Mechanism Design for a Sustainable Federated Session Recommender System

Runchen Xu[1], Mengxiao Zhang[1] (✉), and Jiamou Liu[1]

School of Computer Science, The University of Auckland, Auckland, NZ
rxu274@aucklanduni.ac.nz, mengxiao.zhang@auckland.ac.nz,
jiamou.liu@auckland.ac.nz

Abstract. Federated learning enables privacy-preserving session-based recommendation by keeping raw interactions on devices, but limited budget and heterogeneous participation costs make client selection nontrivial. We formulate federated session recommendation as a budget-feasible procurement problem and propose a unified framework that integrates heterogeneous graph learning with a truthful reverse auction. Each client constructs a local heterogeneous knowledge graph to model long-term preferences and short-term session intent, while the server estimates client value using a novelty-centered signal on a public proxy dataset without accessing raw session data. We establish incentive compatibility, individual rationality, and budget feasibility, and analyze convergence under non-IID session data.

Keywords: Federated learning, Session-based recommendation, Mechanism design

1 Introduction

Modern recommendation services must provide real-time personalization while complying with data protection regulations such as the General Data Protection Regulation (GDPR). These regulations promote data minimisation and restrict the centralized collection of user interaction data [2]. This is especially important for session-based recommendation, where the goal is to infer a user's short-term intent from a small sequence of anonymous clicks and transitions [8]. Federated learning offers a natural solution by keeping raw interactions on client devices and learning from shared model updates instead of centralized data [14].

However, privacy-preserving decentralization does not remove the need for incentives. In federated systems, clients differ in their willingness to contribute, their privacy concerns, and the burden of participating in training rounds. Moreover, a platform usually has a limited incentive budget and cannot reward all contributors equally. As a result, participation is better viewed as a strategic decision rather than a guaranteed one [27, 16]. This makes federated session-based recommendation not only a learning problem, but also an incentive allocation problem: the server must identify and reward the most informative client updates under a fixed budget.

This setting lies at the intersection of session-based recommendation, federated learning, and mechanism design. Session-based recommendation has progressed from recurrent models to graph-based methods such as SR-GNN and GC-SAN, which capture high-order item transitions more effectively [21, 23]. Federated recommendation extends these ideas to decentralized settings, and recent work has incorporated graph structure and social information to reduce sparsity, such as FeSoG and FedPerGNN [11, 20]. Early work such as SFedRec adapts heterogeneous graph modelling to decentralized session data [25]. Yet strategic participation remains underexplored. Most existing studies focus on privacy, heterogeneity, and communication efficiency, while treating client participation as given [19]. At the same time, incentive mechanisms for federated learning often rely on coarse value proxies such as data volume [27]. For session-based recommendation, this is inadequate, because the value of a client update depends less on data size than on whether it reveals useful and previously underrepresented intent patterns [8].

This motivates the following research question: How can we design a truthful and budget-feasible federated session-based recommendation system that rewards informative client contributions without accessing raw interaction data? Addressing this question requires solving two challenges. First, the server must estimate the value of a client update without inspecting private sessions [10]. Second, the incentive mechanism must respect a fixed budget and encourage truthful reporting of private participation costs [15, 18].

To address these challenges, we propose SFRec, a unified framework that combines heterogeneous graph learning with a reverse auction. Each client builds a local heterogeneous knowledge graph to capture both long-term preferences and short-term session intent. The server then evaluates client contributions through a value signal derived from predictive behaviour on a public proxy dataset, without accessing raw local interactions. Based on this signal and the reported costs, the mechanism selects and compensates clients under a fixed budget while preserving standard economic guarantees. In this way, SFRec connects representation learning with incentive-aware client selection in federated session-based recommendation.

Our main contributions are as follows.

- A budget-constrained procurement formulation of federated session-based recommendation under strategic participation.
- A novelty-and-representativeness value metric for privacy-preserving estimation of client contribution.
- A heterogeneous graph learning architecture for long-term preference and short-term session modelling in decentralized settings.
- Theoretical guarantees of IC, IR, and BF, with convergence analysis under non-IID session data, and consistent empirical improvements on three real-world datasets.

2 Related Work

Session-Based Recommendation. Session-based recommendation models short-term intent from interaction sequences. Early methods used recurrent networks, such as GRU4Rec [3], while later models introduced attention mechanisms, such as NARM and STAMP [5, 9]. Recent advances have largely been graph-based [8]. SR-GNN and GC-SAN model item transitions and long-range dependencies on session graphs [21, 23], while DHCN, HIDE, and LS-TGNN further improve representation power through hypergraphs, intent disentanglement, and temporal modelling [22, 7, 17]. However, most SBR methods assume centralized data, which is incompatible with privacy-sensitive settings. Our work extends graph-based SBR to a federated setting with strategic clients.

Federated Learning for Recommendations. Federated learning enables collaborative training without centralizing raw data [14]. In recommendation, a main challenge is strong statistical heterogeneity across clients [6, 19]. Existing methods address this through personalized components or graph-based modelling. FeSoG and FedPerGNN use relational structure to reduce sparsity [11, 20]. SFedRec extends heterogeneous graph modelling to federated session data, and FedGA improves aggregation robustness [25, 12]. However, these methods generally assume participation is given and do not address incentive-aware selection under budget constraints.

Mechanism Design for Model Marketplaces. Mechanism design offers tools for incentive compatibility and budget feasibility in federated learning [15, 18, 27]. Existing methods often value clients using coarse proxies such as quality, reputation, data volume, or local accuracy [1, 26, 4]. These signals are insufficient for session-based recommendation, where useful updates depend on intent patterns rather than dataset size alone. More recent work considers richer valuation signals, such as structural importance or distributional information [10, 24]. Still, prior work does not integrate such valuation with budget-feasible procurement for federated session-based recommendation. Our framework addresses this gap by combining intent-aware contribution evaluation with auction-based client selection.

3 Problem Formulation

We consider a federated learning system consisting of a central server and a set of n clients, denoted by $\mathcal{U}$, together with a set of items, denoted by $\mathcal{I}$. Each client $u \in \mathcal{U}$ holds a private local dataset comprising multiple sessions, denoted by $\mathcal{D}_u$. Each session $\boldsymbol{s}_{u,r} \in \mathcal{D}_u$ is a sequence of items clicked by u. The server additionally maintains a public proxy dataset $\mathcal{Q}$ containing non-sensitive and anonymized sessions, constructed from public or consented historical interactions over the same item domain, so that it provides a shared reference for valuation without exposing clients' private data. Each client $u \in \mathcal{U}$ has a private cost $\theta_u \in \Theta$, where θ_u represents the minimum payment required for participating in a training round by submitting a locally trained model update, and $\Theta \subseteq \mathbb{R}_+$ denotes the

Table 1: Key Notations

Notation	Definition
$\mathcal{U}$	Set of clients
$\mathcal{I}$	Set of items
$\mathcal{D}_u$	Private local dataset of client u
$\boldsymbol{s}_{u,r}$	r-th session of client u
$\mathcal{Q}$	Public proxy dataset
θ_u	True private cost of client u for participation
θ'_u	Reported cost of client u
β	Per-round budget of the server
f_g^t	Global model parameters at round t
f_u^t	Local model parameters of client u at round t
$\mathcal{G}_u$	Local Heterogeneous Knowledge Graph (LHKG) of client u
$\mathcal{V}_u$	Node set of $\mathcal{G}_u$
$\mathcal{E}_u$	Edge set of $\mathcal{G}_u$
ϕ_u	Node type mapping function
φ_u	Edge type mapping function
ω_u	Edge weight function
$\boldsymbol{h}_u^{\text{long}}$	Long-term preference embedding of client u
$\boldsymbol{h}_u^{\text{short}}$	Short-term preference embedding of client u
$\mathsf{score}_u^{\text{nov}}$	Novelty score of client u
$\mathsf{score}_u^{\text{rep}}$	Representativeness score of client u
α	Dynamic balancing parameter for novelty and representativeness
score_u	Comprehensive score of client u
$\mathcal{S}^t$	Selected clients in round t
p_u	Payment to selected client u
π_u	Utility of client u

domain of feasible costs. In each federated learning round, the server has a fixed budget $\beta \in \mathbb{R}_+$ to procure model contributions from clients. A summary of key notations used is shown in Table 1.

Federated Session Recommendation. Each round of federated session recommendation proceeds as follows. At the beginning of round t, the central server broadcasts the current global model parameters f_g^{t-1} to all candidate clients. Each client $u \in \mathcal{U}$ trains a local model f_u^t on its private data $\mathcal{D}_u$ by minimizing a session recommendation loss $\mathcal{L}_u$. The server then evaluates the contribution of local updates without accessing raw data, based on the public proxy dataset $\mathcal{Q}$, and selects a subset of clients $\mathcal{S}^t \subseteq \mathcal{U}$ to participate in the round. Their payments are determined according to the reported costs and the measured contribution of their updates (defined in Sec. 4). The server collects the local model updates from clients in $\mathcal{S}^t$ and aggregates them to obtain the next global model f_g^t.

Incentive Mechanism. Each client $u \in \mathcal{U}$'s true cost θ_u is private, and u reports a declared cost $\theta'_u \in \Theta$ to the server. clients may misreport, i.e., choose $\theta'_u \neq \theta_u$, whenever doing so is beneficial. An incentive mechanism is used to decide which clients are selected and what payments they receive, based on the reported costs. Formally, an incentive mechanism is specified by a pair of two core functions: an allocation function and a payment function. The reported cost profile of all clients is denoted by $\boldsymbol{\theta'} = (\theta'_1, \ldots, \theta'_n) \in \Theta^n$.

Definition 1. *An* incentive mechanism $\mathcal{M} = (\boldsymbol{q}, \boldsymbol{p})$ *is specified by an* allocation function $\boldsymbol{q}$ *and a* payment function $\boldsymbol{p}$*, where*
- *the allocation function is* $\boldsymbol{q} : \Theta^n \to \{0,1\}^n$*, with* $\boldsymbol{q}(\boldsymbol{\theta'}) = (q_1(\boldsymbol{\theta'}), \ldots, q_n(\boldsymbol{\theta'}))$ *and* $q_u(\boldsymbol{\theta'}) \in \{0,1\} \forall u \in \mathcal{U}$*;*
- *the payment function is* $\boldsymbol{p} : \Theta^n \to \mathbb{R}^n$*, with* $\boldsymbol{p}(\boldsymbol{\theta'}) = (p_1(\boldsymbol{\theta'}), \ldots, p_n(\boldsymbol{\theta'}))$ *and* $p_u(\boldsymbol{\theta'}) \in \mathbb{R}_+ \forall u \in \mathcal{U}$*.*

We interpret $q_u(\boldsymbol{\theta'}) = 1$ as selecting client u to participate in the round, and $q_u(\boldsymbol{\theta'}) = 0$ otherwise. The term $p_u(\boldsymbol{\theta'})$ denotes the payment made by the server to client u, and unselected clients receive zero payment, i.e., $p_u(\boldsymbol{\theta'}) = 0$ whenever $q_u(\boldsymbol{\theta'}) = 0$. For each client $u \in \mathcal{U}$, the utility is defined as the net gain from participation: $\pi_u\left(\theta_u, \boldsymbol{\theta'}_{-u}\right) = p_u\left(\theta'_u, \boldsymbol{\theta'}_{-u}\right) - \theta_u \cdot q_u\left(\theta'_u, \boldsymbol{\theta'}_{-u}\right)$, where $\boldsymbol{\theta'}_{-u} = \left(\theta'_1, \ldots, \theta'_{u-1}, \theta'_{u+1}, \ldots, \theta'_n\right)$ denotes the reported costs of all clients except u.

To ensure sustainability in the federated session recommendation system, the proposed mechanism should satisfy the following three properties:
- *Incentive Compatibility* (IC): Truthful reporting is a dominant strategy for every client. Formally, $\pi_u\big(\theta_u, (\theta_u, \boldsymbol{\theta'}_{-u})\big) \geq \pi_u\big(\theta_u, (\theta'_u, \boldsymbol{\theta'}_{-u})\big), \forall u \in \mathcal{U}, \forall \theta_u, \theta'_u \in \Theta, \forall \boldsymbol{\theta'}_{-u} \in \Theta^{n-1}$.
- *Individual Rationality* (IR): Truthful reporting obtains non-negative utility for every client. That is, $\pi_u\big(\theta_u, (\theta_u, \boldsymbol{\theta'}_{-u})\big) \geq 0, \forall u \in \mathcal{U}, \forall \theta_u \in \Theta, \forall \boldsymbol{\theta'}_{-u} \in \Theta^{n-1}$.
- *Budget Feasibility* (BF): The total payment made by the server in each round does not exceed the budget β. Formally, $\sum_{u \in \mathcal{U}} p_u(\boldsymbol{\theta'}) \leq \beta, \forall \boldsymbol{\theta'} \in \Theta^n$.

We aim to design a budget-feasible incentive mechanism for federated session-based recommendation that supports decentralized training under privacy constraints while accounting for strategic client participation. The mechanism should ensure incentive compatibility, individual rationality, and per-round budget feasibility, so that the server can select and reward clients to learn an effective and sustainable global model.

4 SFRec Mechanism

Federated session-based recommendation requires both effective local modelling and incentive-aware client selection under budget constraints. To address this, we propose SFRec (**s**ustainable **f**ederated **rec**-ommendation for sessions), a unified framework that combines local representation learning with budget-constrained participant selection. SFRec proceeds over $t_{\max}$ training rounds. In each round, the server broadcasts the current global model, clients perform local processing and upload non-sensitive signals, and the server evaluates these signals to select participants through a reverse auction. The selected local models are then aggregated to update the global model, and payments are determined accordingly. Figure 1 and Algorithm 1 give an overview of the framework. The following subsections describe main components of the mechanism.

4.1 Local Processing Component

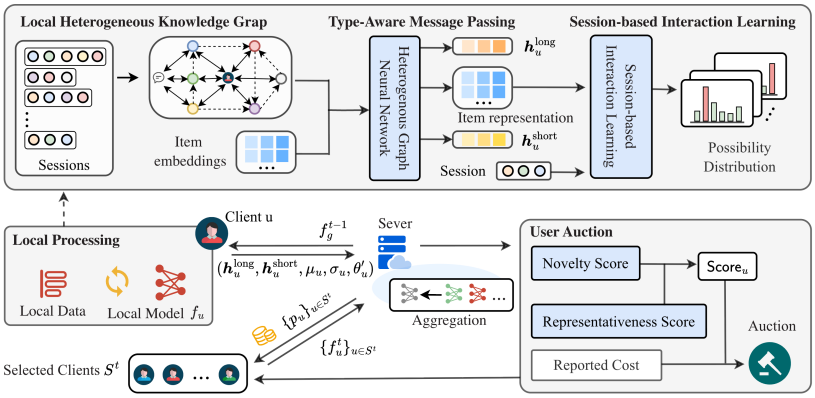

Fig. 1: Overall Framework of SFRec .

Algorithm 1 Overall Framework of SFRec

Input: Initial global model f_g^0, public proxy dataset $\mathcal{Q}$, per-round budget β, client set $\mathcal{U}$, total rounds $t_{\max}$
Output: Final global model $f_g^{t_{\max}}$, payments $\{p_u\}_{u\in\mathcal{S}^t}$ in each round

```
Initialize global model f_g^0
for t = 1 to t_max do
    Server broadcasts f_g^{t-1} and Q to all clients u ∈ U
    Clients execute local processing and upload attribute information to the server
    Server computes client novelty and representativeness scores, then calculates
    comprehensive scores score_u
    Server runs budget-constrained reverse auction to select S^t ⊆ U
    Server aggregates local models of clients in S^t to update global model f_g^t
    Server distributes payments {p_u}_{u∈S^t}
end for
```

The local processing component trains client-side models using private session data. At the beginning of round t, each client $u \in \mathcal{U}$ receives the latest global model f_g^{t-1} and the public proxy dataset $\mathcal{Q}$ from the server. client u then constructs an LHKG from its private session data $\mathcal{D}_u$ and applies a type-aware message-passing scheme to learn representations that capture both long-term and short-term preferences. Based on these representations, the local model f_u^t is optimized by minimizing the session recommendation loss through session-based interaction learning. The overall procedure is summarized in Alg. 2. Next, we describe the three key steps of the local processing component: LHKG construction, type-aware message passing, and session-based interaction learning.

Local Heterogeneous Knowledge Graph. To capture session dynamics and interaction patterns, we construct a *local heterogeneous knowledge graph* (LHKG) for each client. The LHKG of client u is $\mathcal{G}_u = (\mathcal{V}_u, \mathcal{E}_u, \phi_u, \varphi_u, \omega_u)$, where $\mathcal{V}_u$ and $\mathcal{E}_u$ are the node and edge sets, $\phi : \mathcal{V}_u \to \{U, S, I\}$ maps nodes to types, $\varphi : \mathcal{E}_u \to \{UI, II, SI\}$ maps edges to relation types, and $\omega : \mathcal{E}_u \to \mathbb{R}_+$ assigns

Algorithm 2 Local Processing

Input: Global model f_g^{t-1}, local data $\mathcal{D}_u$ and proxy dataset $\mathcal{Q}$
Output: Long-term preference $\boldsymbol{h}_u^{\text{long}}$, short-term preference $\boldsymbol{h}_u^{\text{short}}$, valuation θ'_u, mean μ_u and variance σ_u of the prediction distribution over $\mathcal{Q}$
1: Receive latest global model f_g^{t-1} and proxy dataset $\mathcal{Q}$ from server
2: Construct personalized LHKG $\mathcal{G}_u$ from $\mathcal{D}_u$
3: Extract long-term preference $\boldsymbol{h}_u^{\text{long}}$ and short-term preference $\boldsymbol{h}_u^{\text{short}}$ via type-aware message passing
4: Train local model f_u^t using session-based interaction learning
5: Compute model predictions $f_u^t(\boldsymbol{s}_r)$ for all $\boldsymbol{s}_r \in \mathcal{Q}$
6: Calculate mean μ_u and variance σ_u of the prediction distribution over $\mathcal{Q}$
7: Upload $(\boldsymbol{h}_u^{\text{long}}, \boldsymbol{h}_u^{\text{short}}, \mu_u, \sigma_u, \theta'_u)$ to server

non-negative edge weights. The node set $\mathcal{V}_u$ contains three disjoint parts: a client node u, a session node $s = \boldsymbol{s}_{u,r_{\max}}$ for the most recent session, and an item node set $\mathcal{I}$. The node-type mapping is defined by $\phi(v) = U$ if $v = u$, $\phi(v) = S$ if $v = s$, and $\phi(v) = I$ if $v \in \mathcal{I}$. The edge-type mapping is defined by $\varphi(e) = UI$ if $e = (u, i)$ or $e = (i, u)$ for some $i \in \mathcal{I}$, $\varphi(e) = II$ if $e = (i, j)$ for some $i, j \in \mathcal{I}$, and $\varphi(e) = SI$ if $e = (s, i)$ or $e = (i, s)$ for some $i \in \mathcal{I}$. We construct $\mathcal{E}_u$ as follows.

- Item–item edges $\mathcal{E}_u^{II}$. For each consecutive item pair (i, j) in the client's sessions, we add a directed edge $(i, j) \in \mathcal{E}_u^{II}$ with weight $\omega(i, j) = \sum_{r=1}^{|\mathcal{D}_u|} \sum_{k=1}^{|\boldsymbol{s}_{u,r}|-1} \mathbf{1}(\boldsymbol{s}_{u,r}[k] = i \wedge \boldsymbol{s}_{u,r}[k+1] = j)$.
- Client–item edges $\mathcal{E}_u^{UI}$. If client u has interacted with item i, we add bidirectional edges $(u, i), (i, u) \in \mathcal{E}_u^{UI}$ with weights $\omega(u, i) = \omega(i, u) = \sum_{r=1}^{|\mathcal{D}_u|} \sum_{k=1}^{|\boldsymbol{s}_{u,r}|} \mathbf{1}(\boldsymbol{s}_{u,r}[k] = i)$.
- Session–item edges $\mathcal{E}_u^{SI}$. If item i appears in the most recent session $\boldsymbol{s}_{u,r_{\max}}$, we add bidirectional edges $(s, i), (i, s) \in \mathcal{E}_u^{SI}$ with weights $\omega(s, i) = \omega(i, s) = \sum_{k=1}^{|\boldsymbol{s}_{u,r_{\max}}|} \mathbf{1}(\boldsymbol{s}_{u,r_{\max}}[k] = i)$.

Type-Aware Message Passing. After constructing the LHKG, each client applies a *heterogeneous graph attention network* (HGAT) to learn personalized node embeddings. A naive aggregation scheme that treats all neighbors uniformly, or separately aggregates different neighbor types without proper reweighting, can be dominated by relation types with more neighbors. This is particularly problematic in the LHKG, where client–item, item–item, and session–item relations carry different semantic meanings. To address this issue, HGAT assigns relation-specific transformation parameters and attention weights, so that each relation type can contribute appropriately to the learned representations.

HGAT performs message passing for L layers. All node embeddings are initialized in dimension d. Specifically, the client node u and the session node s are randomly initialized, while each item node $i \in \mathcal{I}$ is initialized using the item embedding broadcast by the server. For a target node $v \in \mathcal{V}_u$ and one of its neighbor nodes $w \in N(v)$, the relation-specific transformed message at layer $l \leq L$ is computed as $\boldsymbol{m}_{v,w}^{(l)} = \boldsymbol{W}_{\varphi((v,w))}^{(l)} \boldsymbol{e}_w^{(l-1)} + \boldsymbol{b}_{\varphi((v,w))}^{(l)}$, where $\boldsymbol{W}_{\varphi((v,w))}^{(l)} \in \mathbb{R}^{d \times d}$ is

the transformation matrix associated with the edge type $\varphi((v,w))$, $\boldsymbol{e}_w^{(l-1)} \in \mathbb{R}^d$ is the embedding of node w from the previous layer, and $\boldsymbol{b}_{\varphi((v,w))}^{(l)} \in \mathbb{R}^d$ is the corresponding bias term.

To quantify the importance of each neighbor, HGAT further computes a type-aware attention coefficient. For node v and its neighbor w, the attention weight at layer l is defined as

$$a_{v,w}^{(l)} = \frac{\exp\Big(\sigma\Big(\boldsymbol{a}_{\varphi((v,w))}^{(l)\top}\Big[\boldsymbol{e}_v^{(l-1)} \parallel \boldsymbol{e}_w^{(l-1)}\Big]\Big)\Big)}{\sum\limits_{w' \in N(v)} \exp\Big(\sigma\Big(\boldsymbol{a}_{\varphi((v,w'))}^{(l)\top}\Big[\boldsymbol{e}_v^{(l-1)} \parallel \boldsymbol{e}_{w'}^{(l-1)}\Big]\Big)\Big)},$$

where $\boldsymbol{a}_{\varphi((v,w))}^{(l)} \in \mathbb{R}^{2d}$ is the attention vector associated with the relation type, $\sigma(\cdot)$ denotes the sigmoid activation function, $\parallel$ denotes vector concatenation, and $N(v)$ denotes the set of neighboring nodes of v.

Besides relation-aware attention, we also incorporate edge weights to reflect the frequency of interactions encoded in the LHKG. Let $\bar{\omega}_{v,w} = \frac{\omega(w,v)}{\sum_{w' \in N(v)} \omega(w',v)}$ denote the normalized edge weight from w to v. The aggregated neighborhood message for node v at layer l is then given by $\tilde{\boldsymbol{m}}_v^{(l)} = \sum_{w \in N(v)} \bar{\omega}_{v,w}\, a_{v,w}^{(l)}\, \boldsymbol{m}_{v,w}^{(l)}$.

The updated representation of node v is computed by combining its previous embedding with the aggregated neighborhood message:

$$\boldsymbol{e}_v^{(l)} = \mathrm{ReLU}\left(\boldsymbol{W}_{\phi(v)}^{(l)}\left[\boldsymbol{e}_v^{(l-1)} \parallel \tilde{\boldsymbol{m}}_v^{(l)}\right] + \boldsymbol{b}_{\phi(v)}^{(l)}\right),$$

where $\boldsymbol{W}_{\phi(v)}^{(l)} \in \mathbb{R}^{d \times 2d}$ and $\boldsymbol{b}_{\phi(v)}^{(l)} \in \mathbb{R}^d$ are the node-type-specific transformation matrix and bias, respectively.

After L layers, the final client-node embedding $\boldsymbol{e}_u^{(L)}$ is taken as the long-term preference representation, denoted by $\boldsymbol{h}_u^{\text{long}}$, while the final session-node embedding $\boldsymbol{e}_s^{(L)}$ is taken as the short-term preference representation, denoted by $\boldsymbol{h}_u^{\text{short}}$. Intuitively, $\boldsymbol{e}_u^{(L)}$ summarizes the client's overall historical interests through the client–item structure, whereas $\boldsymbol{e}_s^{(L)}$ emphasizes the client's most recent intent through the session–item connections.

Session-based Interaction Learning. To capture sequential patterns within each session, we employ a *gated recurrent unit* (GRU) encoder. For a session $\boldsymbol{s}_{u,r}$ of client u, the GRU reads the interacted items in chronological order, that is, one item at a time from the first interaction to the last. At step $k \in \{1, \ldots, |\boldsymbol{s}_{u,r}|\}$, the input to the GRU is the embedding of the current item $\boldsymbol{s}_{u,r}[k]$, given by $\boldsymbol{x}_k = \boldsymbol{e}_{\boldsymbol{s}_{u,r}[k]} \in \mathbb{R}^d$. Starting from the initial hidden state $\boldsymbol{h}_0 = \boldsymbol{0} \in \mathbb{R}^\nu$, where ν is the hidden dimension, the GRU updates its hidden state sequentially through the standard gating mechanism:

$$\begin{aligned}
\boldsymbol{z}_k &= \sigma(\boldsymbol{W}_z \boldsymbol{x}_k + \boldsymbol{U}_z \boldsymbol{h}_{k-1} + \boldsymbol{b}_z),\\
\boldsymbol{r}_k &= \sigma(\boldsymbol{W}_r \boldsymbol{x}_k + \boldsymbol{U}_r \boldsymbol{h}_{k-1} + \boldsymbol{b}_r),\\
\widetilde{\boldsymbol{h}}_k &= \tanh\big(\boldsymbol{W}_h \boldsymbol{x}_k + \boldsymbol{U}_h(\boldsymbol{r}_k \odot \boldsymbol{h}_{k-1}) + \boldsymbol{b}_h\big),\\
\boldsymbol{h}_k &= (1 - \boldsymbol{z}_k) \odot \widetilde{\boldsymbol{h}}_k + \boldsymbol{z}_k \odot \boldsymbol{h}_{k-1},
\end{aligned}$$

where the matrices $\boldsymbol{W}_z, \boldsymbol{W}_r, \boldsymbol{W}_h \in \mathbb{R}^{\nu \times d}$ map the current item embedding to the hidden space, while $\boldsymbol{U}_z, \boldsymbol{U}_r, \boldsymbol{U}_h \in \mathbb{R}^{\nu \times \nu}$ transform the previous hidden state, the vectors $\boldsymbol{b}_z, \boldsymbol{b}_r, \boldsymbol{b}_h \in \mathbb{R}^{\nu}$ are bias terms, $\sigma(\cdot)$ denotes the sigmoid function and $\odot$ denotes element-wise multiplication. Here, the update gate $\boldsymbol{z}_k$ controls how much past information is retained, while the reset gate $\boldsymbol{r}_k$ determines how much of the previous hidden state is used when forming the candidate state $\widetilde{\boldsymbol{h}}_k$. In this way, after reading the first k items, the hidden state $\boldsymbol{h}_k$ summarizes the sequential information contained in the prefix $(\boldsymbol{s}_{u,r}[1], \ldots, \boldsymbol{s}_{u,r}[k])$.

After the final item is processed, the last hidden state $\boldsymbol{h}_{|\boldsymbol{s}_{u,r}|}$ is taken as the personalized session representation, denoted by $\boldsymbol{h}_{u,r} \in \mathbb{R}^{\nu}$. Since the GRU hidden state lies in $\mathbb{R}^{\nu}$ whereas item embeddings lie in $\mathbb{R}^{d}$, we further project $\boldsymbol{h}_{u,r}$ into the item-embedding space $\tilde{\boldsymbol{h}}_{u,r} = \boldsymbol{W}_{\text{out}} \boldsymbol{h}_{u,r} + \boldsymbol{b}_{\text{out}} \in \mathbb{R}^{d}$, where $\boldsymbol{W}_{\text{out}} \in \mathbb{R}^{d \times \nu}$ and $\boldsymbol{b}_{\text{out}} \in \mathbb{R}^{d}$ are trainable local parameters.

For next-item prediction, we construct a candidate set $\mathcal{C}_{u,r} \subseteq \mathcal{I}$ consisting of the ground-truth next item i^{+} and z randomly sampled negative items. For each candidate item $i \in \mathcal{C}_{u,r}$, its relevance score is computed by matching the session representation with the item embedding $\tilde{y}_i = \sigma(\tilde{\boldsymbol{h}}_{u,r}^{\top} \boldsymbol{e}_i)$, where $\boldsymbol{e}_i$ denotes the embedding of item i. The model is trained by minimizing the binary cross-entropy loss on session: $\mathcal{L}_{u,r} = -\frac{1}{|\mathcal{C}_{u,r}|} \sum_{i \in \mathcal{C}_{u,r}} [y_i \log \tilde{y}_i + (1 - y_i) \log(1 - \tilde{y}_i)]$, where $y_i \in \{0, 1\}$ is the ground-truth label of candidate item i, with $y_i = 1$ for the positive item and $y_i = 0$ for each negative item.

4.2 Client Auction Component

The client auction component addresses the second challenge, namely, how to estimate the value of a client's local update without accessing the raw update itself. Intuitively, a local update is more valuable if it can contribute more effectively to improving the global recommendation model. However, directly inspecting raw client session data is infeasible due to privacy constraints. We therefore quantify client value using non-sensitive information uploaded by clients. Specifically, we propose two complementary metrics: a *novelty score*, which measures how much new information a client contributes, and a *representativeness score*, which reflects how well the client's preferences align with the broader population.

Novelty score. Given the proxy dataset $\mathcal{Q}$, the *novelty score* $\mathsf{score}_u^{\text{nov}}$ measures the divergence between client u's local model and the current global model. Specifically, client u computes model predictions for all sessions $\boldsymbol{s}_r \in \mathcal{Q}$ and then calculates the mean μ_u and variance σ_u of the resulting prediction distribution over $\mathcal{Q}$. Similarly, the server computes the mean μ_g and variance σ_g of the global model's prediction distribution over $\mathcal{Q}$.

For each client $u \in \mathcal{U}$, the server then computes the Kullback–Leibler (KL) divergence between the client's prediction distribution and the global model's prediction distribution: $\mathsf{KL}_u = \mathsf{KL}(\mathcal{N}(\mu_u, \sigma_u) \,\|\, \mathcal{N}(\mu_g, \sigma_g))$, where $\mathcal{N}(\mu, \sigma)$ denotes a normal distribution. Intuitively, KL divergence quantifies how much a client's predictive behaviour deviates from the current global model. Thus, a larger KL value indicates that the client captures patterns less represented by

the global model, and is therefore more novel. The novelty score is then normalized to $[0, 1]$:

$$\mathsf{score}_u^{\mathrm{nov}} = \frac{\mathsf{KL}_u - \min_{v \in \mathcal{U}} \mathsf{KL}_v}{\max_{v \in \mathcal{U}} \mathsf{KL}_v - \min_{v \in \mathcal{U}} \mathsf{KL}_v}.$$

A higher $\mathsf{score}_u^{\mathrm{nov}}$ indicates that client u captures patterns that are less well reflected by the current global model.

Representativeness score. Given the clients' preference representations, the *representativeness score* $\mathsf{score}_u^{\mathrm{rep}}$ measures how well client u represents the overall client population. We quantify this by computing the cosine similarity between client u's preference representation and those of other clients.

To account for both long-term and short-term preference signals, the server first combines the two embeddings into a unified joint preference representation $\boldsymbol{h}_u^{\mathrm{joint}}$, defined as $\boldsymbol{h}_u^{\mathrm{joint}} = \gamma \cdot \boldsymbol{h}_u^{\mathrm{long}} + (1-\gamma) \cdot \boldsymbol{h}_u^{\mathrm{short}}$, where $\gamma \in [0, 1]$ is a server-specified hyperparameter that controls the relative importance of long-term and short-term preferences. For example, $\gamma = 0.7$ emphasizes long-term preference, whereas $\gamma = 0.3$ is more suitable for session-aware recommendation. This weighting scheme flexibly balances stable client preferences and transient session behaviours according to the objective of the recommendation task.

For client u and any other client $v \in \mathcal{U}$, the cosine similarity between their joint preference representations is defined as $\cos_{u,v} = \frac{\boldsymbol{h}_u^{\mathrm{joint}} \cdot \boldsymbol{h}_v^{\mathrm{joint}}}{\|\boldsymbol{h}_u^{\mathrm{joint}}\| \, \|\boldsymbol{h}_v^{\mathrm{joint}}\|}$. The cosine similarity is then normalized as $\mathsf{sim}_{u,v} = \frac{1}{2}\big(\cos_{u,v} + 1\big)$, and the average similarity of client u to all other clients is $\overline{\mathsf{sim}}_u = \frac{1}{n-1} \sum_{v \in \mathcal{U} \setminus \{u\}} \mathsf{sim}_{u,v}$. Finally, the representativeness score is defined as

$$\mathsf{score}_u^{\mathrm{rep}} = \frac{\overline{\mathsf{sim}}_u - \min_{v \in \mathcal{U}} \overline{\mathsf{sim}}_v}{\max_{v \in \mathcal{U}} \overline{\mathsf{sim}}_v - \min_{v \in \mathcal{U}} \overline{\mathsf{sim}}_v}.$$

A higher $\mathsf{score}_u^{\mathrm{rep}}$ indicates that client u's preferences are more representative of the overall population.

Auction and Selection at Server. After computing the novelty and representativeness scores for each client, the server combines them into a unified quality metric and performs client selection through a budget-constrained reverse auction. Since the relative importance of novelty and representativeness may vary across training rounds and budget conditions, we adopt a dynamic weighting scheme, inspired by [10], to balance the two components. Specifically, a dynamic parameter $\alpha \in [0, 1]$ is used to control the relative weights of novelty and representativeness across training rounds:

$$\alpha = \left(1 + \frac{\beta}{n\bar{\theta}}\right)^{-c \cdot t / t_{\max}}, \tag{1}$$

where β is the budget, n is the total number of clients, $\bar{\theta}$ is the mean reported cost of clients, and $c > 0$ is a decay coefficient. Under this design, a larger budget or a later training round results in a smaller α, thereby placing greater emphasis on representativeness to improve population coverage. In contrast, a

Algorithm 3 Client Auction and Participant Selection

Input: client submissions $\{(\boldsymbol{h}_u^{\text{long}}, \boldsymbol{h}_u^{\text{short}}, \mu_u, \sigma_u, \theta'_u)\}_{u\in\mathcal{U}}$, budget β
Output: Selected set $\mathcal{S}^t$, payment set $\{p_u\}_{u\in\mathcal{S}^t}$

for each client $u \in \mathcal{U}$ **do**
  Compute novelty score $\mathsf{score}_u^{\text{nov}}$ and representativeness score $\mathsf{score}_u^{\text{rep}}$
  Compute dynamic balancing parameter α (Eqn. (1))
  Compute comprehensive score $\mathsf{score}_u = \alpha \cdot \mathsf{score}_u^{\text{nov}} + (1-\alpha) \cdot \mathsf{score}_u^{\text{rep}}$
end for
Initialize $\mathcal{S}^t \leftarrow \varnothing$
Sort clients in descending order of cost-effectiveness ratio $\mathsf{score}_u/\theta'_u$
Select clients iteratively under budget and obtain selected set $\mathcal{S}^t$
for each selected client $u \in \mathcal{S}^t$ **do**
  Compute payment p_u using Eq. (2)
end for
Notify selected clients $u \in \mathcal{S}^t$ to conduct transaction and submit local model f_u^t
return Selected set $\mathcal{S}^t$, payment set $\{p_u\}_{u\in\mathcal{S}^t}$

Algorithm 4 Global Aggregation and Payment

Input: Selected clients $\mathcal{S}^t$, updated local models $\{f_u^t\}_{u\in\mathcal{S}^t}$, payments $\{p_u\}_{u\in\mathcal{S}^t}$
Output: Updated global model f_g^t

Update global model: $f_g^t = \frac{1}{|\mathcal{S}^t|}\sum_{u\in\mathcal{S}^t} f_u^t$
for each selected client $u \in \mathcal{S}^t$ **do**
  Transfer payment p_u to u
end for
return f_g^t

tighter budget or an earlier training round yields a larger α, thereby assigning more weight to novelty and encouraging the acquisition of diverse information. The overall score of client u is then computed as

$$\mathsf{score}_u = \alpha \cdot \mathsf{score}_u^{\text{nov}} + (1-\alpha) \cdot \mathsf{score}_u^{\text{rep}}.$$

Clients are then sorted in descending order according to their cost-effectiveness ratio $\mathsf{score}_u/\theta'_u$, which measures the amount of value contributed per unit reported cost. The server selects clients iteratively according to this ordering until the budget is exhausted. For each selected $u \in \mathcal{S}^t$, the payment is defined as

$$p_u = \min\left(\frac{\beta}{\sum_{v\in\mathcal{S}^t}\mathsf{score}_v}\cdot\mathsf{score}_u,\ \frac{\theta'_{u^*}}{\mathsf{score}_{u^*}}\cdot\mathsf{score}_u\right), \tag{2}$$

where u^* denotes the first unselected client in the sorted list. After the auction and selection process, the server sends transaction requests to the selected clients.

4.3 Global Aggregation and Payment Component

After receiving local model updates from all selected clients in round t, the server first performs global aggregation to update the global model $f_g^t = \frac{1}{|\mathcal{S}^t|}\sum_{u\in\mathcal{S}^t} f_u^t$. The overall procedure is summarized in Algorithm 4. During the payment phase, the server transfers the pre-computed payment p_u to each selected client $u \in \mathcal{S}^t$, compensating them for their local computation and communication costs and thereby encouraging sustained participation. Finally, the server broadcasts the updated global model f_g^t to all clients to initiate the next training round.

5 Analysis

This section provides formal theoretical guarantees for the proposed SFRec mechanism, including IC, IR and BF. We further analyze computational complexity and convergence properties of the federated training process.

IC, IR & BF. These properties are established in Lemmas 1–3.

Lemma 1. *The mechanism* SFRec *is incentive compatible.*

Proof. For any client $u \in \mathcal{U}$ with true cost θ_u and reported cost θ'_u. We analyze two cases:

Case 1: Client u is selected under truthful reporting ($u \in \mathcal{S}^t$). If $\theta'_u > \theta_u$, then $\mathsf{score}_u/\theta'_u$ decreases. If client u is dropped, giving $\pi_u(\theta'_u, \boldsymbol{\theta}'_{-u}) = 0 \le \pi_u(\theta_u, \boldsymbol{\theta}'_{-u})$; if u stays selected, the lower bound of payment remains unchanged, so the client's payment does not change and $\pi_u(\theta'_u, \boldsymbol{\theta}'_{-u}) = p_u(\theta'_u) - \theta_u \le p_u(\theta_u) - \theta_u = \pi_u(\theta_u, \boldsymbol{\theta}'_{-u})$. If $\theta'_u < \theta_u$, then $\mathsf{score}_u/\theta'_u$ increases, but payment is still bounded by Eq. 2; hence $\pi_u(\theta'_u, \boldsymbol{\theta}'_{-u}) \le \pi_u(\theta_u, \boldsymbol{\theta}'_{-u})$.

Case 2: Client u is not selected under truthful reporting ($u \notin \mathcal{S}^t$). Under truthful reporting, $\pi_u(\theta_u, \boldsymbol{\theta}'_{-u}) = 0$. If $\theta'_u > \theta_u$, then $\mathsf{score}_u/\theta'_u$ becomes smaller, so u remains unselected and $\pi_u(\theta'_u) = 0$. If $\theta'_u < \theta_u$, u may become selected, but Eq. (2) gives $p_u(\theta'_u) \le \frac{\theta'_{u^*}}{\mathsf{score}_{u^*}}\mathsf{score}_u \le \theta_u$, where u^* is the first unselected client after sorting. Thus, $\pi_u(\theta'_u, \boldsymbol{\theta}'_{-u}) = p_u(\theta'_u) - \theta_u \le 0 = \pi_u(\theta_u, \boldsymbol{\theta}'_{-u})$.

All misreporting strategies fail to increase utility, proving truthful reporting is a dominant strategy. □

Lemma 2. *The mechanism* SFRec *is individually rational.*

Proof. If client u is not selected, then $p_u = 0$ and $\pi_u(\theta'_u, \boldsymbol{\theta}'_{-u}) = 0$. If client u is selected, $\pi_u(\theta'_u, \boldsymbol{\theta}'_{-u}) = \min\{(\beta\mathsf{score}_u)/\sum_{v\in\mathcal{S}^t}\mathsf{score}_v,\ (\theta'_{u^*}\mathsf{score}_u)/\mathsf{score}_{u^*}\} - \theta_u \ge 0$. Therefore, IR holds for all clients.

□

Lemma 3. *The mechanism* SFRec *is budget feasible.*

Proof. For each selected client u, $p_u \le \beta\frac{\mathsf{score}_u}{\sum_{j\in\mathcal{S}^t}\mathsf{score}_j}$. Summing over all selected clients yields $\sum_{u\in\mathcal{S}} p_u \le \beta\sum_{u\in\mathcal{S}^t}\frac{\mathsf{score}_u}{\sum_{v\in\mathsf{score}^t}\mathsf{score}_v} = \beta$. Hence BF is satisfied □

Theorem. 1 follows from Lemmas 1 – 3.

Table 2: Statistics of the three datasets

Dataset	clients	Items	Interactions
MovieLens-100K	943	1,682	100,000
MovieLens-1M	6,040	3,706	1,000,209
Delicious	1,313	5,793	266,190

Theorem 1. *The mechanism* SFRec *is incentive compatible, individually rational and budget feasible.*

Complexity analysis. The overall computational complexity of SFRec comes from three components. For each client u, the local processing cost is $O\big(|\mathcal{G}_u| + |\mathcal{D}_u| \cdot \overline{|\boldsymbol{s}_{u,r}|}\big)$, where $|\mathcal{G}_u|$ is the size of its LHKG, $|\mathcal{D}_u|$ its number of local sessions, and $\overline{|\boldsymbol{s}_{u,r}|}$ the average session length. On the server side, computing the overall score requires $O(n + n^2)$ time, while sorting clients by cost-effectiveness in the auction phase takes $O(n \log n)$. Over $t_{\max}$ rounds, the total complexity is $O\big(t_{\max}\big(n + n^2 + n \log n + \max_{u \in \mathcal{U}}(|\mathcal{G}_u| + |\mathcal{D}_u| \cdot \overline{|\boldsymbol{s}_{u,r}|})\big)\big)$. Under the practical assumption that the local cost is bounded independently of n, the quadratic term n^2 dominates, and the overall complexity reduces to $O(t_{\max} n^2)$.
Convergence Analysis. Under standard smoothness and bounded-variance assumptions, SFRec converges to a stationary point despite non-IID client data and partial participation caused by budget-constrained auction selection. As training progresses, the expected objective decreases and the global model updates stabilize. This shows that SFRec preserves the convergence behaviour of federated optimization.

6 Experiments

In this section, we evaluate the performance of SFRec from five aspects: (**1**) recommendation performance against competitive baselines; (**2**) convergence behaviour of federated training; (**3**) verification of key mechanism properties, including IC, IR, and BF; (**4**) the effect of budget level on recommendation quality; and (**5**) ablation studies on the core design components.

6.1 Experiment Setup

Dataset. We use three real-world datasets: *MovieLens-100K*, containing 100,000 explicit movie ratings from 943 clients on 1,682 movies, with sessions segmented by a 30-minute threshold; *MovieLens-1M*, a larger dataset with over 1 million ratings from 6,040 clients on 3,706 movies, offering richer interaction sequences; and *Delicious*, a social bookmarking dataset with 266,190 interactions from 1,313 clients on 5,793 items, where sessions are divided by time intervals in client activity. Key dataset statistics are reported in Table 2.
Baseline Methods. To evaluate the overall performance of SFRec, we compare it with three baselines. Random selects clients uniformly at random. Greedy [18]

assumes equal data quality and selects clients solely by cost. FedClus [13] clusters clients by features and then selects low-cost clients within each cluster to balance value and diversity.

Evaluation Metrics. To evaluate next-item prediction in session-based recommendation, we use three standard top-10 ranking metrics: HR@10, MRR@10, and NDCG@10, treating the ground-truth next item as the only relevant target. HR@10 measures the proportion of test sessions where the ground-truth item appears in the top-10 list. MRR@10 computes the average reciprocal rank of the ground-truth item within the top-10 list, and is 0 if it does not appear. NDCG@10 further evaluates ranking quality by applying a logarithmic discount to the position of the ground-truth item and normalizing the gain.

Ablation Study. To quantify the contributions of the core components, we design four ablation variants by removing key modules. **w/o nov** removes the novelty score and uses only the representativeness score. **w/o rep** removes the representativeness score and relies only on the novelty score. **w/o $\boldsymbol{h}^{\mathbf{long}}$** uses only the short-term preference representation. **w/o $\boldsymbol{h}^{\mathbf{short}}$** uses only the long-term preference representation. All variants keep the same federated training settings, budget constraints, and incentive mechanism for fair comparison.

Implementation and Hyperparameters. All experiments were implemented in Python 3.8 with PyTorch 1.12, and the code is provided in the supplementary materials. In each global round, half of the clients are randomly activated. Each client's cost is sampled via $\mu_u \sim U[0.8, 1.2]$ and $\theta_u \sim \mathcal{N}(\mu_u, 0.1^2)$, clipped to $[0.5, 1.5]$. For the recommendation model, we set the item embedding dimension to 32 and the GRU hidden dimension to 64. Sessions are constructed with a timeout threshold of 1,800 seconds and a maximum length of 50. Each training sample uses 4 negative items. In federated training, we set the number of global rounds to 30, local epochs to 1, batch size to 256, and learning rate to $\eta = 0.1$. The per-round budget is $\beta = 300.0$, the preference fusion weight is $\gamma = 0.5$, the proxy dataset size is 300, and the decay coefficient in α is $c = 0.8$. The proxy dataset $\mathcal{Q}$ contains 300 anonymized sessions from the public split and is excluded from all clients' local training data.

6.2 Experimental Results

Comparison with Baselines. Table 3 compares the overall performance of SFRec with three baselines on the three datasets. SFRec achieves the best HR@10, NDCG@10, and MRR@10 across all datasets, consistently outperforming the baselines. This is because Random lacks informed client selection, Greedy considers only cost, and FedClus improves diversity through clustering but does not effectively capture update novelty. The results confirm the effectiveness of SFRec in valuing client contributions and selecting participants.

Convergence. Figure 2 shows the convergence behaviour of SFRec on the ML-100K dataset. The left subfigure presents the distribution of novelty scores across training rounds. In Round 1, the scores are relatively dispersed, indicating substantial divergence between local and global models. By Round 15 and Round 30, the distribution shifts leftward and becomes more concentrated, suggesting that

Table 3: Performance Comparison

Method	ML-100K			ML-1M			Delicious		
	HR@10	NDCG@10	MRR@10	HR@10	NDCG@10	MRR@10	HR@10	NDCG@10	MRR@10
Random	57.77	33.31	25.80	52.55	30.76	24.05	38.66	24.34	20.73
Greedy	48.83	28.24	21.84	48.85	28.97	22.86	39.06	18.18	11.74
Fedclus	60.74	37.06	29.73	47.87	28.51	22.55	40.49	27.29	24.22
SFRec	**61.60**	**37.27**	**29.75**	**61.26**	**37.09**	**29.62**	**41.37**	**27.29**	**24.78**

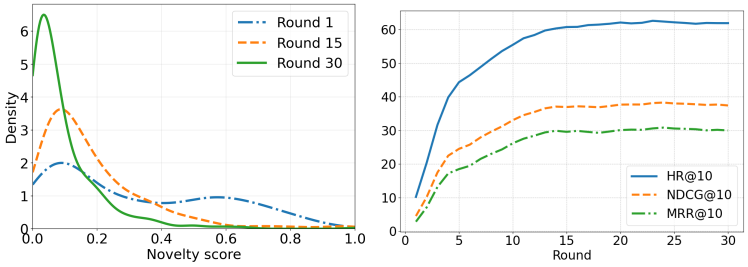

Fig. 2: Convergence performance of SFRec on ML-100k dataset.

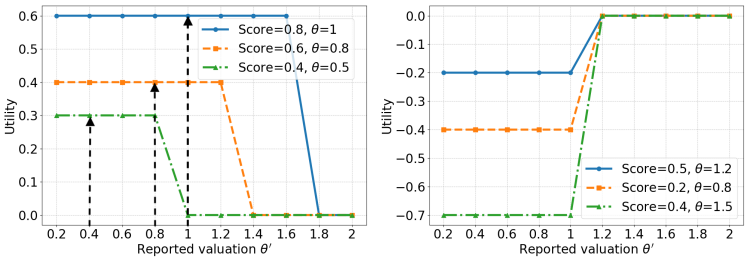

Fig. 3: Utility variation under the SFRec : (Left) Selected client's utility vs reported cost; (Right) Non-selected client's utility vs reported cost.

local models gradually align with the global model and novelty decreases. The right subfigure shows the evolution of the recommendation metrics. HR@10, NDCG@10, and MRR@10 rise quickly in the early rounds and stabilize after about 15–20 rounds. These results indicate that the framework effectively aggregates high-quality local updates while maintaining stable recommendation performance and promoting convergence.

IC and IR. Figure 3 illustrates a client's utility under the mechanism in an assumed population setting. We vary one client's true cost and score while fixing others' reported costs. A critical client u^* has ratio $\mathsf{score}_{u^*}/\theta'_{u^*} = 0.5$. The left subfigure shows selected clients' utility vs. reported cost; the right shows unselected clients'. In both cases, utility peaks at truthful reporting, $\theta' = \theta$, marked by black triangles, and misreporting does not yield higher utility. For example, when $\theta = 1$ and $\mathsf{score} = 0.8$, the utility peaks at $\theta' = 1$ and drops

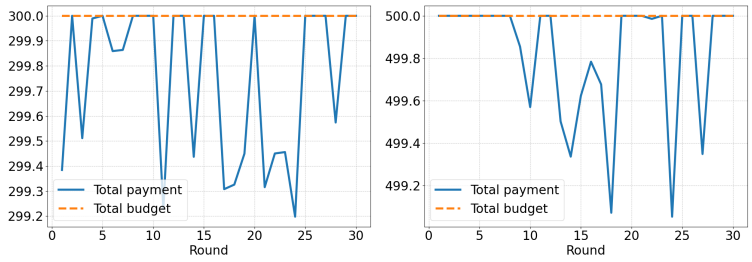

Fig. 4: BF analysis of SFRec on the ML-100K dataset under two budget settings over 30 global training rounds: (left) $\beta = 300$; (right) $\beta = 500$.

Table 4: Performance on Datasets under Different budget β

Dataset	Metric	β=100	β=200	β=300	β=400	β=500
ML-100K	HR@10	49.89	58.30	61.60	61.70	**63.62**
	NDCG@10	29.27	35.38	37.27	39.48	**38.90**
	MRR@10	22.95	28.38	29.75	31.25	**32.54**
ML-1M	HR@10	54.41	58.55	61.26	63.03	**63.78**
	NDCG@10	30.53	33.67	37.09	38.84	**39.38**
	MRR@10	23.19	25.99	29.62	31.34	**31.83**
Delicious	HR@10	30.29	38.11	41.37	43.83	**48.93**
	NDCG@10	13.60	24.95	27.29	28.90	**35.59**
	MRR@10	8.42	20.75	24.78	26.54	**31.36**

Table 5: The impact of each component

Dataset	Metric	w/o nov	w/o rep	w/o h^{long}	w/o h^{short}	SFRec
ML-100K	HR@10	56.28	57.77	56.60	58.53	**61.60**
	NDCG@10	34.50	35.11	34.33	36.94	**37.27**
	MRR@10	27.76	28.14	27.48	29.37	**29.75**
ML-1M	HR@10	55.75	55.48	60.11	58.87	**61.26**
	NDCG@10	34.60	34.62	35.94	35.30	**37.09**
	MRR@10	28.06	28.17	28.51	28.15	**29.62**
Delicious	HR@10	35.66	38.66	39.14	38.27	**41.37**
	NDCG@10	22.38	22.67	20.44	20.12	**27.29**
	MRR@10	18.16	17.57	14.58	14.45	**24.78**

to zero for $\theta' > 1.6$. Unselected clients' utility remains non-positive except at truthful reporting. These results support the IC and IR of the mechanism.

BF. Figure 4 validates the budget feasibility of SFRec on the ML-100K dataset under two budget settings ($\beta = 300$ and $\beta = 500$) over 30 global rounds. In both settings, total payment always remains within the budget, despite minor fluctuations across rounds. This confirms that SFRec effectively controls costs and is economically sustainable for practical deployment.

Impact of Budget. Table 4 reports recommendation performance under different budget levels $\beta \in \{100, 200, 300, 400, 500\}$ on the three datasets. Overall, performance improves as the budget increases. For example, on ML-100K, HR@10 rises from 49.89 at $\beta = 100$ to 63.62 at $\beta = 500$, while on Delicious, MRR@10 increases from 8.42 to 31.36. This is because a larger budget allows the server to select more clients and incorporate more informative local updates. The gains are most noticeable from low to moderate budgets, and become smaller when the budget is already large. These results show that SFRec effectively converts additional budget into better recommendation quality under budget constraints.

Ablation Study. Tab. 5 presents the ablation results for the key components of SFRec on the three datasets. Removing any component lowers performance on all datasets. For example, on ML-100K, HR@10 drops from 61.60 to 56.28 without the novelty score, and to 56.60 without the long-term preference component. These results confirm that the novelty score identifies informative updates, the

representativeness score improves preference coverage, and modeling of long-term preferences and short-term preferences captures both stable and dynamic client intent. All components significantly contribute to SFRec's performance.

7 Conclusion

We studied federated session-based recommendation under strategic client participation and proposed SFRec, a budget-feasible mechanism that combines heterogeneous graph learning with a truthful reverse auction. By modelling long-term and short-term intent locally and evaluating client value through proxy-data signals, SFRec achieves IC, IR, and BF, while delivering consistent improvements on three real-world datasets.

References

1. Cong, M., Yu, H., Weng, X., Qu, J., Liu, Y., Yiu, S.M.: A vcg-based fair incentive mechanism for federated learning. arXiv:2008.06680 (2020)
2. European Parliament, Council of the European Union: Regulation (eu) 2016/679 (general data protection regulation). Off. J. Eur. Union (2016)
3. Hidasi, B., Karatzoglou, A., Baltrunas, L., Tikk, D.: Session-based recommendations with recurrent neural networks. In: ICLR'16 (2016)
4. Jiao, Y., Wang, P., Niyato, D., Lin, B., Kim, D.I.: Toward an automated auction framework for wireless federated learning services market. IEEE Trans. Mob. Comput. **20**(10), 3034–3048 (2020)
5. Li, J., Ren, P., Chen, Z., Ren, Z., Lian, T., Ma, J.: Neural attentive session-based recommendation. In: CIKM'17. pp. 1419–1428 (2017)
6. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. In: MLSys'20 (2020)
7. Li, Y., Gao, C., Luo, H., Jin, D., Li, Y.: Enhancing hypergraph neural networks with intent disentanglement for session-based recommendation. In: SIGIR'22 (2022)
8. Li, Z., Yang, C., Chen, Y., Wang, X., Chen, H., Xu, G., Yao, L., Sheng, Q.Z.: Graph and sequential neural networks in session-based recommendation: A survey. ACM Comput. Surv. **57**(2), Article 40 (2025)
9. Liu, Q., Zeng, Y., Mokhosi, R., Zhang, H.: STAMP: Short-term attention/memory priority model for session-based recommendation. In: KDD'18. pp. 1831–1839 (2018)
10. Liu, Y., Zhang, M., Liu, J., Yang, S.: Data pricing for graph neural networks without pre-purchased inspection. In: AAMAS'25. pp. 1380–1388 (2025)
11. Liu, Z., Yang, L., Fan, Z., Peng, H., Yu, P.S.: Federated social recommendation with graph neural network. ACM Trans. Intell. Syst. Technol. **13**(4), 1–24 (2022)
12. Lou, J., Rong, C., Cheng, Y., Feng, W., Liang, X.: Efficient federated graph aggregation for privacy-preserving session-based recommendation using graph neural networks. Sci. Rep. (2025)
13. Lu, R., Zhang, W., Wang, Y., Li, Q., Zhong, X., Yang, H., Wang, D.: Auction-based cluster federated learning in mobile edge computing systems. IEEE Trans. Parallel Distrib. Syst. **34**(4), 1145–1158 (2023)

14. McMahan, H.B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: AISTATS'17 (2017)
15. Myerson, R.B.: Optimal auction design. Math. Oper. Res. **6**(1), 58–73 (1981)
16. Nguyen, T.L., Hoang, D.T., Nguyen, D.N., Pham, Q.: Right reward right time for federated learning. arXiv:2503.07869 (2025)
17. Ou, Z., Zhang, X., Zhu, Y., Lyu, S., Liu, J., Ao, T.: LS-TGNN: Long and short-term temporal graph neural network for session-based recommendation. In: AAAI'25. pp. 12426–12434 (2025)
18. Singer, Y.: Budget feasible mechanisms. In: FOCS'10. pp. 765–774 (2010)
19. Sun, Z., Xu, Y., Liu, Y., He, W., Kong, L., Wu, F., Jiang, Y., Cui, L.: A survey on federated recommendation systems. IEEE Trans. Neural Netw. Learn. Syst. **36**(1), 6–20 (2024)
20. Wu, C., Wu, F., Lyu, L., Qi, T., Huang, Y., Xie, X.: A federated graph neural network framework for privacy-preserving personalization. Nat. Commun. **13**(1), 3091 (2022)
21. Wu, S., Tang, Y., Zhu, Y., Wang, L., Xie, X., Tan, T.: Session-based recommendation with graph neural networks. In: AAAI'19. pp. 346–353 (2019)
22. Xia, X., Yin, H., Yu, J., Wang, Q., Cui, L., Zhang, X.: Self-supervised hypergraph convolutional networks for session-based recommendation. In: AAAI'21. pp. 4503–4511 (2021)
23. Xu, C., Zhao, P., Liu, Y., Sheng, V.S., Xu, J., Zhuang, F., Fang, J., Zhou, X.: Graph contextualized self-attention network for session-based recommendation. In: IJCAI'19. pp. 3940–3946 (2019)
24. Ye, W., An, X., Wang, J., Yan, X., Carle, G.: FedABC: Attention-based client selection for federated learning with long-term view. arXiv:2507.20871 (2025)
25. Zhang, H., Tang, Y., Liu, J., Chen, W.: SFedRec: A federated learning framework for dynamic session-based recommendation. In: AAMAS'25 (Extended Abstracts). pp. 2825–2828 (2025)
26. Zhang, J., Wu, Y., Pan, R.: Incentive mechanism for horizontal federated learning based on reputation and reverse auction. In: WWW'21. pp. 947–956 (2021)
27. Zhang, M., Beltrán, F., Liu, J.: A survey of data pricing for data marketplaces. IEEE Transactions on Big Data **9**(4), 1038–1056 (2023)