FEDGRES: FEDERATED GENERATIVE RECOMMENDATION VIA GLOBAL SEMANTIC-ID RELATION SHARING

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative recommendation predicts next items by autoregressively generating their hierarchical Semantic IDs (SIDs). We study this paradigm in cross-silo federated learning, where user-item interaction sequences remain local to each client. Learning sequence patterns for next-item prediction is challenging when exclusive-user preferences vary across clients and shared users' interaction histories are fragmented across them. To address these challenges, we propose `FedGReS`, which complements parameter aggregation with first-level SID graphs built from pooled transition and co-occurrence counts. Distinct item pairs mapped to the same pair of first-level SID codewords contribute to shared counts, making cross-client relational evidence available to local sequence learning. The resulting graphs condition first-level SID representations of historical items through graph propagation and gated residual injection. The graph-conditioning mechanism and the generative model are jointly trained under the full-SID generation objective. `FedGReS` exchanges neither user identifiers nor interaction sequences. Across three Amazon Reviews 2023 datasets, `FedGReS` achieves the highest mean performance in all 12 dataset-metric settings, with relative gains of up to 6.98% over the strongest baseline.

1 INTRODUCTION

Generative recommendation formulates next-item prediction as autoregressive generation over discrete item tokens. Under hierarchical quantization, each item is assigned a Semantic ID (SID) comprising a sequence of codewords, with earlier codewords capturing coarse semantic similarity and later codewords encoding progressively finer distinctions. TIGER (Rajput et al., 2023) pioneered this direction using RQ-VAE tokenization (Lee et al., 2022) and Transformer generation. Subsequent methods refined SID construction and incorporated collaborative signals (Zheng et al., 2024; Wang et al., 2024c; Zhu et al., 2024; Qu et al., 2025). These methods nevertheless assume centralized access to user interaction histories.

In cross-silo recommendation, data-localization constraints often prevent institutional clients from pooling their interaction logs (Li et al., 2025). As illustrated in Figure 1, clients may have partially overlapping user populations: *client-exclusive users* (hereafter *exclusive users*) interact with only one client, whereas *shared users* interact with multiple clients and have histories distributed across their local logs. Federated learning (FL) enables these clients to train a shared model without pooling their users' interaction logs. Given a shared item-SID mapping that aligns the SID vocabulary across clients, a natural baseline is to train the SID-based Transformer with FedAvg (McMahan et al., 2017).

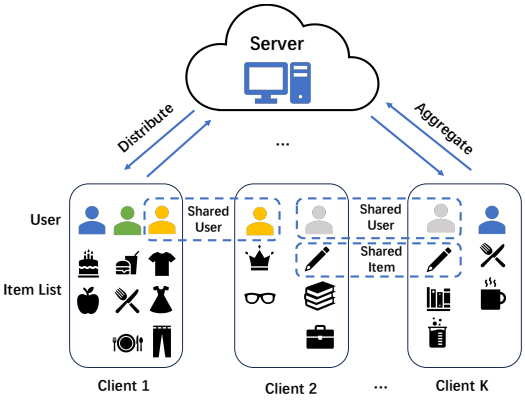


Figure 1: Cross-silo federated generative recommendation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

This setting poses two challenges for learning sequence patterns that support next-item prediction. **First**, exclusive users can have different preferences across clients, yielding heterogeneous local sequence patterns. **Second**, shared users' histories are fragmented across clients, limiting the sequential context available for local learning. FedAvg transfers sequence-derived information through parameter aggregation, but does not explicitly pool observed transitions and contextual associations across clients. We therefore seek aligned sharing units that allow observations from different clients and item pairs to supplement local sequence learning without cross-client user matching.

We propose **`FedGReS`** (**Fed**erated **G**enerative **Re**commendation with Semantic-ID Relation **S**haring) to complement local sequence learning with cross-client relational information pooled over first-level SID codewords. With a fixed shared item-SID mapping, first-level codewords define aligned coarse groups from item-content embeddings for cross-client relation pooling, while complete SIDs retain item-level distinctions. Counting relations between first-level SID codewords allows different item pairs mapped to the same codeword pair to contribute to a common transition or co-occurrence count. The server pools local transition and co-occurrence counts once into fixed sparse graphs. During training, graph propagation aggregates codeword embeddings over these graphs, and gated residual injection incorporates the resulting context into first-level SID representations. By exposing each client to codeword relations absent from its local data, the pooled context can support learning under heterogeneous exclusive-user preferences and fragmented shared-user histories. Count pooling requires neither user matching nor raw-sequence exchange and is compatible with secure aggregation (Bonawitz et al., 2017).

The main contributions of this paper are summarized as follows:

- We formulate SID-based federated generative recommendation for cross-silo settings with both exclusive and shared users. We identify two challenges for learning sequence patterns: heterogeneous preferences across clients' exclusive users and fragmented interaction histories of shared users.
- We propose `FedGReS`, which complements federated parameter aggregation with pooled transition and co-occurrence graphs over aligned first-level SID codewords. Its graph-conditioning mechanism learns to incorporate their context into first-level history representations under the full-SID generation objective, without cross-client user matching or raw-sequence exchange.
- We evaluate `FedGReS` on three real-world datasets from Amazon Reviews 2023 under varying levels of data heterogeneity and federation size. `FedGReS` achieves the highest mean performance in all 12 dataset-metric settings, with relative gains of up to 6.98% over the strongest baseline in each setting, and remains the strongest method across all evaluated federated configurations.

2 Related Work

Federated Recommendation. A large body of federated recommendation work extends collaborative filtering to decentralized user-item interactions while keeping raw data local (Ammad-Ud-Din et al., 2019; Chai et al., 2021; Zhang et al., 2023; Li et al., 2024; 2025). More recent work models ordered interaction histories to capture sequential dependencies (Luo et al., 2022; Yuan et al., 2025a; Imran et al., 2023; Zhang et al., 2024) or uses large language models to enrich preference modeling with semantic knowledge (Yuan et al., 2025b; Zeng et al., 2024; Zhao et al., 2025). These approaches, however, do not autoregressively generate hierarchical quantized SIDs. Concurrent work FedCGR (Liu et al., 2026) also studies SID-based federated generative recommendation, personalizing parameter aggregation based on similarities between domain prototypes while keeping collaborative item representations local. In contrast, `FedGReS` explicitly pools behavioral relation counts over first-level SIDs, complementing parameter transfer with shared relational context for learning from heterogeneous and fragmented histories.

SID-based generative recommendation. TIGER (Rajput et al., 2023) combines RQ-VAE item tokenization (Lee et al., 2022) with autoregressive Transformer generation of hierarchical SIDs. Subsequent work improves SID construction through contrastive quantization and learnable tokenization (Zhu et al., 2024; Qu et al., 2025; Wang et al., 2024a), and incorporates collaborative signals through multi-stream architectures or unified semantic and collaborative codes (Zheng et al.,

2024; Wang et al., 2024b; Xiao et al., 2025). Other methods extend generative recommendation toward unified retrieval and ranking (Yang et al., 2025; Deng et al., 2025). The SID-based methods reviewed in this paragraph are developed for centralized training and do not address client-local interaction histories in cross-silo FL.

3 Problem Formulation

Semantic-ID construction. Let $\mathcal{I}$ denote the item catalog. Each item $i \in \mathcal{I}$ has textual attributes $\mathbf{a}_i$ (e.g., title and category), which a pre-trained content encoder g_ϕ maps to a semantic embedding $\mathbf{x}_i = g_\phi(\mathbf{a}_i) \in \mathbb{R}^d$. For a user u, let $\mathcal{S}_u = (i_{u,1}, \ldots, i_{u,n_u})$ be the chronological interaction history and $y_u = i_{u,n_u+1}$ the next item to predict.

Let M denote the number of semantic quantization levels. At each level m, let $\mathcal{C}_m = \{\boldsymbol{\mu}_{m,1}, \ldots, \boldsymbol{\mu}_{m,K}\}$ denote a codebook of K representative vectors. A residual quantizer maps the content embedding $\mathbf{x}_i$ to an M-token semantic code. Starting from $\mathbf{r}_{i,1} = \mathbf{x}_i$, it performs the following codeword selection and residual update for each $m = 1, \ldots, M$:

$$s_{i,m} = \underset{j \in \{1,\ldots,K\}}{\arg\min} \|\mathbf{r}_{i,m} - \boldsymbol{\mu}_{m,j}\|_2^2, \quad \mathbf{r}_{i,m+1} = \mathbf{r}_{i,m} - \boldsymbol{\mu}_{m,s_{i,m}}. \tag{1}$$

Here, $s_{i,m}$ is the index of the codeword selected for item i at level m. This tokenization can be instantiated by RQ-VAE (Lee et al., 2022) or Residual K-Means (Deng et al., 2025). The semantic code $(s_{i,1}, \ldots, s_{i,M})$ represents item i at successively finer semantic levels. Different items may nevertheless share the same semantic code. Following Rajput et al. (2023), we append a collision-disambiguation token $s_{i,L}$ at position $L = M + 1$ and define the complete Semantic ID (SID) as

$$\mathbf{s}_i = \tau(i) = \big(s_{i,1}, \ldots, s_{i,M}, s_{i,L}\big), \qquad |\mathbf{s}_i| = L, \tag{2}$$

where $s_{i,\ell}$ denotes the ℓ-th SID token. The final token $s_{i,L} \in \{1, \ldots, K\}$ uniquely indexes item i among items sharing the same semantic code, assuming that no semantic code is assigned to more than K items. Consequently, τ is injective. Thus, every valid SID maps to exactly one item, while only the first M SID tokens encode semantic information.

Autoregressive recommendation. Replacing each historical item with its SID yields $\mathbf{X}_u = \tau(i_{u,1}) \oplus \cdots \oplus \tau(i_{u,n_u})$, a token sequence of length $n_u L$, where $\oplus$ denotes concatenation. A Transformer f_θ generates the target SID autoregressively:

$$p_\theta(\tau(y_u) \mid \mathbf{X}_u) = \prod_{\ell=1}^{L} p_\theta(s_{y_u,\ell} \mid \mathbf{X}_u, \mathbf{s}_{y_u,<\ell}). \tag{3}$$

Given history-target pairs $\mathcal{D} = \{(\mathcal{S}_r, y_r)\}_{r=1}^{|\mathcal{D}|}$, the teacher-forced training objective is

$$\mathcal{L}_{\mathrm{rec}}(\theta; \mathcal{D}) = -\frac{1}{|\mathcal{D}|} \sum_{(\mathcal{S},y)\in\mathcal{D}} \sum_{\ell=1}^{L} \log p_\theta(s_{y,\ell} \mid \mathbf{X}(\mathcal{S}), \mathbf{s}_{y,<\ell}). \tag{4}$$

Federated Generative Recommendation. We consider a cross-silo federation (Li et al., 2025) with N clients $\mathcal{P} = \{P_1, \ldots, P_N\}$ and a central server. Client P_k holds a local item catalog $\mathcal{I}_k \subseteq \mathcal{I}$, a local user set $\mathcal{U}_k$, and a private training set

$$\mathcal{D}_k = \{(\mathcal{S}_r^{(k)}, y_r^{(k)})\}_{r=1}^{|\mathcal{D}_k|}, \qquad y_r^{(k)} \in \mathcal{I}_k, \tag{5}$$

where each pair is derived from a local user's chronological interactions. Client catalogs and user populations may partially overlap: $\mathcal{I}_k \cap \mathcal{I}_j \neq \varnothing$ and $\mathcal{U}_k \cap \mathcal{U}_j \neq \varnothing$ may hold for $k \neq j$.

Each client minimizes $\mathcal{L}_k(\theta) = \mathcal{L}_{\mathrm{rec}}(\theta; \mathcal{D}_k)$, and the round-$t$ weighted objective is

$$\mathcal{L}^t(\theta) = \sum_{k=1}^{N} p_k^t \mathcal{L}_k(\theta), \qquad p_k^t = m_k^t / \sum_{j=1}^{N} m_j^t, \tag{6}$$

Here m_k^t counts training examples actually processed in round t, including repeated draws after augmentation. FedAvg (McMahan et al., 2017) alternates local updates and server-side parameter averaging with these round-specific weights.

4 THE PROPOSED FEDGRES FRAMEWORK

4.1 OVERVIEW AND DESIGN RATIONALE

SID-based generative recommenders learn sequence dependencies through autoregressive training. In cross-silo FL, heterogeneous preferences and fragmented histories give clients different and incomplete relation observations. FedGReS complements parameter aggregation with a shared relation context over first-level SID codewords. We choose this level because it quantizes item-content embeddings into coarse semantic groups. The fixed item–SID mapping τ aligns these units across clients. In particular, distinct item pairs (i, j) and (i', j') observed by different clients contribute to the same pooled relation count whenever $s_{i,1} = s_{i',1}$ and $s_{j,1} = s_{j',1}$. This grouping pools observations beyond exact item-pair matches while limiting each relation matrix to $K \times K$ entries. Sharing relations over these aligned groups requires no user matching, while complete SIDs retain item-level distinctions.

Figure 2 summarizes the two-phase framework of FedGReS. Before federated training, clients count first-level SID transitions and co-occurrences from local training sequences, and the server pools these counts once into two fixed sparse graphs. Their retained edges define shared relation neighborhoods, while edge weights guide aggregation of current codeword embeddings. Two graph attention networks (GATs) (Veličković et al., 2018) encode successor and co-occurrence context for gated residual injection into encoder-side first-level SID representations. The full-SID generation objective jointly trains this conditioning mechanism and the generative model through local optimization and FedAvg. Appendix A gives the optimization protocol and pseudocode.

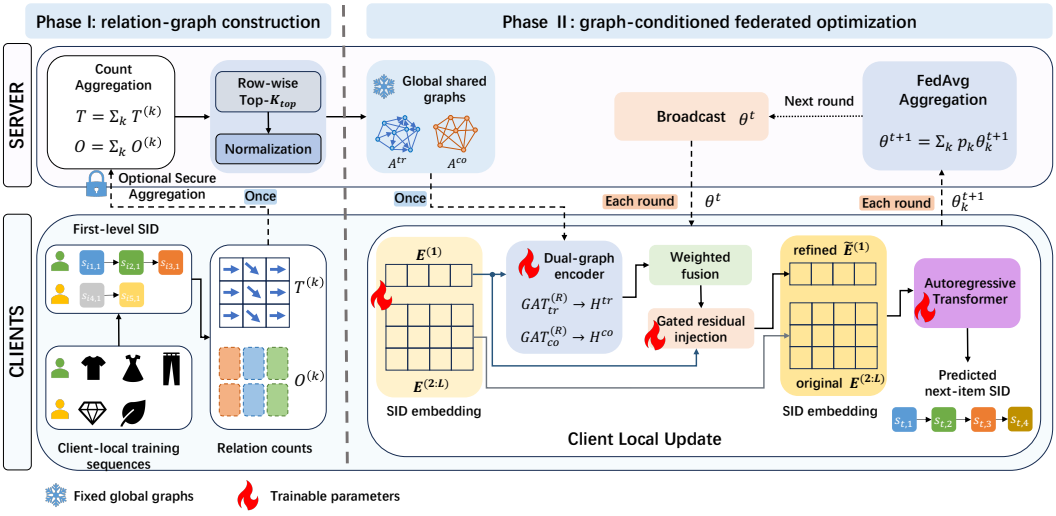


Figure 2: Overview of FedGReS. **Phase I:** Client-local first-level SID transition and co-occurrence counts are pooled once into fixed graphs. **Phase II:** Graph-conditioned first-level history representations are learned jointly with the generative model under FedAvg.

4.2 FIRST-LEVEL SID RELATION GRAPH CONSTRUCTION

Client-local relation extraction. For each user $u \in \mathcal{U}_k$, let $\mathcal{S}_u^{(k)} = (i_{u,1}^{(k)}, \ldots, i_{u,n_u^{(k)}}^{(k)})$ denote the user's chronologically ordered item sequence in client P_k's local training split. All relation statistics are computed from these client-local training sequences without cross-client user matching. Mapping each item to its first-level SID codeword, $z_{u,t}^{(k)} = s_{i_{u,t}^{(k)},1}$, yields a client-local sequence of first-level codewords. We compute two complementary relation statistics over pairs of first-level SID codewords $a, b \in \{1, \ldots, K\}$.

Definition 1 (Directed lag-one transition matrix)**.** The directed lag-one transition matrix $T^{(k)} \in \mathbb{Z}_{\geq 0}^{K \times K}$ counts client-local transitions, with each entry $T_{ab}^{(k)}$ indicating how often codeword b imme-

diately follows codeword a.

$$T_{ab}^{(k)} = \sum_{u \in \mathcal{U}_k} \sum_{t=1}^{n_u^{(k)}-1} \mathbf{1}\Big(z_{u,t}^{(k)} = a,\, z_{u,t+1}^{(k)} = b\Big). \tag{7}$$

Definition 2 (Symmetric windowed co-occurrence matrix)**.** For a window length of $W \geq 2$ items, the symmetric co-occurrence matrix $O^{(k)} \in \mathbb{Z}_{\geq 0}^{K \times K}$ has entries defined as follows.

$$O_{ab}^{(k)} = \sum_{u \in \mathcal{U}_k} \sum_{w=1}^{W-1} \sum_{t=1}^{n_u^{(k)}-w} \mathbf{1}\Big((z_{u,t}^{(k)}, z_{u,t+w}^{(k)}) \in \{(a,b),(b,a)\}\Big). \tag{8}$$

Thus, $T^{(k)}$ captures directed lag-one transitions, whereas $O^{(k)}$ captures order-invariant co-occurrences within windows of W items. Both are aggregate first-level codeword-pair counts computed from client-local training sequences.

Cross-silo aggregation and graph construction. Because the client-local count matrices use the same first-level codeword indexing, the server sums them entrywise across clients as

$$T = \sum_{k=1}^{N} T^{(k)}, \qquad O = \sum_{k=1}^{N} O^{(k)}. \tag{9}$$

For each source codeword, we sparsify both matrices by retaining up to the K_{top} largest positive row entries. Specifically, for $K_{\text{top}} \in \{1, \ldots, K\}$, let $\mathcal{N}_a^{\text{tr}}$ and $\mathcal{N}_a^{\text{co}}$ denote the column indices of these retained entries in row a of T and O, respectively. We then prune both matrices row-wise:

$$\tilde{T}_{ab} = T_{ab}\mathbf{1}\{b \in \mathcal{N}_a^{\text{tr}}\}, \qquad \tilde{O}_{ab} = O_{ab}\mathbf{1}\{b \in \mathcal{N}_a^{\text{co}}\}. \tag{10}$$

Ties are resolved by codeword index, and each row retains at most K_{top} positive relations.

Let $d_a^{\text{tr}} = \sum_q \tilde{T}_{aq}$ and $d_a^{\text{co}} = \sum_q \tilde{O}_{aq}$. For each positive-degree row, we divide the retained entries by the corresponding row sum:

$$A_{ab}^{\text{tr}} = \frac{\tilde{T}_{ab}}{d_a^{\text{tr}}}, \qquad A_{ab}^{\text{co}} = \frac{\tilde{O}_{ab}}{d_a^{\text{co}}}. \tag{11}$$

We set every entry in a zero-degree row to zero. Each positive-degree row is row-stochastic over retained successors in A^{tr} or co-occurrence neighbors in A^{co}, preserving their relative count magnitudes. The server distributes $A^{\text{tr}}, A^{\text{co}} \in \mathbb{R}^{K \times K}$ as fixed graph inputs to all clients. Although O is symmetric, row-wise pruning and normalization can make A^{co} asymmetric.

Together, the transition and co-occurrence graphs provide all clients with complementary sequential and order-invariant relational context at the first-level SID granularity.

Privacy and communication scope. The entrywise sums in Eq. (9) can be computed using secure aggregation (Bonawitz et al., 2017). In this case, the server obtains only the pooled matrices T and O, rather than the individual matrices $T^{(k)}$ and $O^{(k)}$. Otherwise, the server receives these client-level matrices. In either case, raw interaction sequences and user identities remain local. The server constructs the two relation graphs once before federated training. Each client uploads $2K^2$ scalar counts, and the server broadcasts both graphs once, in addition to per-round FedAvg traffic.

4.3 Dual-Graph-Conditioned SID Representation Learning

Let d_e denote the Transformer token-embedding dimension. The model uses a level-aware embedding table $\mathbf{E} \in \mathbb{R}^{(LK) \times d_e}$, with a separate block $\mathbf{E}^{(\ell)} \in \mathbb{R}^{K \times d_e}$ for each SID position. Token $s_{i,\ell}$ is represented by $\mathbf{E}_{s_{i,\ell},:}^{(\ell)}$. All blocks are trained and aggregated through FedAvg, while graph conditioning applies only to $\mathbf{E}^{(1)}$ at the history encoder input.

Dual-graph attention encoding. Pooled relation graphs expose SID neighborhoods that may be absent from a client's local observations. We use two independently parameterized R-layer GATs (Veličković et al., 2018) to turn these neighborhoods into representations for generation. For

$r \in \{\mathrm{tr}, \mathrm{co}\}$, let $\mathcal{N}_a^r = \{b : A_{ab}^r > 0\}$ denote the retained neighbors of codeword a. For one layer and attention head, let $\mathbf{h}_a^r$ denote the current node representation and e_{ab}^r the attention score computed from the representations of a and b. For $b \in \mathcal{N}_a^r$, the relation-weighted attention and aggregated message are

$$\beta_{ab}^r = \frac{A_{ab}^r \exp(e_{ab}^r)}{\sum_{q \in \mathcal{N}_a^r} A_{aq}^r \exp(e_{aq}^r)}, \qquad \mathbf{m}_a^r = \sum_{b \in \mathcal{N}_a^r} \beta_{ab}^r \mathbf{W}^r \mathbf{h}_b^r. \tag{12}$$

Here, $\mathbf{W}^r$ is the head's trainable feature projection, and layer and head indices are omitted for clarity. Thus, A^r defines the available neighbors and preserves their relative pooled relation strengths, while attention learns how to weight their transformed representations. The transition branch aggregates successor representations, whereas the co-occurrence branch aggregates contextual neighbor representations. Stacking R layers combines information along retained relation paths. Concatenated head outputs yield d_e-dimensional codeword representations. After R layers,

$$\mathbf{H}^{\mathrm{tr}} = \mathrm{GAT}_{\mathrm{tr}}^{(R)}(\mathbf{E}^{(1)}; A^{\mathrm{tr}}), \qquad \mathbf{H}^{\mathrm{co}} = \mathrm{GAT}_{\mathrm{co}}^{(R)}(\mathbf{E}^{(1)}; A^{\mathrm{co}}), \tag{13}$$

where $\mathbf{H}^{\mathrm{tr}}, \mathbf{H}^{\mathrm{co}} \in \mathbb{R}^{K \times d_e}$. Each row is the representation of one first-level codeword conditioned on the corresponding relation type. We combine the two branches using a mixing coefficient $\alpha \in [0, 1]$, which remains fixed during each run:

$$\mathbf{H}^{\mathrm{graph}} = \alpha \mathbf{H}^{\mathrm{co}} + (1 - \alpha)\mathbf{H}^{\mathrm{tr}}. \tag{14}$$

Although the graph inputs remain fixed, each client recomputes $\mathbf{H}^{\mathrm{graph}} \in \mathbb{R}^{K \times d_e}$ at every local optimization step using its current first-level embedding block and GAT parameters.

Gated residual injection. A codeword- and dimension-wise gate controls the injection of graph-conditioned information into the original first-level embedding block:

$$\mathbf{G}^{(1)} = \sigma\Big([\mathbf{E}^{(1)} \,\|\, \mathbf{H}^{\mathrm{graph}}]\mathbf{W}_g^\top + \mathbf{1}_K \mathbf{b}_g^\top\Big), \tag{15}$$

$$\tilde{\mathbf{E}}^{(1)} = \mathbf{E}^{(1)} + \mathbf{G}^{(1)} \odot \big(\mathbf{H}^{\mathrm{graph}} \mathbf{U}^\top\big), \tag{16}$$

where $[\cdot\|\cdot]$ concatenates row features, $\mathbf{1}_K$ is the all-ones column vector, σ is the sigmoid function, and $\odot$ denotes element-wise multiplication. The gate entries lie in $(0, 1)$. The trainable parameters have dimensions $\mathbf{W}_g \in \mathbb{R}^{d_e \times 2d_e}$, $\mathbf{b}_g \in \mathbb{R}^{d_e}$, and $\mathbf{U} \in \mathbb{R}^{d_e \times d_e}$, where $\mathbf{U}$ parameterizes a bias-free projection. Thus, both $\mathbf{G}^{(1)}$ and $\tilde{\mathbf{E}}^{(1)}$ lie in $\mathbb{R}^{K \times d_e}$. The residual connection retains a direct path from the original codeword representation. At the Transformer encoder input, $\tilde{\mathbf{E}}^{(1)}$ replaces $\mathbf{E}^{(1)}$ for first-level tokens in the historical SID sequence. Gradients from complete target-SID generation jointly update the codeword embeddings, both GAT branches, and the injection parameters, training how pooled relational context contributes to history encoding.

5 Experiments

5.1 Experimental Setup

Datasets and Evaluation Metrics. We evaluate `FedGReS` on Baby Products (**Baby**), **Software**, and Video Games (**Video**) from Amazon Reviews 2023 (Hou et al., 2024). We apply leave-one-out evaluation separately to each retained client-local sequence. We report micro-averaged Recall@K and NDCG@K for $K \in \{5, 10\}$, averaged over three runs, and abbreviate them as R@K and N@K in tables.

Baselines. We compare `FedGReS` with three baseline families: federated collaborative-filtering methods (PFedRec (Zhang et al., 2023) and FedRAP (Li et al., 2024)), federated sequential recommenders (GRU4Rec+FedAvg (Hidasi et al., 2016; McMahan et al., 2017), SASRec+FedAvg (Kang & McAuley, 2018; McMahan et al., 2017), and FedDCSR (Zhang et al., 2024)), and the TIGER-style generative backbone trained with FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020), or FedAdam (Reddi et al., 2021).

Implementation. All SID-based generative methods use the same TIGER-style architecture and item-SID mapping. Unless otherwise specified, we use $N = 3$ fully participating clients, an exclusive-user ratio of $\rho_{\mathrm{ex}} = 0.3$, and Dirichlet concentration $\gamma = 10$ for shared-user allocation. Appendix B provides dataset statistics, partitioning and evaluation procedures, baseline descriptions, and implementation details. Section 5.4 discusses hyperparameter selection.

5.2 PERFORMANCE OVERVIEW

Table 1 compares FedGReS with eight federated baselines on three datasets. FedGReS achieves the highest mean performance in all 12 dataset-metric settings, with the largest relative gain over the strongest baseline reaching 6.98% on Baby Recall@5 (0.0184 versus 0.0172). Its gains over FedAvg, FedProx, and FedAdam under the same TIGER-style backbone and item-SID mapping are consistent with the benefit of complementing parameter aggregation with explicit relation sharing.

Table 1: Mean recommendation performance over three runs. R/N denote Recall/NDCG. Bold marks the best result per column. Standard deviations for FedGReS appear in Appendix B.5.

Family	Method	Video				Software				Baby			
		R@5	N@5	R@10	N@10	R@5	N@5	R@10	N@10	R@5	N@5	R@10	N@10
CF	PFedRec	0.0276	0.0168	0.0476	0.0233	0.0715	0.0453	0.1095	0.0585	0.0155	0.0094	0.0259	0.0127
	FedRAP	0.0285	0.0205	0.0415	0.0246	0.0715	0.0466	0.1096	0.0586	0.0154	0.0099	0.0253	0.0130
Sequential	GRU4Rec+FedAvg	0.0323	0.0201	0.0538	0.0270	0.0679	0.0443	0.1100	0.0578	0.0161	0.0099	0.0259	0.0134
	SASRec+FedAvg	0.0327	0.0186	0.0554	0.0256	0.0675	0.0444	0.1067	0.0573	0.0170	0.0098	0.0260	0.0130
	FedDCSR	0.0313	0.0203	0.0498	0.0262	0.0702	0.0455	0.1063	0.0579	0.0157	0.0101	0.0260	0.0137
Generative	FedAvg	0.0330	0.0209	0.0530	0.0274	0.0698	0.0451	0.1089	0.0578	0.0172	0.0118	0.0261	0.0145
	FedProx	0.0350	0.0224	0.0540	0.0285	0.0717	0.0462	0.1097	0.0584	0.0171	0.0112	0.0251	0.0138
	FedAdam	0.0332	0.0207	0.0526	0.0270	0.0671	0.0425	0.1044	0.0546	0.0161	0.0104	0.0226	0.0125
	FedGReS	**0.0355**	**0.0226**	**0.0566**	**0.0294**	**0.0745**	**0.0483**	**0.1134**	**0.0609**	**0.0184**	**0.0121**	**0.0267**	**0.0148**

5.3 ABLATION STUDY

We assess component contributions through removal ablations and examine SID relation sharing and graph conditioning through design replacements.

Table 2: Ablation results averaged over three runs. **Full** denotes FedGReS. **w/o** and **w/** indicate component removal and design replacement, respectively. R/N denote Recall/NDCG. Bold marks the best result per column.

Family	Variant	Video				Software				Baby			
		R@5	N@5	R@10	N@10	R@5	N@5	R@10	N@10	R@5	N@5	R@10	N@10
Component removals	w/o Agg	0.0347	0.0225	0.0545	0.0288	0.0741	0.0480	0.1131	0.0602	0.0174	0.0114	0.0258	0.0142
	w/o Trans	0.0350	0.0221	0.0560	0.0291	0.0731	0.0478	0.1133	0.0608	0.0177	0.0118	0.0261	0.0145
	w/o CoOccur	0.0341	0.0216	0.0550	0.0283	0.0733	0.0476	0.1130	0.0607	0.0172	0.0113	0.0256	0.0141
	w/o Gate	0.0346	0.0215	0.0549	0.0281	0.0743	0.0482	0.1128	0.0606	0.0169	0.0111	0.0249	0.0137
Design replacements	w/ Rand-Rel	0.0328	0.0211	0.0533	0.0272	0.0726	0.0465	0.1108	0.0588	0.0163	0.0103	0.0248	0.0131
	w/ L2-Rel	0.0348	0.0221	0.0538	0.0282	0.0697	0.0438	0.1091	0.0565	0.0180	0.0115	0.0252	0.0139
	w/ Linear	0.0340	0.0216	0.0561	0.0288	0.0709	0.0454	0.1101	0.0580	0.0172	0.0116	0.0250	0.0141
	Full	**0.0355**	**0.0226**	**0.0566**	**0.0294**	**0.0745**	**0.0483**	**0.1134**	**0.0609**	**0.0184**	**0.0121**	**0.0267**	**0.0148**

Component contributions. w/o Agg uses client-local relation counts. **w/o Trans**, **w/o CoOccur**, and **w/o Gate** remove the transition branch, co-occurrence branch, and gated residual fusion module, respectively. Each removal yields a lower mean across all datasets and metrics (Table 2), providing consistent evidence for the contribution of each component.

Relation-sharing alternatives. w/ Rand-Rel builds relation graphs from random item groups, while **w/ L2-Rel** moves graph construction, input embeddings, and injection to the second SID codeword. Both yield lower performance than **Full** across all datasets and metrics. This supports our choice of first-level SIDs, whose coarse content-based groups provide a common basis for pooling behavioral relations across distinct item pairs and clients.

Graph-conditioning alternatives. w/ Linear replaces each GAT with a linear projection of relation rows using separate learnable weights. Its lower performance than **Full** across all datasets and metrics is consistent with the benefit of propagating current SID embeddings over pooled relation neighborhoods to enrich local history representations.

5.4 Hyperparameter Sensitivity and Selection

Figure 3 shows Recall@5 sensitivity to one-at-a-time changes in window size W, GAT depth R, and mixing weight α, measured as relative gains over each dataset's strongest evaluated baseline (Table 1). Within these scans, $W = R = 3$ yields the highest mean Recall@5, with $\alpha = 0.7$, 0.1, and 0.7 for Video, Software, and Baby, respectively. These configurations improve mean Recall@5 by 1.43%, 3.91%, and 6.98% over FedProx on Video and Software and FedAvg on Baby. For new datasets or SID mappings, we suggest starting with $W = R = 3$, selecting α by validation Recall@5, and then tuning W and R, as the best settings may differ. Appendix C provides complete metrics and search ranges.

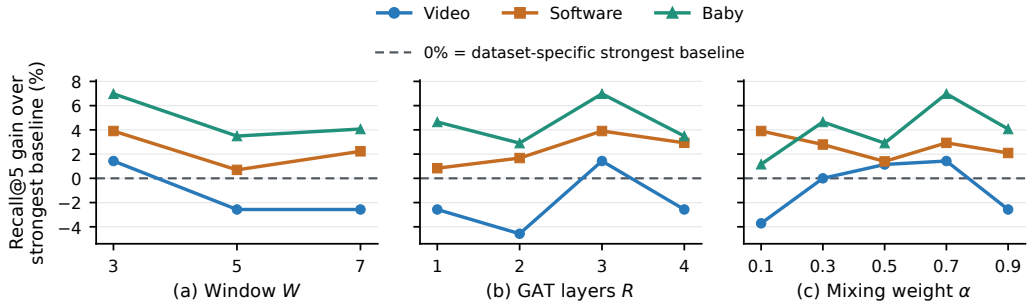


Figure 3: Hyperparameter sensitivity of Recall@5, expressed as relative gains over the strongest evaluated baseline on each dataset. The dashed line at 0% denotes parity with FedProx on Video and Software and FedAvg on Baby from Table 1. Each panel varies W, R, or α with the other two hyperparameters fixed at the dataset-specific reference configuration.

5.5 Impact of Data Heterogeneity and Federation Size

Starting from the default configuration $(N, \rho_{\text{ex}}, \gamma) = (3, 0.3, 10)$, we vary one partition factor at a time on Video: $\rho_{\text{ex}} \in \{0.3, 0.6, 0.9\}$, $\gamma \in \{1, 3, 10\}$, or $N \in \{3, 5, 7\}$. Lower values of γ produce more imbalanced allocations of shared-user sequences across clients. Figure 4 reports Recall@5 and NDCG@5. Corresponding Recall@10 and NDCG@10 results appear in Appendix D.

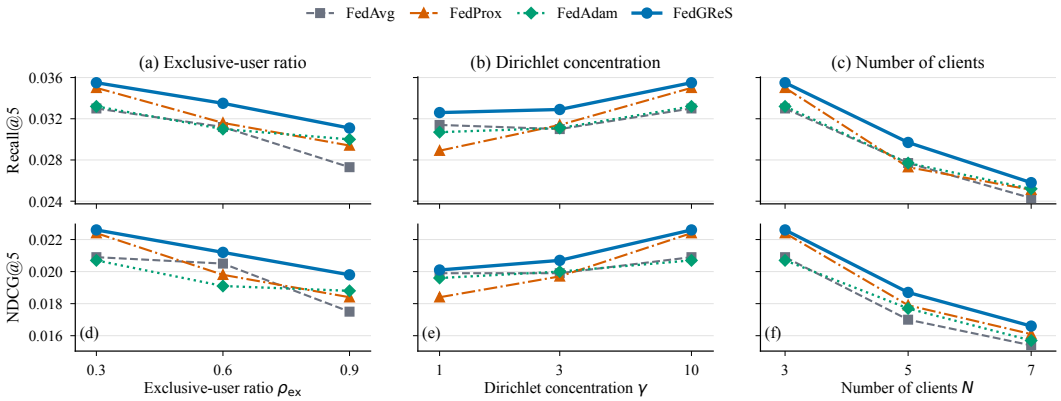


Figure 4: Video Recall@5 and NDCG@5 when varying one partition factor at a time from the default configuration $(N, \rho_{\text{ex}}, \gamma) = (3, 0.3, 10)$.

Effects of partition factors. Figure 4 shows that performance decreases for all four methods as ρ_{ex} increases, consistent with reduced user overlap across silos. Performance also decreases as N increases, suggesting that distributing shared-user histories across more clients produces shorter client-local subsequences and makes sequential patterns harder to learn. All methods perform better at $\gamma = 10$ than at $\gamma = 1$, consistent with a more even allocation of shared-user sequences, although not every baseline improves monotonically across the three settings.

Performance of `FedGReS` across configurations. `FedGReS` achieves the highest mean Recall@5 and NDCG@5 at every evaluated configuration, outperforming the strongest of FedAvg, FedProx, and FedAdam at each point by 0.89% to 7.22%. This consistent advantage supports the robustness of `FedGReS` to the tested partition variations on Video.

5.6 Graph and Representation Analysis

Graph fidelity. We compare client-local and federation-pooled graphs with a reference graph built from complete centralized training sequences. Across datasets and relation types, the federation-pooled graphs achieve Jensen-Shannon (JS) divergences that are 65.2% to 73.1% lower than the mean divergence of the three client-local graphs (Appendix E, Table 6 and Figure 7). These results indicate that federation-pooled graphs more closely match the reference neighbor-weight distributions than client-local graphs.

Behavioral evidence in pooled SID relations. Our design aims to combine sparse item-pair observations into shared behavioral relations between content-based groups. We examine this aggregation through two representative relations from Video in Figure 5(a). The transition from codeword 249 to 128 connects two Xbox-related semantic groups and pools 261 events across 242 distinct item pairs. The co-occurrence between codewords 68 and 253 connects Wii U- and Nintendo 3DS/2DS-related groups, with 550 events across 410 pairs. In these cases, exact item pairs contribute few observations on average, and both codeword relations receive support from all three clients. These examples illustrate how first-level SID graphs consolidate behavioral evidence across distinct item pairs and clients, providing a concrete interpretation of the relations shared by FedGReS.

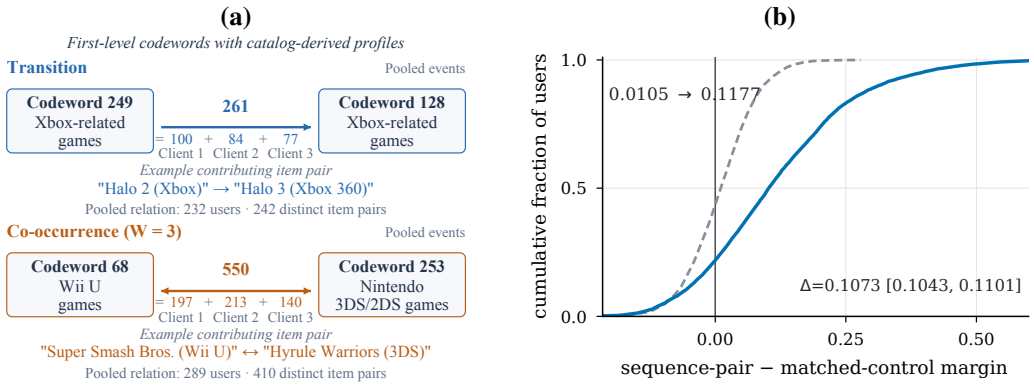


Figure 5: Video SID relations and representation effects. (a) Pooled SID relation case studies. (b) Empirical cumulative distributions of user margins before (gray dashed) and after (blue) graph conditioning. Brackets show the 95% paired user-bootstrap confidence interval for mean change Δ.

Representation alignment. To determine whether these pooled relations affect representation learning, we use a frozen model checkpoint for each dataset to compare first-level SID representations before and after graph conditioning. For each user, the *margin* measures how much more similar the target SID is to the two preceding SIDs than to matched control SIDs outside the user's sequence. Appendix E.3 details its computation. Figure 5(b) compares the empirical cumulative distributions of user margins before and after graph conditioning. A rightward shift indicates stronger alignment between the target SID and its recent sequence context relative to matched controls. On Video, the mean user margin increases from 0.0105 to 0.1177. Software and Baby exhibit the same pattern (Appendix Figure 9). These results indicate that graph conditioning strengthens target-history alignment relative to matched controls.

6 Conclusion

We introduced FedGReS, a cross-silo federated generative recommendation framework that complements federated parameter aggregation with first-level SID relation sharing. First-level SID groups pool relational evidence across clients and distinct item pairs, while graph propagation makes this evidence available to first-level history representations under the generation objective. The resulting mechanism connects coarse relation sharing with complete SID generation without user matching or raw-sequence exchange. Across three Amazon Reviews 2023 datasets, FedGReS achieves the highest mean result in all 12 dataset-metric settings, with a maximum relative improvement of 6.98% over the strongest baseline for the corresponding setting. Ablations and relation-graph analyses support the value of pooled SID relations beyond parameter aggregation alone.

AI USE STATEMENT

Generative AI tools were used to assist with migration of the manuscript between conference LaTeX templates, literature-grounded drafting of baseline descriptions, evidence-grounded drafting and related formatting checks. The authors reviewed the resulting changes and take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The model formulation is described in Sections 3 and 4. Appendix A details the federated optimization protocol and provides pseudocode. Appendix B documents the datasets, federated partition, baseline definitions, and model implementation details. Appendix E provides the graph-fidelity evaluation protocol and results, relation-edge case studies, and graph-conditioned first-level SID representation analysis.

REFERENCES

Muhammad Ammad-Ud-Din, Elena Ivannikova, Suleiman A Khan, Were Oyomno, Qiang Fu, Kuan Eeik Tan, and Adrian Flanagan. Federated collaborative filtering for privacy-preserving personalized recommendation system. *arXiv preprint arXiv:1901.09888*, 2019.

Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1175–1191. Association for Computing Machinery, 2017. doi: 10.1145/3133956.3133982. URL https://doi.org/10.1145/3133956.3133982.

Di Chai, Leye Wang, Kai Chen, and Qiang Yang. Secure federated matrix factorization. *IEEE Intelligent Systems*, 36(5):11–20, 2021. doi: 10.1109/MIS.2020.3014880.

Jiaxin Deng, Shiyao Wang, Kuo Cai, Lejian Ren, Qigen Hu, Weifeng Ding, Qiang Luo, and Guorui Zhou. Onerec: Unifying retrieve and rank with generative recommender and iterative preference alignment. *arXiv preprint arXiv:2502.18965*, 2025.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1511.06939.

Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952*, 2024.

Mubashir Imran, Hongzhi Yin, Tong Chen, Quoc Viet Hung Nguyen, Alexander Zhou, and Kai Zheng. Refrs: Resource-efficient federated recommender system for dynamic and diversified user preferences. *ACM Transactions on Information Systems*, 41(3):1–30, 2023.

Clark Mingxuan Ju, Liam Collins, Leonardo Neves, Bhuvesh Kumar, Louis Yufeng Wang, Tong Zhao, and Neil Shah. Generative recommendation with semantic ids: A practitioner's handbook. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM)*, CIKM '25, pp. 6420–6425, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400720406. doi: 10.1145/3746252.3761612. URL https://doi.org/10.1145/3746252.3761612.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 197–206, 2018. doi: 10.1109/ICDM.2018.00035.

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11523–11532, 2022.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems*, volume 2, pp. 429–450, 2020.

Yichen Li, Yijing Shan, YI LIU, Haozhao Wang, Cheng Wang, wangshi.ww, Yi Wang, and Ruixuan Li. Efficient knowledge transfer in federated recommendation for joint venture ecosystem. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 124805–124826. Curran Associates, Inc., 2025. doi: 10.52202/085713-4160. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/b4f9b809750373d111728b8a5f649ece-Paper-Conference.pdf.

Zhiwei Li, Guodong Long, and Tianyi Zhou. Federated recommendation with additive personalization. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=xkXdE81mOK.

Zhuodong Liu, Hugen Lv, Xiangyu Li, Bohan Guo, and Peiyu Hu. FedCGR: Federated cross-domain generative recommendation. *arXiv preprint arXiv:2608.10929*, 2026. URL https://arxiv.org/abs/2608.10929v1.

Sichun Luo, Yuanzhang Xiao, Yang Liu, Congduan Li, and Linqi Song. Towards communication efficient and fair federated personalized sequential recommendation. In *2022 5th International Conference on Information Communication and Signal Processing (ICICSP)*, pp. 1–6. IEEE, 2022.

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*, pp. 1273–1282. Pmlr, 2017.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 1864–1874, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.146. URL https://aclanthology.org/2022.findings-acl.146/.

Haohao Qu, Wenqi Fan, Zihuai Zhao, and Qing Li. Tokenrec: Learning to tokenize id for llm-based generative recommendations. *IEEE Transactions on Knowledge and Data Engineering*, 37(10): 6216–6231, 2025. doi: 10.1109/TKDE.2025.3599265.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Tran, Jonah Samost, et al. Recommender systems with generative retrieval. *Advances in Neural Information Processing Systems*, 36:10299–10315, 2023.

Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and H. Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=LkFG3lB13U5.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Wenjie Wang, Honghui Bao, Xinyu Lin, Jizhi Zhang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. Learnable item tokenization for generative recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 2400–2409, 2024a.

Ye Wang, Jiahao Xun, Minjie Hong, Jieming Zhu, Tao Jin, Wang Lin, Haoyuan Li, Linjun Li, Yan Xia, Zhou Zhao, and Zhenhua Dong. Eager: Two-stream generative recommender with behavior-semantic collaboration. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, pp. 3245–3254, New York, NY, USA, 2024b. Association for Computing Machinery. ISBN 9798400704901. doi: 10.1145/3637528.3671775. URL https://doi.org/10.1145/3637528.3671775.

Yidan Wang, Zhaochun Ren, Weiwei Sun, Jiyuan Yang, Zhixiang Liang, Xin Chen, Ruobing Xie, Su Yan, Xu Zhang, Pengjie Ren, et al. Content-based collaborative generation for recommender systems. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 2420–2430, 2024c.

Longtao Xiao, Haozhao Wang, Cheng Wang, Linfei Ji, Yifan Wang, Jieming Zhu, Zhenhua Dong, Rui Zhang, and Ruixuan Li. Unger: Generative recommendation with a unified code via semantic and collaborative integration. *ACM Trans. Inf. Syst.*, 44(2), December 2025. ISSN 1046-8188. doi: 10.1145/3773771. URL https://doi.org/10.1145/3773771.

Yuhao Yang, ZhI JI, Zhaopeng Li, Yi Li, Zhonglin Mo, Yue Ding, Kai Chen, Zijian Zhang, Jie Li, shuanglong li, and LIU LIN. Sparse meets dense: Unified generative recommendations with cascaded sparse-dense representations. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 93746–93770. Curran Associates, Inc., 2025. doi: 10.52202/085713-3135. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/86ba836d4c5dd859d795a172911745e2-Paper-Conference.pdf.

Wei Yuan, Chaoqun Yang, Liang Qu, Quoc Viet Hung Nguyen, Guanhua Ye, and Hongzhi Yin. Ptf-fsr: A parameter transmission-free federated sequential recommender system. *ACM Transactions on Information Systems*, 43(2):1–24, 2025a.

Wei Yuan, Chaoqun Yang, Guanhua Ye, Tong Chen, Quoc Viet Hung Nguyen, and Hongzhi Yin. Fellas: Enhancing federated sequential recommendation with llm as external services. *ACM Transactions on Information Systems*, 43(6):1–24, 2025b.

Huimin Zeng, Zhenrui Yue, Qian Jiang, and Dong Wang. Federated recommendation via hybrid retrieval augmented generation. In *2024 IEEE International Conference on Big Data (BigData)*, pp. 8078–8087. IEEE, 2024.

Chunxu Zhang, Guodong Long, Tianyi Zhou, Peng Yan, Zijian Zhang, Chengqi Zhang, and Bo Yang. Dual personalization on federated recommendation. In Edith Elkind (ed.), *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pp. 4558–4566. International Joint Conferences on Artificial Intelligence Organization, 8 2023. doi: 10.24963/ijcai.2023/507. URL https://doi.org/10.24963/ijcai.2023/507. Main Track.

Hongyu Zhang, Dongyi Zheng, Xu Yang, Jiyuan Feng, and Qing Liao. Feddcsr: Federated cross-domain sequential recommendation via disentangled representation learning. In *Proceedings of the 2024 SIAM International Conference on Data Mining (SDM)*, pp. 535–543. SIAM, 2024. doi: 10.1137/1.9781611978032.62.

Jujia Zhao, Wenjie Wang, Chen Xu, See-Kiong Ng, and Tat-Seng Chua. A federated framework for LLM-based recommendation. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 2852–2865, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.155. URL https://aclanthology.org/2025.findings-naacl.155/.

Bowen Zheng, Yupeng Hou, Hongyu Lu, Yu Chen, Wayne Xin Zhao, Ming Chen, and Ji-Rong Wen. Adapting large language models by integrating collaborative semantics for recommendation. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pp. 1435–1448. IEEE, 2024.

Jieming Zhu, Mengqun Jin, Qijiong Liu, Zexuan Qiu, Zhenhua Dong, and Xiu Li. Cost: Contrastive quantization based semantic tokenization for generative recommendation. In *Proceedings of the 18th ACM Conference on Recommender Systems*, RecSys '24, pp. 969–974, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400705052. doi: 10.1145/3640457.3688178. URL https://doi.org/10.1145/3640457.3688178.

A Federated Training Protocol and Pseudocode

Graph construction is performed once from the local training sequences, after which A^{tr} and A^{co} remain fixed. For `FedGReS`, we extend θ to include all trainable parameters of the generative backbone, SID embedding table, two GAT branches, and gated-fusion module; the precomputed graphs and fixed coefficient α are excluded from θ. At communication round t, the server broadcasts θ^t, and each client P_k initializes from it and performs E local optimization steps on $\mathcal{D}_k$. Each forward pass computes the graph-conditioned first-level block through Eqs. (13)-(16), and all trainable components are updated jointly using $\mathcal{L}_{\mathrm{rec}}$ in Eq. (4). Clients return (θ_k^{t+1}, m_k^t); the server aggregates $\theta^{t+1} = \sum_{k=1}^{N} p_k^t \theta_k^{t+1}$ using the weights in Eq. (6). Algorithm 1 gives the pseudocode.

Algorithm 1 Complete `FedGReS` Training Protocol

Input: Clients $\mathcal{P} = \{P_k\}_{k=1}^N$ with nonempty private datasets $\{\mathcal{D}_k\}_{k=1}^N$ and local training-sequence sets $\mathcal{Q}_k = \{\mathcal{S}_u^{(k)}\}_{u \in \mathcal{U}_k}$; shared, fixed, and aligned item-SID mapping τ; codebook size K; window size $W \geq 2$; sparsity $1 \leq K_{\mathrm{top}} \leq K$; GAT depth $R \geq 1$; fixed $\alpha \in [0, 1]$; initial parameters θ^0; rounds $T_{\mathrm{FL}} \geq 1$; local steps $E \geq 1$; local optimizer OptStep

Output: Trained parameters $\theta^{T_{\mathrm{FL}}}$ and fixed graph inputs $A^{\mathrm{tr}}, A^{\mathrm{co}}$

Phase I: one-time federation-pooled relation-graph construction
for $k = 1, \ldots, N$ **in parallel do**
  Compute $T^{(k)}$ and $O^{(k)}$ from $\mathcal{Q}_k$ using Eqs. (7)-(8)
  Submit $(T^{(k)}, O^{(k)})$ for additive aggregation
end for
Server obtains T and O using Eq. (9)
Sparsify each row of T and O independently by retaining up to K_{top} largest positive entries
Row-normalize the retained counts using Eq. (11) to obtain A^{tr} and A^{co}
Server broadcasts both graph inputs once; all clients keep them fixed
Phase II: graph-conditioned federated optimization
for $t = 0, \ldots, T_{\mathrm{FL}} - 1$ **do**
  Server broadcasts θ^t to every client
  for $k = 1, \ldots, N$ **in parallel do**
    $\theta_k^{t,0} \leftarrow \theta^t$; $m_k^t \leftarrow 0$
    for $e = 0, \ldots, E - 1$ **do**
      Sample and augment local data to form training mini-batch $\mathcal{B}_k^{t,e}$
      From the current local state $\theta_k^{t,e}$, recompute the temporary $\widetilde{\mathbf{E}}^{(1)}$ using Eqs. (13)-(16)
      Compute the teacher-forced loss $\ell_k^{t,e} \leftarrow \mathcal{L}_{\mathrm{rec}}(\theta_k^{t,e}; \mathcal{B}_k^{t,e})$ using Eq. (4), with $\widetilde{\mathbf{E}}^{(1)}$ used only for encoder-side first-level tokens and all other embedding blocks unconditioned but trainable
      $\theta_k^{t,e+1} \leftarrow \mathrm{OptStep}\left(\theta_k^{t,e}, \nabla_{\theta_k^{t,e}} \ell_k^{t,e}\right)$; $m_k^t \leftarrow m_k^t + |\mathcal{B}_k^{t,e}|$
    end for
    $\theta_k^{t+1} \leftarrow \theta_k^{t,E}$; return (θ_k^{t+1}, m_k^t) to the server
  end for
  $p_k^t \leftarrow m_k^t / \sum_{j=1}^N m_j^t$ for $k = 1, \ldots, N$
  $\theta^{t+1} \leftarrow \sum_{k=1}^N p_k^t \theta_k^{t+1}$
end for
return $\theta^{T_{\mathrm{FL}}}, A^{\mathrm{tr}}, A^{\mathrm{co}}$

B Detailed Experimental Setup

B.1 Datasets, Partitioning, and Evaluation

Table 3: Dataset statistics for the three real-world Amazon Reviews 2023 datasets.

Dataset	Start Time	End Time	# Users	# Items	Sequence Length		
					Mean	Min	Max
Baby	1996-05	2020-04	8,995	36,297	21.03	15	60
Software	1996-05	2023-09	15,444	34,350	22.99	15	60
Video	1996-05	2020-10	7,953	32,872	22.90	15	60

We use three real-world datasets from the public Amazon Reviews 2023 corpus (Hou et al., 2024), corresponding to the Baby Products (**Baby**), **Software**, and Video Games (**Video**) categories. Following the sequence-construction convention of Rajput et al. (2023), we sort each user's review history by timestamp to form a chronological item sequence. To support the subsequent division into client-local subsequences while bounding sequence length, we retain users with 15-60 reviews. Table 3 reports dataset statistics after this filtering.

Following the user-partitioning protocol of Li et al. (2025), adapted here from tabular interactions to sequential data, we simulate cross-silo partitions as follows. We set the exclusive-user ratio to $\rho_{\text{ex}} = 0.3$, assign each exclusive user's complete chronological sequence to exactly one of $N = 3$ clients, and distribute these users evenly across clients. The remaining $1 - \rho_{\text{ex}} = 0.7$ of users are treated as shared users. For each shared user u, we sample client-allocation proportions as $\boldsymbol{\pi}_u \sim \text{Dirichlet}(\gamma \mathbf{1}_N)$ with $\gamma = 10$ and partition the full sequence accordingly into contiguous client-local subsequences, preserving chronological order within each client. Client-user subsequences with fewer than 5 interactions are discarded before leave-one-out evaluation.

We apply leave-one-out separately to each retained client-local sequence: its last item is the test target, its penultimate item is the validation target, and the remaining prefix is used for training. Relation matrices are computed exclusively from the resulting training prefixes; validation and test interactions are excluded. Within each client, Recall@K and NDCG@K are averaged over its retained test records. Client-level means are then weighted by their numbers of test records, which is equivalent to a micro-average over all retained client-local test records in a run.

Decoding and evaluation protocol. All TIGER-style methods use the same autoregressive beam-search decoder with a beam width of 10. Four-token SID sequences are ranked by the product of their conditional token probabilities without prefix-validity constraints. Evaluation uses exact SID matching. Generated SIDs that do not correspond to catalog items retain their ranked positions and count as misses, without replacement or replenishment. No history-item filtering or additional deduplication is applied to the generated outputs.

B.2 Baseline Descriptions

PFedRec (Zhang et al., 2023). PFedRec is a personalized federated recommender that combines shared item embeddings with private user-specific scoring functions and locally adapted item representations. Our cross-silo adaptation retains this dual-personalization design for silo-local users.

FedRAP (Li et al., 2024). FedRAP introduces additive personalization by representing each item as the sum of globally shared and private embeddings, coupled with local user representations. Only the shared component is federated, enabling global knowledge transfer while preserving silo-specific preferences.

GRU4Rec+FedAvg (Hidasi et al., 2016; McMahan et al., 2017). GRU4Rec is a seminal recurrent sequential recommender that encodes interaction histories with gated recurrent units and directly scores catalog items. Unlike TIGER's autoregressive Semantic-ID generation, it predicts item IDs from recurrent states; we federate it using FedAvg.

SASRec+FedAvg (Kang & McAuley, 2018; McMahan et al., 2017). SASRec is a canonical self-attentive sequential recommender that combines positional embeddings with causal self-attention to capture long-range dependencies. It directly ranks item IDs rather than generating Semantic IDs; we federate it using FedAvg.

FedDCSR (Zhang et al., 2024). FedDCSR targets federated cross-domain sequential recommendation by disentangling sequence representations into domain-shared and domain-exclusive components and enhancing the latter through contrastive infomax. We treat each silo as a pseudo-domain and federate only shared knowledge.

FedAvg (McMahan et al., 2017). FedAvg is the foundational federated learning algorithm, alternating local client optimization with sample-weighted parameter averaging at the server. We apply it to the same graph-free TIGER-style backbone as the principal backbone-matched baseline.

FedProx (Li et al., 2020). FedProx is a classical heterogeneity-aware extension of FedAvg that adds a proximal penalty to constrain local models around the broadcast global model. We apply it to the same graph-free TIGER-style backbone.

FedAdam (Reddi et al., 2021). FedAdam is a representative adaptive federated optimizer that applies Adam-style first- and second-moment estimation to aggregated client updates at the server. It uses the same graph-free TIGER-style backbone.

B.3 Model Implementation Details

For each item, we serialize its title, brand, categories, and price as a single text sequence, truncate it to 128 tokens, and encode it with the frozen pre-trained Sentence-T5-XXL model (Ni et al., 2022). Masked mean pooling over the final token representations yields a 1,024-dimensional semantic embedding.

We quantize these embeddings using 3-level Residual K-Means (RKM) (Deng et al., 2025), with 256 centroids per level. The levels are trained sequentially for 300, 400, and 500 optimization steps, respectively (1,200 steps in total), using SGD with a learning rate of 0.5. We use an RKM batch size of 4,096 for all datasets. Following Rajput et al. (2023), we append a collision-disambiguation token to each 3-level code, yielding a unique 4-token SID for every item.

All TIGER-backbone methods use 4 Transformer encoder layers and 4 decoder layers, 6 attention heads per layer, $d_{\text{model}} = 128$, $d_{\text{kv}} = 64$, and 2 feed-forward hidden layers of width 1,024. We use a dropout rate of 0.15. Our implementation builds on the GRID codebase of Ju et al. (2025).

B.4 Federated Implementation Details

Unless otherwise specified, all federated TIGER-backbone methods use 3 fully participating clients. Each round, clients initialize from the global model and take 800 local Adam steps (learning rate 7×10^{-4}, weight decay 10^{-4}). Each data-loader batch has 32 local sequences. Contiguous-subsequence augmentation yields at most 512 training examples per step. We sum the actual augmented batch sizes to obtain m_k^t, counting each repeated draw. FedAvg, FedProx, and `FedGReS` use $p_k^t = m_k^t / \sum_j m_j^t$. FedAdam applies its adaptive server update to the same weighted client aggregate.

We tuned the hyperparameters of all baselines and used their best-performing configurations. For FedProx, the selected proximal coefficient was 10^{-3}. For FedAdam, the selected server learning rate was 5×10^{-3}, with $\beta_1 = 0.9$, $\beta_2 = 0.99$, $\tau = 10^{-4}$, and bias correction. For `FedGReS`, we pool client relation counts once to construct fixed graphs. The default configuration uses $W = 3$, $K_{\text{top}} = 15$, and 3 layers in each transition and co-occurrence GAT branch, with 4 attention heads per layer and a dropout rate of 0.1. We use gated residual fusion. The sensitivity and federation-size studies vary the corresponding factors as specified in Sections 5.4 and 5.5. For each dataset and experimental setting, we select the dual-graph mixing coefficient from $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ according to validation $\mathrm{Recall@5}$. We cap training at 100 communication rounds and select checkpoints by validation $\mathrm{Recall@5}$. We repeated each recommendation experiment with three different random seeds, keeping the data partition, item-SID mapping, hyperparameters, and all other experimental settings fixed. We report the mean performance across the three runs. Appendix E describes the fixed split and selected checkpoints used for the graph and representation analyses.

B.5 Run-to-Run Variability

Table 4 reports the mean and sample standard deviation of `FedGReS` over the same three runs used for its results in Table 1.

Table 4: Mean $\pm$ sample standard deviation of `FedGReS` over three runs with different random seeds. All other experimental settings were fixed. R/N denote Recall/NDCG.

Dataset	R@5	N@5	R@10	N@10
Video	0.0355 ± 0.00093	0.0226 ± 0.00046	0.0566 ± 0.00046	0.0294 ± 0.00020
Software	0.0745 ± 0.00095	0.0483 ± 0.00012	0.1134 ± 0.00030	0.0609 ± 0.00021
Baby	0.0184 ± 0.00097	0.0121 ± 0.00078	0.0267 ± 0.00031	0.0148 ± 0.00055

C Hyperparameter Sensitivity and Selection Details

Search procedure and sensitivity scans. Parameter exploration follows iterative coordinate search: two parameters are fixed while the third is varied, and an improved configuration becomes the reference for subsequent iterations. The reported scans vary $W \in \{3, 5, 7\}$, $R \in \{1, 2, 3, 4\}$, or $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ around $(W, R, \alpha) = (3, 3, 0.7)$ for Video/Baby and $(3, 3, 0.1)$ for Software, with the other two parameters fixed. These are conditional slices, not an exhaustive joint grid. The same α candidate set is used for the main comparisons (Appendix B.4).

Table 5 gives all four metrics, averaged over three runs, for the 30 dataset-configuration combinations in the reported sensitivity scans. Figure 3 plots the relative gain $g_d = 100(s_d/b_d - 1)$ for each dataset d, where s_d is the mean Recall@5 of the evaluated `FedGReS` configuration and b_d is the highest mean Recall@5 among the eight baselines in Table 1. We use $b_d = 0.0350$, 0.0717, and 0.0172 for Video, Software, and Baby, respectively, corresponding to FedProx, FedProx, and FedAvg. Each dataset is normalized by its own baseline, so the common dashed line at 0% represents parity with the corresponding baseline. Positive and negative values indicate higher and lower mean Recall@5, respectively. The reference configurations achieve the highest Recall@5 within the reported scans, with means of 0.0355, 0.0745, and 0.0184 for Video, Software, and Baby, respectively.

Table 5: Complete hyperparameter results (means of three runs). R and N denote Recall and NDCG. Stars mark the reference configurations. The highest value of each metric within each dataset is bold.

	Video				Software				Baby			
Setting	R@5	N@5	R@10	N@10	R@5	N@5	R@10	N@10	R@5	N@5	R@10	N@10
$W = 5$	0.0341	0.0217	0.0544	0.0283	0.0722	0.0469	0.1109	0.0593	0.0178	0.0116	0.0258	0.0142
$W = 7$	0.0341	0.0215	0.0547	0.0281	0.0733	0.0472	0.1121	0.0598	0.0179	0.0120	0.0256	0.0144
$R = 1$	0.0341	0.0216	0.0529	0.0276	0.0723	0.0468	0.1119	0.0596	0.0180	0.0117	0.0255	0.0142
$R = 2$	0.0334	0.0208	0.0543	0.0275	0.0729	0.0470	0.1106	0.0592	0.0177	0.0116	0.0265	0.0144
$R = 4$	0.0341	0.0218	0.0534	0.0279	0.0738	0.0477	0.1126	0.0602	0.0178	0.0119	0.0260	0.0146
$\alpha = 0.1$	0.0337	0.0215	0.0546	0.0282	**0.0745***	**0.0483**	0.1134	**0.0609**	0.0174	0.0115	0.0251	0.0139
$\alpha = 0.3$	0.0350	0.0222	0.0557	0.0288	0.0737	0.0480	0.1122	0.0604	0.0180	0.0118	0.0261	0.0144
$\alpha = 0.5$	0.0354	0.0225	0.0564	0.0292	0.0727	0.0470	**0.1146**	0.0597	0.0177	0.0117	0.0254	0.0141
$\alpha = 0.7$	**0.0355***	**0.0226**	**0.0566**	**0.0294**	0.0738	0.0481	0.1133	**0.0609**	**0.0184***	**0.0121**	**0.0267**	**0.0148**
$\alpha = 0.9$	0.0341	0.0218	0.0539	0.0281	0.0732	0.0474	0.1128	0.0602	0.0179	0.0120	0.0262	0.0147

Practical selection guidance. For a new dataset or SID mapping, start from $W = R = 3$, select α by validation Recall@5, and then revisit W and R. With a limited tuning budget, $W \in \{3, 5\}$, $R \in \{2, 3, 4\}$, and $\alpha \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ provide an initial search range; expand it if the best validation setting lies at a boundary or performance remains unsatisfactory. Repeat validation after changing the data or SID construction, since the present scans do not establish transferable optimal constants.

D Cutoff-10 Results under Federated Heterogeneity

The cutoff-10 results reproduce the strict declines of all four methods as ρ_{ex} and N increase. The response to γ is not uniformly monotonic for every baseline, but each method's value at $\gamma = 10$ exceeds its value at 1. `FedGReS` is never below any baseline and is strictly highest at every cutoff-10 point except NDCG@10 at $\gamma = 1$, where it ties FedAdam at 0.0265.

E Graph and Representation Analysis

E.1 Graph Fidelity

We compared client-local and federation-pooled graphs with a reference graph constructed from complete centralized training sequences. Each local graph used its client's complete training sequences, and the pooled graph summed the three clients' raw counts before graph construction. All

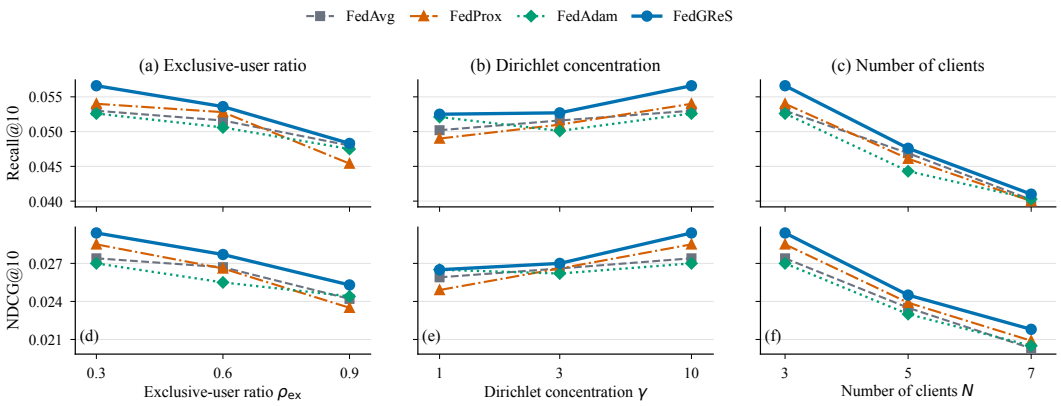


Figure 6: Cutoff-10 results for the controlled Video partitions (means of three runs). The top and bottom rows report Recall@10 and NDCG@10 for the same four methods; columns, controls, and visual encodings match Fig. 4. Markers denote reported means and lines are visual guides; higher is better.

graphs used the same 256 first-level SID nodes, directed lag-one transition counts, and windowed co-occurrence counts with $W = 3$, followed by top-15 row-wise pruning and row normalization. This diagnostic used the full training sequences rather than augmented training examples.

The federated training sequences are not a lossless partition of centralized training. Shared-user histories are divided across clients, and each client-local segment is truncated separately for leave-one-out evaluation. This removes some interactions and cross-client boundary relations, so pooling local counts does not reproduce the centralized reference exactly. The comparison measures how closely the local and pooled graphs match this empirical reference.

We measured Jensen-Shannon (JS) divergence between neighbor-weight distributions using base-2 logarithms and averaged it over reference-active source rows. Lower values indicate closer agreement, with zero denoting identical distributions. Local results summarize the three client graphs. We computed ΔJS as the local mean minus pooled JS, so positive values favor pooling. The 95% descriptive intervals (DIs) were obtained from 2,000 paired source-row bootstrap resamples on the fixed split. They describe variation across source rows rather than variation across clients or random splits.

Table 6: Graph fidelity against the centralized reference. Lower JS divergence is better. Local values are the mean $\pm$ standard deviation across three client graphs. ΔJS is the local mean minus pooled JS, with 95% source-row bootstrap descriptive intervals in brackets.

Dataset	Relation	Local JS	Pooled JS	ΔJS [95% DI]
Video	Transition	0.2690 ± 0.0051	**0.0723**	$0.1968\ [0.1821, 0.2107]$
Video	Co-occurrence	0.1955 ± 0.0024	**0.0553**	$0.1402\ [0.1266, 0.1544]$
Software	Transition	0.2802 ± 0.0025	**0.0845**	$0.1957\ [0.1783, 0.2141]$
Software	Co-occurrence	0.2150 ± 0.0011	**0.0605**	$0.1545\ [0.1356, 0.1727]$
Baby	Transition	0.4226 ± 0.0035	**0.1469**	$0.2757\ [0.2639, 0.2881]$
Baby	Co-occurrence	0.3224 ± 0.0005	**0.1107**	$0.2117\ [0.1982, 0.2252]$

Table 6 summarizes discrepancies across reference-active source rows. Figure 7 complements this comparison by showing edge-weight errors for selected high-frequency SID nodes. Both analyses indicate closer agreement with the centralized reference after pooling.

For the edge-weight visualization, both axes use the same eight first-level SIDs, selected by the highest outgoing reference counts. We sliced their weights from the full 256-node, top-15 row-normalized graphs without further normalization. Within each dataset and relation type, the local

and pooled panels share a zero-based linear color scale whose upper limit is the largest block error across all three local graphs and the pooled graph, with no clipping.

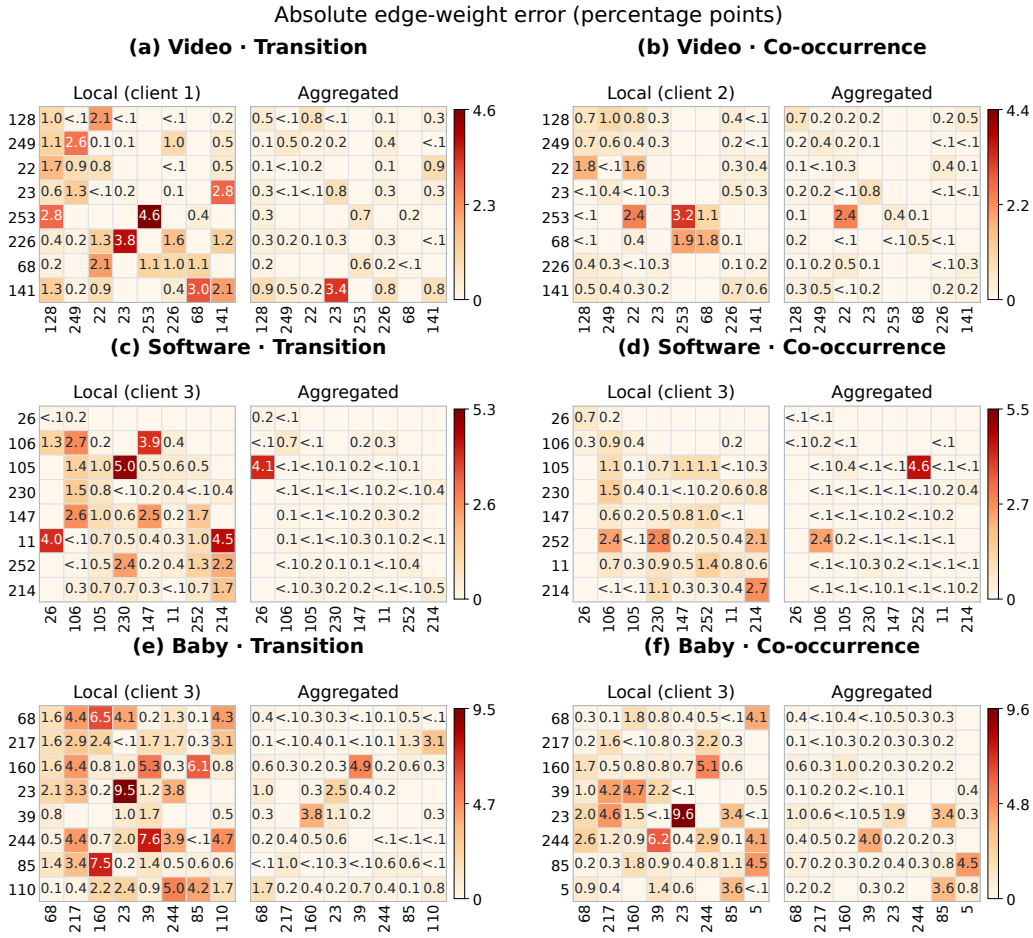


Figure 7: Edge-weight errors relative to the centralized reference. Rows show Video, Software, and Baby, and columns show transition and co-occurrence relations. Each pair compares a local graph (left) with the pooled graph (right). Axes index source and neighbor SIDs. Cell values show $100|A_{ij} - A_{ij}^{\mathrm{ref}}|$ in percentage points. Blank cells have zero error, and <0.1 denotes a positive error below 0.1 percentage points. Lighter colors indicate smaller errors. Color scales are shared within each pair but differ across pairs.

E.2 SID Relation Case Studies

Figure 8 extends the case analysis in Section 5.6 to Software and Baby. These examples illustrate how a codeword relation pools observations from multiple distinct item pairs. Node profiles are derived from catalog information and describe overlapping semantic groups, so different codewords may share a broad category. Software profiles use product-title terms because official categories are sparse.

Event counts, example item pairs, and coverage statistics were computed from the contributing client-local training events without concatenating sequences across clients. Coverage summarizes the entire codeword relation rather than the displayed example pair. For this offline case analysis, users were deduplicated by benchmark user ID across clients, and item pairs by global catalog IDs, preserving transition direction and aligning co-occurrence endpoints by SID. These user identifiers were not used by the federated training protocol.

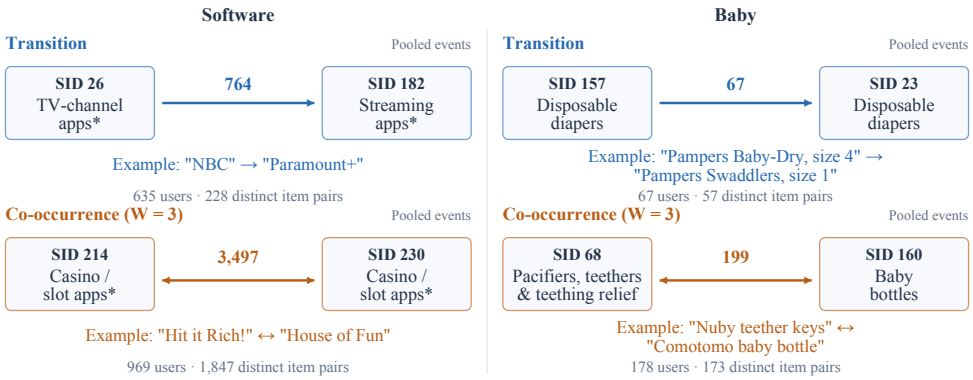


Figure 8: Pooled SID relation case studies on Software and Baby. Blue arrows show directed transitions, and orange links show co-occurrences within $W = 3$. Counts and coverage refer to the full codeword relation, with one contributing item pair shown as an example. Software profiles ($*$) use product-title terms.

E.3 Graph-Conditioned SID Representation Analysis

For each dataset, we select one compatible `FedGReS` checkpoint by validation Recall@5 before the analysis. With dropout disabled, we extract the same 256 first-level SID vectors in 128 dimensions before the graph module and after dual-GAT encoding and gated residual fusion. The latter vectors are the representations passed to the Transformer.

For this analysis, u indexes a client-local user sequence. A shared user appearing at multiple clients is evaluated separately within each client. For each sequence, we paired the final test target SID t with the SIDs q of the preceding two items, excluding same-SID pairs, to form $\mathcal{P}_u$. The target and lag-1 positions were held out from graph construction, and lag-2 was the last training-prefix item. Each pair was compared with all SIDs absent from the complete client-local sequence that matched q's training-frequency and deployed-graph-degree quintiles, using frequency alone when the joint group was empty. This control set $\mathcal{N}_{u,q}$ was fixed across stages. For $s \in \{\text{before}, \text{after}\}$, the margin for client-local sequence u is

$$m_u^{(s)} = \frac{1}{|\mathcal{P}_u|} \sum_{(t,q)\in\mathcal{P}_u} \left[\cos\left(\mathbf{z}_t^{(s)}, \mathbf{z}_q^{(s)}\right) - \frac{1}{|\mathcal{N}_{u,q}|} \sum_{n\in\mathcal{N}_{u,q}} \cos\left(\mathbf{z}_t^{(s)}, \mathbf{z}_n^{(s)}\right) \right]. \tag{17}$$

Larger margins indicate stronger similarity to sequence-related SIDs relative to matched controls. All margin confidence intervals (CIs) used 5,000 paired bootstrap resamples of client-local user sequences, with the before and after margins resampled together.

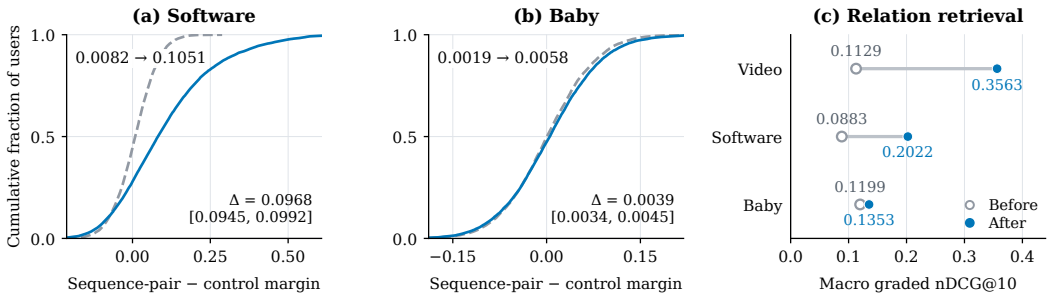


Figure 9: Sequence alignment and relation retrieval before and after graph conditioning. (a,b) Software and Baby ECDFs of client-local user margins, before (gray dashed) and after (blue). Annotations give means across client-local sequences and their paired change with a bracketed 95% CI. Dataset-specific x-ranges omit the outer 0.25% of pooled margins for display only. (c) Macro graded nDCG@10 on all three datasets, before (gray open circles) and after (blue filled circles).

To test whether related SIDs are easier to retrieve, we use each source SID with at least five held-out lag-1/2 events as a query and rank the other 255 SIDs by cosine similarity. Symmetrized event counts define graded relevance, so frequently observed sequence relations receive greater weight. We compute nDCG@10 for each query and average equally across queries. A higher score means that strongly related SIDs appear closer to the top of the ranking.

The margin distributions in Figure 5(b) and Figure 9(a,b), together with the retrieval results in Figure 9(c), show consistent improvements across all three datasets. Larger margins mean that sequence-related SIDs gain similarity relative to matched controls; higher nDCG@10 shows that these relations are also easier to recover by ranking. These results support stronger sequence alignment and relation retrieval after graph conditioning.