Online Resource 1

S1. FedGPR module implementation details

TPGNN-FedGPR consists of three main modules:

(1) User geographic association modeling: We mitigate cold-start and sparsity issues by integrating location-aware neighbor signals into the construction of user representations.

(2) Triple-level privacy-preserving GNN construction: We apply a privacy-by-design strategy at three stages: embedding construction, local prediction, and gradient perturbation.

(3) Federated optimization with fuzzy client selection: To enhance communication efficiency and training stability, we sample clients using a fuzzy clustering-based strategy and aggregate weighted gradients for federated optimization.

By integrating these modules, TPGNN-FedGPR achieves high-quality POI recommendation while mitigating privacy threats and ensuring efficient training in federated environments. The complete training and update process is outlined in Alg. 1.

S1.1. User geographic association modeling

To capture latent spatial correlations among users and enhance user embeddings under sparse interactions, we adopt a two-stage geographic modeling scheme: (1) *geographic gridding* and (2) *user link establishment*.

S1.1.1. Geographic gridding

Before constructing geographic links between users, we partition the entire area encompassing all POIs into discrete spatial grids. This grid-based generalization groups geographically proximate POIs [1], facilitating the modeling of spatial correlations among items and users. Formally, we divide the area into $\mathbb{L}$ square grids, denoted as $\mathcal{R} = \{\mathbb{G}_1, \mathbb{G}_2, \ldots, \mathbb{G}_{\mathbb{L}}\}$, where each grid $\mathbb{G}_l$ contains a set of POIs $\mathbb{G}_l = \{p_l^1, p_l^2, \ldots, p_l^{\mathbb{M}}\}$ and is uniquely indexed. An illustrative example of the grid partitioning is shown in Fig. 1.

To ensure spatial consistency, we adopt a uniform 2D tiling strategy in a planar coordinate system. Specifically, the map is partitioned into square grids of size 1 km $\times$ 1 km based on projected coordinates, ensuring comparable spatial density across regions. This grid resolution strikes a balance between semantic precision and data availability: finer grids better capture local spatial semantics but may lead to increased sparsity and reduced user overlap, while coarser grids provide broader coverage at the cost of geographic specificity.

S1.1.2. User links establishment

We assume that POIs within the same grid often exhibit functional similarity (e.g., shopping centers, business districts, or tourist attractions). Based on this assumption, users who interact

Algorithm 1: *TPGNN-FedGPR*

Input: Client models $\{\mathcal{M}_1, \mathcal{M}_2, ..., \mathcal{M}_N\}$ with embeddings $\{\mathbf{e}_u^1, \mathbf{e}_u^2, ..., \mathbf{e}_u^N\}$, item embeddings $\{\mathbf{e}_p^1, \mathbf{e}_p^2, ..., \mathbf{e}_p^M\}$, learning rate η

Output: Model parameters θ_{global}, local user interaction item rating $\hat{R}_{up}$

Global Server:
Initializing θ_{global};
$S_u \leftarrow$ Calculate gradient sensitivity with Hasse Diagram;
while *not converge* **do**
  $\mathcal{M} \leftarrow$ Fuzzy clustering-based client selection;
  Send the latest global model parameters θ_{global} and corresponding gradient sensitivity in S_u for each model in $\mathcal{M}$;
  $\mathrm{g}^{(n)}, \mathcal{S}_n \leftarrow$ Uploading gradient with differential privacy (Eq. (12));
  $\bar{\mathrm{g}}_m \leftarrow \frac{\sum_{n\in\mathcal{M}} |\mathcal{S}_n| \cdot \tilde{\mathrm{g}}_m^{(n)}}{\sum_{n\in\mathcal{M}} |\mathcal{S}_n|}$ (Eq. (13));
  $\bar{\mathrm{g}}_{r,p} \leftarrow \frac{\sum_{n\in\mathcal{M}} |\mathcal{S}_n| \cdot (\tilde{\mathrm{g}}_r^{(n)} || \tilde{\mathrm{g}}_p^{(n)})}{\sum_{n\in\mathcal{M}} |\mathcal{S}_n|}$ (Eq. (13));
  $\theta_{\text{global}} \leftarrow \theta_{\text{global}} - \eta \cdot \bar{\mathrm{g}}_m$;
  $\mathbf{e}_p^m \leftarrow \mathbf{e}_p^m - \eta \cdot \bar{\mathrm{g}}_{r,p}$;
end
return θ_{global}
Local Client:
Download the final model parameters and item embeddings from **Server**;
$\mathbf{e}_{u_n}^* \leftarrow \gamma_s \cdot \mathbf{e}_u + \gamma_u \cdot \mathbf{t}_{\mathcal{G}} + \gamma_p \cdot \mathbf{t}_{\mathcal{P}}$; // Compute local client embedding.(Eq.(6))
$\hat{R}_{up} \leftarrow \mathbf{e}_{u_n}^* \cdot \mathbf{e}_p$; //Predict user ratings for items.
return $\hat{R}_{up}$

with POIs in the same grid are likely to share similar preferences. Geographic links between users are constructed in three steps:

User-to-grid mapping: For each grid $\mathbb{G}_l$, the set of users who have interacted with its POIs is recorded as $\mathbb{G}_l^u = \{u_i, u_j, \ldots, u_k\}$, where $i, j, k \in N$ and N denotes the total number of users.

Grid-to-user mapping: For each user u_n, the set of grids containing their interacted POIs is recorded as $u_n^{\mathbb{G}} = \{\mathbb{G}_i, \mathbb{G}_j, \ldots, \mathbb{G}_k\}$, where $i, j, k \in \mathbb{L}$.

User-user link construction: A geographic link is established between two users if the number of shared grids exceeds a predefined threshold $T_{\mathbb{G}}$. The resulting neighbor set for each user u_n is denoted as $ul_n = \{u_i, u_j, \ldots, u_k\}$, where $i, j, k \in N$.

These inferred user links are integrated into user embeddings to preserve high-order spatial dependencies. This design ensures that preference modeling is guided not only by individual POI interactions but also by spatial patterns shared with semantically similar neighbors.

S1.2. Triple-level privacy-preserving GNN construction

To enable private federated training without leaking sensitive user behavior, TPGNN adopts a layered privacy-aware strategy. Each level addresses privacy at a different modeling stage.

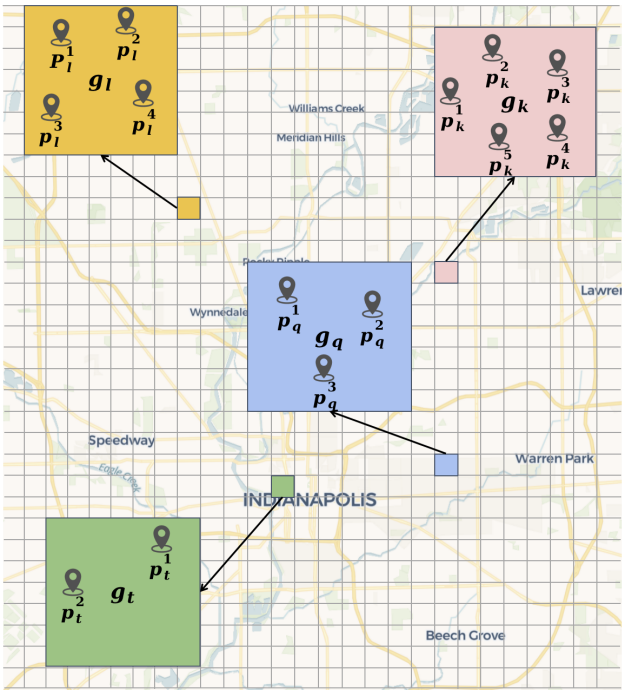


Figure 1: Geographic gridding. A grid display of the area where the POI items are located, where different grids contain different numbers of POI items.

S1.2.1. Level 1: Identity obfuscation via embedding construction

The first level of protection aims to prevent user identity exposure during embedding construction. Instead of using explicit user identifiers, embeddings are derived solely from interaction context. A multi-source attention mechanism integrates information from interacted POIs, geographically neighboring users, and self-representations.

POI-based initialization: We initialize user embeddings without using raw user IDs. Instead, each embedding is derived as the mean of the embeddings of the POIs the user has interacted with, as defined in Eq. (1).

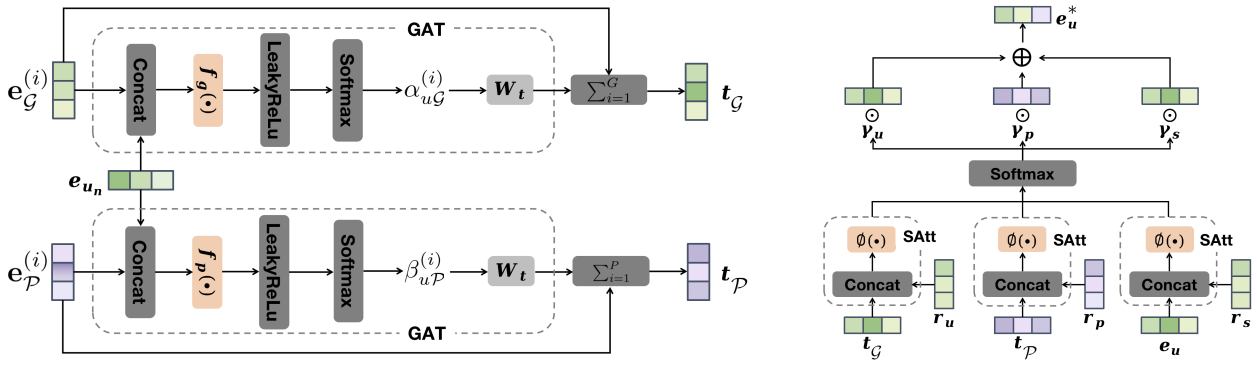
$$\mathbf{e}_u = \frac{1}{m} \sum_{i=1}^{m} \mathbf{e}_{p_i} \tag{1}$$

where $\mathbf{e}_{p_i} \in \mathbb{R}^d$ denotes the embedding of the i-th POI visited by user u, m is the number of visited POIs, and d is the embedding dimension. This initialization captures users'coarse preferences while avoiding the use of identifiable features.

Relation-aware neighborhood enhancement: To improve embedding expressiveness, we introduce two types of semantic neighbors: geographical neighbor users $\mathcal{G}$ and interacted POIs $\mathcal{P}$. Their contributions are modeled through Graph Attention (GAT) based aggregation, as shown in Eq. (2).

$$\begin{cases} \alpha_{u\mathcal{G}}^{(i)} = \frac{\exp(f_g(\mathbf{e}_u, \mathbf{e}_{\mathcal{G}}))}{\sum_{i=1}^{G} \exp(f_g(\mathbf{e}_u, \mathbf{e}_{\mathcal{G}}^{(i)}))}, \\ \beta_{u\mathcal{P}}^{(i)} = \frac{\exp(f_p(\mathbf{e}_u, \mathbf{e}_{\mathcal{P}}))}{\sum_{i=1}^{P} \exp(f_p(\mathbf{e}_u, \mathbf{e}_{\mathcal{P}}^{(i)}))} \end{cases} \tag{2}$$

where f_g and f_p are feedforward attention functions that learn relevance scores between the user and its neighbor users or POIs. G and P denote the number of geographical neighbors and



(a) Relation embedding generation based on GAT

(b) Relation embedding aggregation based on SAtt.

Figure 2: Relation embeddings aggregation. (a) Generate hidden embedding representation of user neighbor embeddings through Graph Attention (GAT) mechanism. (b) Obtain local user embedding by aggregating relation embeddings through the self-attention (SAtt) mechanism.

interacted POIs, respectively. These attention weights are used to compute intermediate relational embeddings as shown in Eq. (3).

$$\mathbf{t}_{\mathcal{G}} = \sum_{i=1}^{G} \alpha_{u\mathcal{G}}^{(i)} W_t \mathbf{e}_{\mathcal{G}}^{(i)}, \quad \mathbf{t}_{\mathcal{P}} = \sum_{i=1}^{P} \beta_{u\mathcal{P}}^{(i)} W_t \mathbf{e}_{\mathcal{P}}^{(i)} \tag{3}$$

where W_t is the linear mapping weight matrix, $\mathbf{t}_{\mathcal{G}}$ and $\mathbf{t}_{\mathcal{P}}$ are the hidden embeddings of neighbor users and neighbor POIs, respectively. Their specific generation process is shown in Fig. 2 (a).

Context-aware fusion with self-attention: We compute the fused user embedding by softly attending to multiple semantic sources: spatial neighbors, interacted POIs, and the user's initial representation. Inspired by the Self-Attention (SAtt) mechanism [2], we introduce three relation semantic vectors—$\mathbf{r}_s$ (self), $\mathbf{r}_u$ (neighbor user), and $\mathbf{r}_p$ (POI interaction)—and compute their relevance as shown in Eq. (4).

$$z_k = \exp(\phi(\mathbf{h}_k, \mathbf{r}_k)), \quad k \in s, u, p \tag{4}$$

where $\phi(\cdot)$ is a dot-product or bilinear scoring function, and $\mathbf{h}_k \in \{\mathbf{e}_u, \mathbf{t}_{\mathcal{G}}, \mathbf{t}_{\mathcal{P}}\}$ represents each semantic component. These scores are normalized to obtain the attention weights, as shown in Eq. (5).

$$\gamma_k = \frac{z_k}{z_s + z_u + z_p}, \quad k \in s, u, p \tag{5}$$

The final embedding is computed as shown in Eq. (6).

$$\mathbf{e}_u^* = \gamma_s \cdot \mathbf{e}_u + \gamma_u \cdot \mathbf{t}_{\mathcal{G}} + \gamma_p \cdot \mathbf{t}_{\mathcal{P}} \tag{6}$$

where $\mathbf{e}_u^*$ is the local user embedding used for prediction and the aggregation process is shown in Fig. 2 (b).

S1.2.2. Level 2: Interaction obfuscation via pseudo-POI selection

Although user embeddings are not directly shared with the server, gradients from local training can still reveal sensitive user-POI interaction patterns. In particular, only interacted POIs generate

non-zero gradients, exposing the model to Membership Inference Attacks (MIAs). To address this, we introduce an interaction obfuscation mechanism that generates plausible pseudo-interactions based on the K-anonymity principle.

K-anonymity based pseudo-POI selection: To conceal real check-in behavior during training, each user locally generates pseudo-POI sets by generalizing interactions within geographic grids. For a real POI p_r located in grid $\mathbb{G}$, $k-1$ additional POIs $\{p_f^1, \ldots, p_f^{k-1}\}$ are randomly selected from the same grid to form a K-anonymous set $A = \{p_r, p_f^1, \ldots, p_f^{k-1}\}$. Repeating this for all M real POIs results in a combined training set, as defined in Eq. (7).

$$RP = \bigcup_{m=1}^{M} A_m \tag{7}$$

where each A_m represents a K-anonymized interaction group for a single POI. This process ensures that no individual POI interaction can be uniquely attributed to the user, thereby providing obfuscation at the interaction level.

Local prediction with mixed interactions: For recommendation, the predicted rating $\hat{R}_{up}$ for a POI p is computed via the inner product of the user's embedding $\mathbf{e}_u^*$ and the POI embedding $\mathbf{e}_p$, as shown in Eq. (8).

$$\hat{R}_{up} = \mathbf{e}_u^* \cdot \mathbf{e}_p \tag{8}$$

The loss for each user is defined as the Root Mean Square Error (RMSE) between predicted and ground-truth ratings, computed over both real and pseudo interactions, as shown in Eq. (9).

$$\mathcal{L}_u = \sqrt{\frac{\sum_{x \in RP^{(u)}} (R_{up} - \hat{R}_{up})^2}{\left|RP^{(u)}\right|}} \tag{9}$$

where $RP^{(u)}$ denotes the set of real and pseudo POIs for user u. For pseudo-POIs, the ground-truth rating R_{up} is set as the rounded prediction $\hat{R}_{up}$, introducing controlled randomness that acts as noise injection and further protects the true interaction signals [3].

S1.2.3. Level 3: Personalized gradient perturbation via hierarchical differential privacy

To improve resistance to model inversion attacks while minimizing accuracy loss, TPGNN adopts a personalized gradient perturbation strategy as its third privacy layer. The key idea is to calibrate differential privacy noise based on each user's data sensitivity, quantified using a Hasse diagram-based hierarchy. In contrast, standard DP mechanisms apply a uniform noise scale to all clients, as shown in Eq. (10).

$$\tilde{\mathbf{g}} = \text{clip}(\mathbf{g}, \delta) + \text{Laplace}\left(0, \frac{\Delta f}{\varepsilon}\right) \tag{10}$$

where $\mathbf{g}$ is the client gradient, δ is a clipping threshold, Δf is the global sensitivity, and ε is the privacy budget. However, this approach fails to account for variations in user behavior–some clients contribute far more information than others, which may lead to disproportionate privacy leakage.

Sensitivity modeling with hasse diagram: To enable personalized noise injection, we propose a structure-aware sensitivity modeling framework based on partial order theory. Instead of estimating sensitivity from raw interaction volume, we construct a Hasse diagram to capture subset inclusion relationships between user behavior profiles.

Specifically, Each user's POI interaction set is abstracted as a subset of the global item space, and pairwise comparisons are conducted to identify strict inclusion relations, denoted as $u_i \supset u_j$,

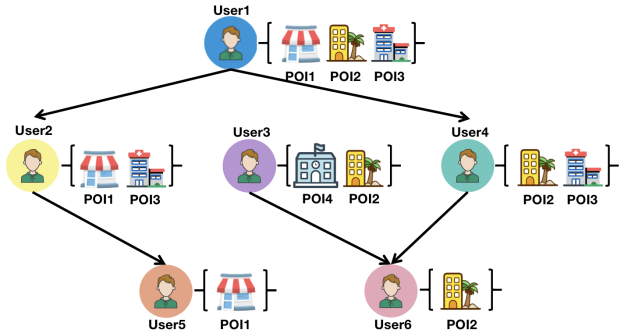


Figure 3: Hasse diagram of partial order relationship of user interaction items. User icons of different colors are located at different levels. $\{\cdot\}$ represents the user's interaction items set. The bottom user has fewer interaction items, while the upper user has more interaction items. The edge represents the inclusion relationship of the interaction item set from top to bottom.

where all elements in u_j's set are contained in u_i's set, and no intermediate subset exists between them. These relations are used to construct a Hasse diagram–a directed acyclic graph representing the partial order structure. In this graph, nodes represent users, and directed edges indicate dominance in behavior patterns, revealing hierarchical exposure levels. An illustrative example is shown in Fig. 3.

To preserve privacy during graph construction, each client locally encodes its POI set into a privacy-preserving behavioral representation, such as a Bloom filter or category count vector. These representations are uploaded to the server, which performs approximate set containment tests to infer inclusion relations without accessing the original interaction records.

We define the sensitivity of each user u_i as the number of users whose behaviors are strictly contained within theirs, reflecting how many individuals are behaviorally "covered" by u_i. We compute a normalized sensitivity score as defined in Eq. (11).

$$S_u^i = \frac{|u_j \in U \mid u_i \supset u_j|}{\sum_{n=1}^{N} |u_j \in U \mid u_k \supset u_j|} \tag{11}$$

where U denotes the set of all users and $|\cdot|$ represents the cardinality of the set. The numerator counts the number of users behaviorally dominated by u_i, while the denominator computes the total number of such dominance relations across all users, ensuring that the resulting sensitivity score $S_u^i \in (0, 1)$ is normalized.

This formulation exploits the partial order nature of user behavior and provides a graph-theoretic basis for measuring privacy risk in a personalized manner. By avoiding dependence on raw POI disclosure and enabling structural differentiation between users, the proposed sensitivity modeling strategy supports fair and fine-grained noise calibration for privacy-preserving federated optimization.

Gradient upload with personalized DP: For the federated recommendation model designed in this paper, the clients mainly upload three types of gradients to the server, namely, real POI embedding gradient, pseudo POI embedding gradient and model gradient, and their instantiations are $\mathrm{g}_r^{(n)}$, $\mathrm{g}_p^{(n)}$, $\mathrm{g}_m^{(n)}$ respectively. Combining these gradients, the total client upload gradient is $\mathrm{g}^{(n)} = \{\mathrm{g}_r^{(n)}, \mathrm{g}_p^{(n)}, \mathrm{g}_m^{(n)}\} = \frac{\partial \mathcal{L}_u}{\partial \theta_{\text{global}}}$, where θ_{global} represents all trainable parameters.

Since different users have different gradient sensitivities when adding Laplace noise to uploaded gradients for differential privacy protection, it is necessary to design a weight $w = 1 - S_u$ to adjust

the size of the privacy budget ε. Therefore, according to Eq. (10), the noise perturbation method based on gradient sensitivity is shown in Eq. (12).

$$\tilde{\mathrm{g}}^{(n)} = clip(\mathrm{g}^{(n)}, \delta) + Laplacian(0, \frac{\Delta f}{w^{(n)} \cdot \varepsilon}) \tag{12}$$

Differential privacy is a widely used method to protect user privacy. It will be proved in S3.2 that this noise perturbation method based on gradient sensitivity satisfies ε-differential privacy. This hierarchical scheme enables fine-grained control over privacy strength: users with richer data contribute more to the model but receive stronger perturbation, while low-activity users are less disturbed, preserving learning stability. Compared to flat noise injection [4], this approach achieves stronger privacy without sacrificing recommendation quality.

S1.3. Federated optimization with fuzzy client selection

To ensure scalable and communication-efficient model training, TPGNN-FedGPR employs a two-stage federated optimization strategy, consisting of client selection via fuzzy clustering and privacy-aware gradient aggregation for model updates.

S1.3.1. Client selection via fuzzy clustering

In contrast to conventional random sampling approaches commonly used in federated recommendation systems [5, 6], we implement a fuzzy clustering-based client selection mechanism to improve training stability and reduce communication overhead. Specifically, we use a soft clustering algorithm to group clients based on similarities in their local embeddings.

During each communication round, one or more clients are sampled from each cluster by the server, with their selection probabilities guided by the clients' fuzzy membership degrees. This strategy helps ensure that the chosen clients are not only behaviorally diverse but also statistically representative, thereby promoting faster convergence and improved generalization performance.

To prevent exposure of sensitive data during the clustering process, local user embeddings used for similarity computation are not uploaded to the server. Instead, peer-to-peer (P2P) exchanges are conducted among clients to derive local embedding relationships. Each client then computes a local fuzzy membership matrix, which is transmitted to the server. The server aggregates these matrices and performs client sampling based on aggregated membership information.

S1.3.2. Gradient aggregation for model optimization

The server in TPGNN-FedGPR collects client-uploaded gradients to update model parameters and embeddings, enabling collaborative optimization. Each client uploads gradients $\mathrm{g}^{(n)} = \{\mathrm{g}_r^{(n)}, \mathrm{g}_p^{(n)}, \mathrm{g}_m^{(n)}\}$, representing real item, pseudo item, and model parameter gradients, respectively. In each round, a batch of clients (e.g., 128), denoted by $\mathcal{M}$, is selected. The server sends the current global parameters θ_m and item embeddings $E_{r,p}$ to these clients, then aggregates their local gradients based on the new updates, as defined in Eq. (13).

$$\begin{cases} \bar{\mathrm{g}}_m = \frac{\sum_{n \in \mathcal{M}} |\mathcal{S}_n| \cdot \tilde{\mathrm{g}}_m^{(n)}}{\sum_{n \in \mathcal{M}} |\mathcal{S}_n|}, \\ \bar{\mathrm{g}}_{r,p} = \frac{\sum_{n \in \mathcal{M}} |\mathcal{S}_n| \cdot (\tilde{\mathrm{g}}_r^{(n)} || \tilde{\mathrm{g}}_p^{(n)})}{\sum_{n \in \mathcal{M}} |\mathcal{S}_n|} \end{cases} \tag{13}$$

where $\mathcal{S}_n$ denotes the set of items associated with client n, including both real and pseudo interactions, and is used to weight the client's gradient contribution. Since the server is unable to

distinguish between real and pseudo item gradients, both components are aggregated jointly to update item embeddings. After such average aggregation, the server updates the parameter θ_{global} using gradient descent by Eq. (14).

$$\theta^*_{\text{global}} = \theta_{\text{global}} - \eta \cdot \bar{\text{g}} \tag{14}$$

where η denotes the learning rate. Through iterative training, the model gradually converges to the optimal global parameter θ^*_{global}.

S2. Embedding alignment under privacy constraints

In TPGNN-FedGPR, embedding alignment is achieved **implicitly through federated optimization**, rather than through any explicit sharing of intermediate representations. To ensure privacy and avoid sensitive information leakage, we adopt the following design:

(1) No raw embedding vectors or user identifiers are transmitted to the server or shared across clients.

(2) Alignment occurs via client-uploaded gradients, which are subjected to **personalized differential privacy**.

(3) The attention-based embedding fusion operates locally per user, incorporating POI-level and neighbor-level context without requiring global alignment.

The embedding space is aligned via DP-SGD-based updates, and no centralized user-POI graph is maintained. Clients operate on locally induced graphs, ensuring privacy protection and decentralization.

S3. Theoretical analysis

This section provides a unified theoretical analysis of TPGNN-FedGPR from three perspectives: the privacy-utility trade-off introduced by its triple-level protection, the formal ε-differential privacy guarantee of the personalized gradient perturbation, and the system's time complexity in federated settings. Although multiple privacy mechanisms are integrated, their effects on model utility and efficiency remain significant. A layer-wise utility analysis is thus presented, followed by a formal privacy proof and a complexity analysis to demonstrate the framework's scalability and practical viability.

S3.1. Privacy-utility trade-off in triple-level protection

The triple-level privacy design in TPGNN-FedGPR addresses distinct privacy threats at the identity, interaction, and gradient levels. Each layer enhances privacy through specific transformations, which can affect the learning process. We analyze the impact of each layer on utility below and describe the corresponding mitigation strategies.

Level 1: Embedding-based identity obfuscation

To prevent identity leakage, user embeddings are constructed without raw identifiers. Instead, they are initialized from POI interactions and enriched by aggregating information from semantic neighbors.

Privacy benefit: Prevents exposure of explicit user IDs in the training process.

Utility cost: Reduces representation specificity due to the removal of personalized ID features.

Utility mitigation strategy: Embedding expressiveness is preserved through multi-source attention mechanisms that integrate contextual signals from interacted POIs and geographically similar users. This enhances the semantic richness of user representations, thereby compensating for the loss of identifier-based personalization.

Level 2: K-anonymity based interaction obfuscation

To defend against membership inference attacks, each real POI interaction is augmented with $k-1$ pseudo-POIs sampled from the same geographic grid. These pseudo-interactions are incorporated into training alongside real ones.

Privacy benefit: Obscures precise user behavior by blending real and plausible fake check-ins.

Utility cost: Introduces noise into the training data, potentially affecting prediction accuracy.

Utility mitigation strategy: Semantic consistency is preserved by sampling pseudo-POIs from the same spatial grid as the real ones, minimizing feature shift. Additionally, surrogate labels are generated via self-predicted ratings, reducing reliance on noisy or externally provided annotations.

Level 3: Personalized gradient perturbation

To enhance protection against gradient inversion attacks, Laplace noise is injected into client-uploaded gradients. Rather than using a uniform noise scale, a personalized approach is adopted.

Privacy benefit: Shields gradient information and reduces the risk of model inversion.

Utility cost: May hinder convergence and impair learning due to excessive noise in sensitive cases.

Utility mitigation strategy: A Hasse diagram-based sensitivity model is employed to assess each user's behavioral coverage, from which a personalized sensitivity score is derived. The noise scale is adjusted accordingly: users with broader behavioral profiles receive stronger perturbation, while low-risk users retain gradient fidelity. This adaptive scheme ensures a fine-grained privacy-utility balance superior to conventional uniform differential privacy.

S3.2. ε-Differential privacy guarantee

Differential privacy is a widely adopted standard for quantifying and enforcing privacy guarantees in machine learning systems. In the TPGNN-FedGPR framework, the third layer of the privacy mechanism introduces a personalized gradient perturbation strategy based on a sensitivity-aware Laplace mechanism. In this section, it is formally shown that this mechanism satisfies ε-differential privacy.

Theorem 1. The personalized gradient perturbation mechanism based on user-specific sensitivity scores satisfies ε-differential privacy.

Proof. Let Δf denote the global sensitivity of the gradient function, and let $w^{(n)} \in (0, 1)$ denote the scaling weight associated with user n, derived from their sensitivity score. The noise scale of the Laplace mechanism is defined as:

$$\lambda = \frac{\Delta f}{w^{(n)} \cdot \varepsilon} \tag{15}$$

Consider two adjacent datasets D and D', differing by a single interaction item, such that $D = D' \cup \{p\}$. Let the gradients computed over these datasets be:

$$\nabla_\theta \mathcal{L}_D(\theta) = \frac{\partial \mathcal{L}_u(D)}{\partial \theta}, \quad \nabla_\theta \mathcal{L}_{D'}(\theta) = \frac{\partial \mathcal{L}_u(D')}{\partial \theta} \tag{16}$$

To satisfy differential privacy, Laplace noise $x \sim \text{Lap}(0, \lambda)$ is added to each gradient:

$$\mathcal{A}(D) = \nabla_\theta \mathcal{L}_D(\theta) + x, \quad \mathcal{A}(D') = \nabla_\theta \mathcal{L}_{D'}(\theta) + x \tag{17}$$

For any possible output value a in the range of $\mathcal{A}(\cdot)$, the probability densities of observing a from I and I' are given by:

$$\begin{aligned} Pr[\mathcal{A}(D) = a \in S] &= Pr[Laplacian(0, \lambda) = x] \\ &= \frac{1}{2\lambda} \exp\left(-\frac{|x|}{\lambda}\right) \\ &= \frac{1}{2\lambda} \exp\left(-\frac{|a - \nabla\theta\mathcal{L}_D(\theta)|}{\lambda}\right) \end{aligned} \tag{18}$$

$$\begin{aligned} Pr[\mathcal{A}(D') = a \in S] &= Pr[Laplacian(0, \lambda) = x] \\ &= \frac{1}{2\lambda} \exp\left(-\frac{|x|}{\lambda}\right) \\ &= \frac{1}{2\lambda} \exp\left(-\frac{|a - \nabla\theta\mathcal{L}_{D'}(\theta)|}{\lambda}\right) \end{aligned} \tag{19}$$

Taking the ratio of the two probabilities yields:

$$\begin{aligned} \frac{Pr[\mathcal{A}(D) \in S]}{Pr[\mathcal{A}(D') \in S]} &= \frac{\exp\left(-\frac{|a - \nabla\theta\mathcal{L}_D(\theta)|}{\lambda}\right)}{\exp\left(-\frac{|a - \nabla\theta\mathcal{L}_{D'}(\theta)|}{\lambda}\right)} \\ &= \exp\left(\frac{|a - \nabla\theta\mathcal{L}_{D'}(\theta)| - |a - \nabla\theta\mathcal{L}_D(\theta)|}{\lambda}\right) \\ &= \exp\left(\frac{|a - \nabla\theta\mathcal{L}_{D'}(\theta)| - |a - \nabla\theta\mathcal{L}_D(\theta)|}{\frac{\Delta f}{w^{(n)} \cdot \varepsilon}}\right) \end{aligned} \tag{20}$$

Since the gradient function satisfies the global sensitivity constraint $|\nabla\theta\mathcal{L}_D(\theta) - \nabla\theta\mathcal{L}_{D'}(\theta)| \leq \Delta f$, and $0 < w^{(n)} < 1$, it follows that:

$$\begin{aligned} \frac{Pr[\mathcal{A}(D) \in S]}{Pr[\mathcal{A}(D') \in S]} &= \exp\left(\frac{|\nabla\theta\mathcal{L}_D(\theta) - \nabla\theta\mathcal{L}_{D'}(\theta)|}{\frac{\Delta f}{w^{(n)} \cdot \varepsilon}}\right) \\ &\leq \exp\left(\frac{\Delta f}{\frac{\Delta f}{w^{(n)} \cdot \varepsilon}}\right) = e^{w^{(n)} \cdot \varepsilon} \\ &\leq e^{\varepsilon} \end{aligned} \tag{21}$$

Therefore, the personalized Laplace mechanism is proven to satisfy ε-differential privacy.

This result confirms that the sensitivity-weighted noise injection strategy used in TPGNN-FedGPR provides formal differential privacy guarantees during federated gradient sharing. Compared to uniform noise injection, this personalized mechanism adapts the noise scale based on individual user sensitivity, offering stronger protection for high-risk users while preserving utility for those with lower exposure.

S3.3. Time complexity analysis

We analyze the time complexity of TPGNN-FedGPR with respect to both client-side and server-side operations within a single communication round. The analysis is divided into three primary stages: user embedding construction, graph convolution propagation, and model update. Let d denote the embedding dimensionality, L the number of GNN layers, $\mathcal{I}_u$ the set of real POIs for user u, and $\mathcal{I}'_u$ the corresponding set of pseudo-POIs.

Embedding construction stage: For each client $u \in U$, the initial representation is computed based on interacted POIs and spatial neighbors, with a time complexity of $\mathcal{O}(|\mathcal{I}_u| \cdot d)$.

Graph convolution stage:, Neighborhood aggregation is performed over both real and pseudo POIs. For each client, the computation incurs a complexity of $\mathcal{O}((|\mathcal{I}_u| + |\mathcal{I}'_u|) \cdot dL)$. On the server side, the aggregation of representations across all users results in a total complexity of $\mathcal{O}\left(\sum_{u=1}^{n}(|\mathcal{I}_u| + |\mathcal{I}'_u|) \cdot dL\right)$, where n denotes the number of participating users.

Model update stage: Local gradients are computed by each client based on both real and pseudo interactions, incurring a complexity of $\mathcal{O}((|\mathcal{I}_u| + |\mathcal{I}'_u|) \cdot d)$. Subsequently, these gradients are aggregated at the server, leading to an additional complexity of $\mathcal{O}\left(\sum_{u=1}^{n}(|\mathcal{I}_u| + |\mathcal{I}'_u|) \cdot d\right)$.

The per-iteration computational cost for each client is primarily determined by the graph convolution phase, resulting in a complexity of $\mathcal{O}((|\mathcal{I}_u| + |\mathcal{I}u'|) \cdot dL)$. On the server side, each communication round involves aggregating user embeddings, with a complexity of $\mathcal{O}\left(\sum_{u=1}^{n}(|\mathcal{I}_u| + |\mathcal{I}u'|) \cdot dL\right)$, and combining gradients, with a corresponding complexity of $\mathcal{O}\left(\sum_{u=1}^{n}(|\mathcal{I}_u| + |\mathcal{I}'_u|) \cdot d\right)$.

Efficiency improvement via fuzzy client selection. To enhance scalability, a fuzzy clustering-based client selection strategy is adopted instead of conventional random sampling. Clients are grouped based on semantic similarity, and only a subset of representative clients, denoted as $\mathcal{C}$, is selected per communication round, where $|\mathcal{C}| \ll n$.

Through this mechanism, the server-side computational and communication overhead is reduced from $\mathcal{O}(n)$ to $\mathcal{O}(|\mathcal{C}|)$. Consequently, the complexities associated with embedding aggregation and gradient aggregation are reduced to $\mathcal{O}(\sum_{u\in\mathcal{C}}(|\mathcal{I}_u| + |\mathcal{I}'_u|) \cdot dL)$ and $\mathcal{O}\left(\sum_{u\in\mathcal{C}}(|\mathcal{I}_u| + |\mathcal{I}'_u|) \cdot d\right)$, respectively.

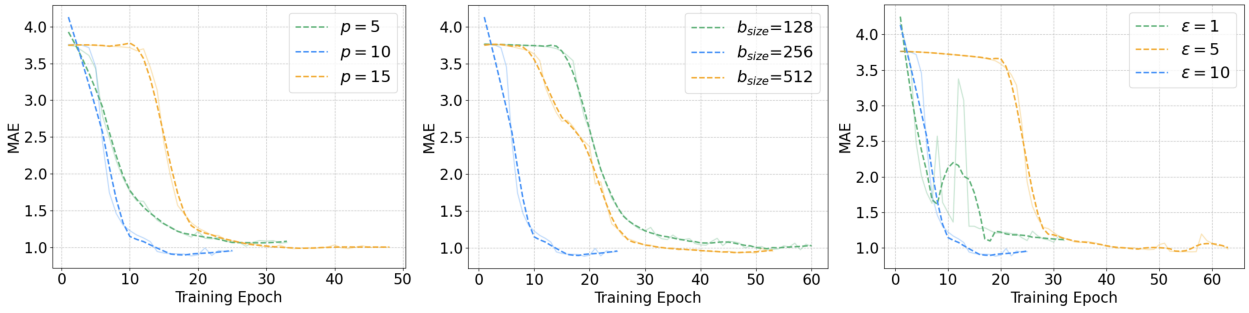
S4. Hyperparameters analysis

We determine the optimal hyperparameter configuration for FedGPR through a series of controlled experiments, adjusting the number of pseudo-interacted items (p), the user batch size (b_{size}), and the privacy budget (ε). In each experiment, we vary only one hyperparameter while keeping the others constant. The resulting effects on MAE, RMSE, MI, and convergence time are analyzed to evaluate both performance and the privacy-utility trade-offs.

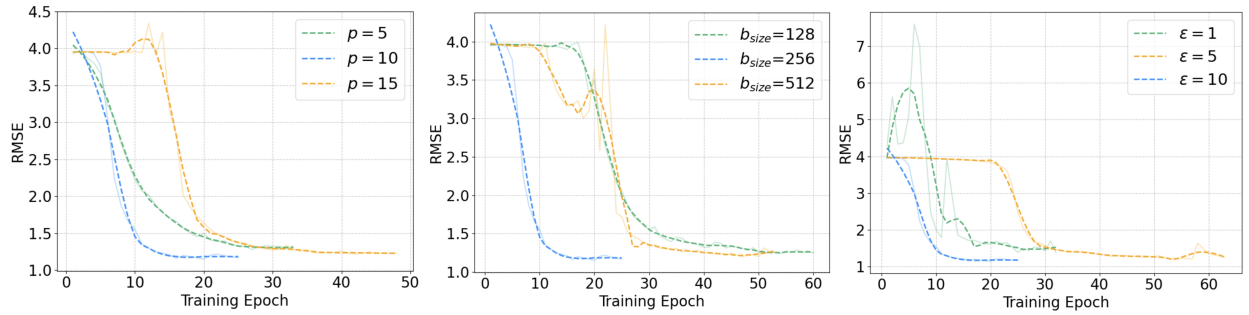
As shown in Fig. 4, the impact of different hyperparameter settings on model convergence and predictive accuracy is visualized over training epochs. Each subplot corresponds to one hyperparameter–p, b_{size}, or ε–with three representative values evaluated for each setting to illustrate performance variation.

Figs. 4(a) and 4(d) show that setting $p = 10$ provides a favorable balance between convergence speed and final accuracy. A smaller value ($p = 5$) results in slower convergence and higher error, while increasing p to 15 introduces slight degradation in early-stage performance–likely due to redundancy or overfitting introduced by excessive pseudo-interactions.

Figs. 4(b) and 4(e) analyze the effect of batch size. A configuration of $b_{\text{size}} = 256$ achieves the most stable and rapid convergence, outperforming both smaller (128) and larger (512) batch sizes. The results indicate that this setting offers an effective compromise between generalization and training efficiency.

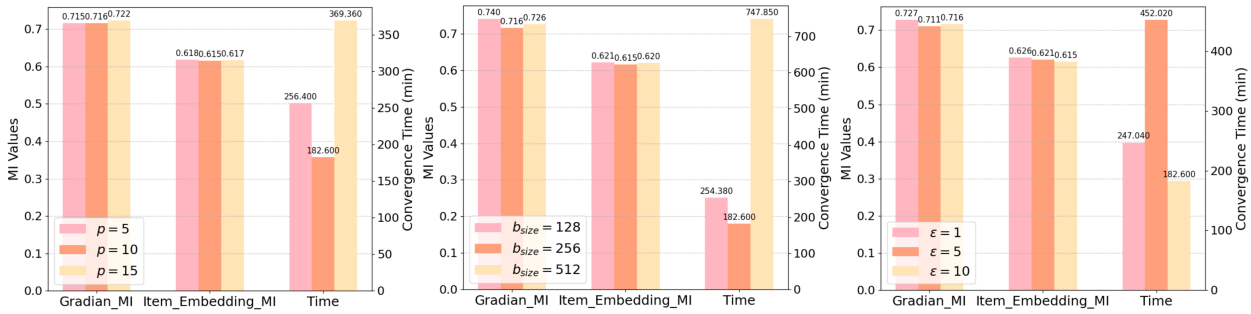


(a) MAE of different p (b) MAE of different b_{size} (c) MAE of different ε



(d) RMSE of different p (e) RMSE of different b_{size} (f) RMSE of different ε

Figure 4: Comparison of effects of different hyperparameter settings on MAE and RMSE.



(a) MI and time of different p (b) MI and time of different b_{size} (c) MI and time of different ε

Figure 5: Comparison of effects of different hyperparameter settings on MI and convergence time.

Figs. 4(c) and 4(f) illustrate the influence of the privacy budget. While stronger privacy ($\varepsilon = 1$) guarantees lead to higher error and less stable convergence, a moderate budget ($\varepsilon = 10$) substantially improves accuracy and convergence speed. This suggests that excessive noise under low ε values impairs learning, whereas $\varepsilon = 10$ retains adequate utility without severely compromising privacy.

To assess the trade-offs between privacy and training efficiency, Fig. 5 presents the MI and convergence time under varying hyperparameter configurations. Gradient-level and item embedding-level MI are used to evaluate potential privacy leakage, while convergence time serves as a proxy for computational cost.

As shown in Fig. 5(a), varying the number of pseudo items p has minimal impact on privacy leakage, with gradient MI values ranging from 0.715 to 0.722 and item embedding MI remaining close to 0.617. In terms of convergence time, an intermediate value ($p = 10$) yields the fastest

Table 1: Summary of notations encountered frequently.

Notation	Description
$\mathcal{R}$	Entire area
$\mathbb{L}$	Number of grids
$\mathbb{M}$	Number of POIs contained in the grid
$\mathbb{G}$	A grid containing a set of POIs
N	Total number of users
u_n	User n
p	POI p
p_f	Pseudo POI
$T_{\mathbb{G}}$	Threshold for the number of grids
$\mathbf{e}_u$	User embeddings
$\mathbf{e}_{pi}$	Embedding of the i-th POI
d	Embedding dimension
$\mathcal{G}$	Geographical neighbor users
$\mathcal{P}$	Interacted POIs
f_g/f_p	Feedforward attention functions
G	Number of geographical neighbors
P	Number of interacted POIs
W_t	Linear mapping weight matrix
$\mathbf{r}$	Relation semantic vectors
A	A K-anonymous set
PR	Real POI and Pseudo-POI Sets
$\hat{R}_{up}$	Predicted rating
$\mathcal{L}_i$	Local loss on user u_i
Δf	Global sensitivity
ε	Privacy budget
U	A set of all users
S_u^i	Normalized sensitivity score
$g_r^{(n)}$	Real POI embedding gradient
$g_p^{(n)}$	Pseudo-POI embedding gradient
$g_m^{(n)}$	Model parameters gradient
$\mathbf{g}^{(n)}$	All the uploaded gradients of client n
θ_{global}	Global model parameters
$\mathcal{M}$	A batch of clients
$w^{(n)}$	Scaling weight
λ	Noise scale
D, D'	Adjacent datasets
$\Pr[\cdot]$	Probability of privacy leakage
$\mathcal{I}_u/\mathcal{I}'_u$	A set of real / pseudo-POIs
L	The number of GNN layers
$\mathcal{O}$	Time complexity
$\mathcal{C}$	A subset of representative clients

convergence at 182.6 minutes, whereas both lower ($p = 5$) and higher ($p = 15$) values result in longer training times (256.4 and 369.36 minutes, respectively). This indicates that a moderate

number of pseudo items is key to balancing convergence speed and privacy, as both too small and too large p values increase training time without improving privacy.

In Fig. 5(b), the effect of batch size b_{size} is evaluated. As batch size increases from 128 to 256, convergence time decreases significantly from 254.38 to 182.6 minutes, while MI values remain relatively stable. However, further increasing b_{size} to 512 leads to a substantial rise in convergence time (747.85 minutes) and a slight increase in gradient MI, suggesting that overly large batch sizes may hinder training efficiency without improving privacy protection.

Fig. 5(c) confirms that the privacy budget ε directly influences MI. A smaller value, such as $\varepsilon = 5$, reduces gradient-level MI to 0.711 but results in a longer training time of 452.02 minutes. In contrast, $\varepsilon = 10$ yields a slightly higher gradient MI, yet achieves the lowest item embedding MI (0.615) and the shortest training duration (182.6 minutes), suggesting a more effective trade-off between privacy protection and computational efficiency.

In summary, the configuration $p = 10$, $b_{\text{size}} = 256$, and $\varepsilon = 10$ delivers strong predictive performance, low privacy leakage, and minimal convergence time. making it the most practical setting in terms of both scalability and privacy-utility trade-off.

S5. Standardized symbols

The standardized results of symbols used in the paper are shown in Table 1.

References

[1] Huang J, Tong Z, Feng Z. Geographical poi recommendation for internet of things: A federated learning approach using matrix factorization. International Journal of Communication Systems, 2022, e5161

[2] Liu Z, Yang L, Fan Z, Peng H, Yu P S. Federated social recommendation with graph neural network. ACM Transactions on Intelligent Systems and Technology (TIST), 2022, 13(4): 1–24

[3] Liu Z, Fan Z, Wang Y, Yu P S. Augmenting sequential recommendation with pseudo-prior items via reversely pre-training transformer. In: Proceedings of the 44th international ACM SIGIR conference on Research and development in information retrieval. 2021, 1608–1612

[4] Wu C, Wu F, Cao Y, Huang Y, Xie X. Fedgnn: Federated graph neural network for privacy-preserving recommendation. arXiv preprint arXiv:2102.04925, 2021

[5] Hu P, Lin Z, Pan W, Yang Q, Peng X, Ming Z. Privacy-preserving graph convolution network for federated item recommendation. Artificial Intelligence, 2023, 324: 103996

[6] Soltani B, Haghighi V, Mahmood A, Sheng Q Z, Yao L. A survey on participant selection for federated learning in mobile networks. In: Proceedings of the 17th ACM Workshop on Mobility in the Evolving Internet Architecture. 2022, 19–24