022.

[16] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. Contrastive learning for sequential recommendation. In *2022 IEEE 38th international conference on data engineering (ICDE)*, pages 1259–1273. IEEE, 2022.

[17] Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Guandong Xu, and Quoc Viet Hung Nguyen. On-device next-item recommendation with self-supervised knowledge distillation. In *ACM SIGIR*, pages 546–555, 2022.

[18] SeongKu Kang, Junyoung Hwang, Wonbin Kweon, and Hwanjo Yu. De-rrd: A knowledge distillation framework for recommender system. In *ACM ICKM*, pages 605–614, 2020.

[19] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z Sheng, and Mehmet Orgun. Sequential recommender systems: challenges, progress and prospects. *arXiv preprint arXiv:2001.04830*, 2019.

[20] Hui Fang, Danning Zhang, Yiheng Shu, and Guibing Guo. Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations. *ACM TOIS*, 39(1):1–42, 2020.

[21] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. Sequence-aware recommender systems. *ACM CSUR*, 51(4):1–36, 2018.

[22] Shoujin Wang, Longbing Cao, Yan Wang, Quan Z Sheng, Mehmet A Orgun, and Defu Lian. A survey on session-based recommender systems. *ACM CSUR*, 54(7):1–38, 2021.

[23] Shoujin Wang, Liang Hu, Longbing Cao, Xiaoshui Huang, Defu Lian, and Wei Liu. Attention-based transactional context embedding for next-item recommendation. In *ISSN AAAI*, volume 32, 2018.

[24] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *ACM CIKM*, pages 1441–1450, 2019.

[25] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? *Advances in neural information processing systems*, 27, 2014.

[26] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[27] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge distillation meets self-supervision. In *Springer ECCV*, pages 588–604. Springer, 2020.

[28] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[29] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. In *IEEE CVPR*, pages 9653–9663, 2022.

[30] Fangye Wang, Yingxu Wang, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, and Ning Gu. Cl4ctr: A contrastive learning framework for ctr prediction. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 805–813, 2023.

[31] Mengyuan Jing, Yanmin Zhu, Tianzi Zang, and Ke Wang. Contrastive self-supervised learning in recommender systems: A survey. *arXiv preprint arXiv:2303.09902*, 2023.

[32] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. *ACM KBS*, 216:106775, 2021.

[33] Le Wang, Zijun Huang, Qingqi Pei, and Shen Wang. Federated cf: Privacy-preserving collaborative filtering cross multiple datasets. In *IEEE ICC*, pages 1–6. IEEE, 2020.

[34] Michael Zhu and Suyog Gupta. To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*, 2017.

[35] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM CSUR*, 52(1):1–38, 2019.

[36] Yongjun Chen, Zhiwei Liu, Jia Li, Julian McAuley, and Caiming Xiong. Intent contrastive learning for sequential recommendation. In *Proceedings of the ACM Web Conference 2022*, pages 2172–2182, 2022.

[37] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020.

[38] Chuhan Wu, Fangzhao Wu, Yang Cao, Yongfeng Huang, and Xing Xie. Fedgnn: Federated graph neural network for privacy-preserving recommendation. *arXiv preprint arXiv:2102.04925*, 2021.

[39] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *IEEE CVPR*, pages 10713–10722, 2021.

[40] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *IEEE ICDM*, pages 197–206. IEEE, 2018.

[41] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *ACM CIKM*, pages 1893–1902, 2020.

[42] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *ACM WWW*, pages 811–820, 2010.

[43] Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM TOIS*, 22(1):143–177, 2004.

[44] Walid Krichene and Steffen Rendle. On sampled metrics for item recommendation. In *ACM SIGKDD*, pages 1748–1757, 2020.

[45] Jiaxi Tang and Ke Wang. Ranking distillation: Learning compact ranking models with high performance for recommender system. In *ACM SIGKDD*, pages 2289–2298, 2018.