Federated Learning-Driven Smart Catering Demand Forecasting And User Consumption Data Privacy Protection

Gang Wei

*Corresponding author. E-mail: 13809033369@163.com

Received July 21, 2026; accepted August 17, 2026

Restaurant demand forecasting underpins inventory planning, staff scheduling, and spoilage control, yet order, membership, delivery, and payment data are scattered across stores and platforms, so centralized modeling risks exposing consumer behavior. This study proposes SPFed-RDF, a scalable, privacy-preserving federated learning framework for collaborative demand forecasting without sharing raw data. Each node performs local feature-processing (imputation, normalization, temporal encoding), and a BiGRU-Attention model extracts temporal demand features; during training, gradient clipping, differential-privacy noise, Top-k compression, secure aggregation, and dynamic weighted aggregation (by sample size, error, and communication stability) jointly address non-IID heterogeneity, privacy leakage, and communication overhead. Results show SPFed-RDF achieves RMSE 20.91, beating ARIMA, SVR, XGBoost, Local-BiGRU, FedAvg, FedProx, and FedAvg+DP by 18.35% versus FedAvg (23.44% under severe non-IID). With 100 clients, Top-k compression cuts communication from 1280 MB to 752 MB (41.27%), and member-inference attack success drops from 78.46% to 44.84%.

Keywords: federated learning, restaurant demand forecasting, consumer data privacy, differential privacy, secure aggregation

http://dx.doi.org/10.6180/jase.202612_35.008

1. Introduction

Digital restaurant services are shifting from experience-driven to data-driven approaches, generating order, sales, member, payment, delivery, and review data valuable for demand forecasting across time, weather, and promotions. ML and deep learning improve sales and traffic prediction [1–3], but restaurant data is inherently distributed: stores, takeout platforms, payment systems, and headquarters each see only part of the picture, and centrally uploading it increases privacy and trade-secret risk.

Restaurant data is inherently distributed with sensitive profiles based on consumption frequency, food preference, and location, so a single unified dataset does not exist in practice without cross-institution data sharing, which raises compliance and competitive concerns.

Restaurant demand forecasting is thus a scalable distributed data-mining problem, not a single-point time-series problem. Real scenarios face data silos, privacy compliance, and communication constraints; user orders contain sensitive profiles that can be re-identified even after removing names and phone numbers. Model training should follow data minimization and local retention principles.

To address these issues, this study proposes SPFed-RDF, a scalable privacy-preserving federated restaurant demand prediction framework for distributed systems spanning multiple stores, platforms, and terminals. It uses a local BiGRU-Attention model for temporal demand features, dynamic weighted federated aggregation to mitigate non-IID impact, and gradient clipping, differential privacy, secure aggregation, and Top-k communication compression for privacy protection and communication efficiency.

Contributions: a scalable federated learning framework transforming multi-store demand prediction into a privacy-

preserving distributed data-mining problem; a BiGRU-Attention local model integrating historical demand, consumption behavior, time, weather, promotions, and platform traffic to characterize demand periodicity and drivers; a dynamic weighted aggregation method based on sample size, local error, and communication stability for non-IID stability; differential privacy, secure aggregation, and Top-k compression to reduce communication overhead while protecting consumption data; and simulation verification across algorithm performance, privacy attacks, communication scalability, client disconnection, and t-SNE feature distribution. SPFed-RDF's novelty is integrating temporal demand modeling, reliability-aware aggregation, differential privacy, secure aggregation, and sparse communication as one end-to-end workflow rather than isolated components.

2. Methods

2.1. System Model and Problem Definition

2.1.1. Distributed Catering Information System Scenario

This study considers a distributed catering information system composed of multiple catering store nodes, online ordering platform nodes, a membership behavior system, and a federated aggregation server. Let the set of catering nodes participating in the training be denoted as.

$$\mathcal{C} = \{C_1, C_2, \ldots, C_N\}, \tag{1}$$

Where C_i represents the i catering node, and N is the number of nodes participating in federated training. Node C_i stores its private dataset locally.

$$D_i = \{(x_i^t, y_i^t)\}_{t=1}^{T_i}, \tag{2}$$

Where x_i^t represents the multi-source input features of the i node at time t, y_i^t represents the catering demand at the corresponding time or within the prediction window, and T_i is the number of local samples. The global dataset can be formalized as

$$D = \bigcup_{i=1}^{N} D_i, \tag{3}$$

However, in the actual training process, D_i is always kept locally on the node, and the central server cannot directly access any original consumption records. The catering demand prediction task can be represented as learning the prediction function given the historical window length l and the prediction step size h.

$$\hat{y}_i^{t+h} = f_\theta\left(x_i^{t-l+1}, x_i^{t-l+2}, \ldots, x_i^t\right), \tag{4}$$

Where θ represents the model parameters, and $\hat{y}_i^{t+h}$ represents the prediction requirement of node C_i in the next h steps. Unlike single-store prediction, federated prediction requires learning a global model that can generalize to multiple restaurant nodes while protecting local data.

2.1.2. Local Feature Composition

Table 1 presents the local feature composition of the restaurant demand prediction node. As shown in Table 1, consumer behavior features and platform features are highly sensitive to privacy concerns. If this data were centrally uploaded, it could potentially leak user dietary preferences, spending power, consumption time, and store operation strategies. Therefore, this study employs a federated learning mechanism to ensure that these features only participate in model training locally.

2.1.3. Privacy Threat Model

This study considers four privacy/security threats: an honest-but-curious server may try to infer local consumption information from uploaded gradients; malicious nodes may infer other stores' demand distribution by observing global model changes; external attackers may eavesdrop and reconstruct local samples via parameter recovery; and post-training member-inference attacks may determine whether a specific record was in the training data. Existing research notes that retaining raw data locally alone is not equivalent to complete privacy protection [4–6]. The threat model assumes authenticated channels, a semi-honest (honest-but-curious) server, and collusion bounded below the secret-sharing reconstruction threshold used in secure aggregation (Section 2.2.4); it allows the server, malicious clients, and external observers to exploit model updates within these bounds, but an actively malicious or majority-colluding server is out of scope; poisoning, backdoor, and Byzantine attacks are treated as out-of-scope, revisited as future robustness extensions.

2.1.4. Optimization Objective

The global optimization objective of federated restaurant demand prediction is defined as follows:

$$\min_\theta F(\theta) = \sum_{i=1}^{N} \frac{|D_i|}{|D|} F_i(\theta), \tag{5}$$

Wherein, the local objective function is

$$F_i(\theta) = \frac{1}{|D_i|} \sum_{(x,y)\in D_i} \mathcal{L}\left(f_\theta(x), y\right). \tag{6}$$

To simultaneously constrain large and general errors, this study employs a combined loss of squared error and absolute error:

Table 1. Local Restaurant Demand Prediction Feature Composition

Feature categories	Feature Examples	Data source	Privacy sensitivity
Historical demand characteristics	Hourly order volume, daily sales revenue, food sales volume, and number of preorders.	POS System	Medium
Consumer behavior characteristics	Purchase frequency, average order value, preferred dishes, coupon usage	Membership System	High
Time characteristics	Hour, day of the week, weekend, holiday, mealtime	Calendar System	Low
Weather characteristics	Temperature, rainfall, wind speed, weather type	Public Weather Interface	Low
Promotional characteristics	Discounts, spending-to-refund, group buying, platform subsidies	Marketing System	Medium
Platform characteristics	Food delivery percentage, delivery delays, platform exposure, cancellation rate	Food Delivery Platform	High
Store context characteristics	Area, business district type, number of seats, business hours	Store Management System	Medium

$$\mathcal{L}\left(f_\theta(x), y\right) = \alpha\left(y - \widehat{y}\right)^2 + \beta\left|y - \widehat{y}\right|, \tag{7}$$

Where α and β are balance coefficients. When catering demand fluctuates significantly during holidays or promotional periods, the squared error term penalizes larger deviations, while the absolute error term improves the model's robustness to abnormal fluctuations.

2.2. SPFed-RDF Framework Design

2.2.1. *Overall Architecture*

SPFed-RDF consists of a distributed data layer, local feature processing layer, local demand prediction layer, privacy-preserving federated training layer, and scalable prediction service layer. Figure 1 illustrates the overall framework structure.

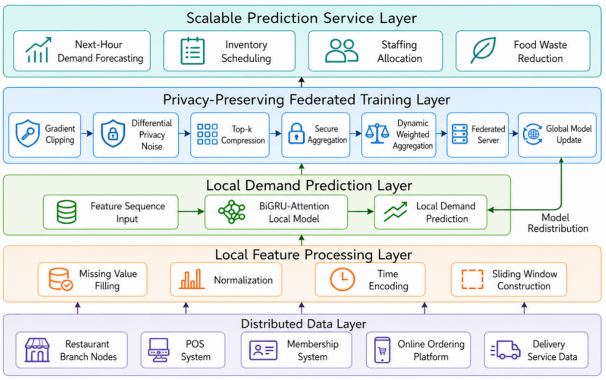

Fig. 1. SPFed-RDF Scalable Privacy-Preserving Federated Restaurant Demand Forecasting Framework

As shown in Figure 1, each restaurant node completes data cleaning, time-window construction, and normalization locally, then trains a local BiGRU-Attention model. Nodes upload model updates — after gradient clipping, DP perturbation, and Top-k compression — rather than raw order data. The central server obtains the aggregated update via secure aggregation, updates the global model by dynamic weights, and redistributes it for the next training round or local forecasting.

2.2.2. *Local BiGRU-Attention Prediction Model*

Restaurant demand shows significant time dependence and cyclical fluctuation: lunch/dinner order volumes exceed other times, weekday/weekend patterns differ, and holidays, weather, or promotions cause short-term surges. This study uses a local BiGRU-Attention model to capture these patterns.

Given Input Sequence

$$X_i^t = \left[x_i^{t-l+1}, x_i^{t-l+2}, \ldots, x_i^t\right], \tag{8}$$

The hidden state of the forward GRU is

$$\overrightarrow{h}_i^t = GRU\left(x_i^t, \overrightarrow{h}_i^{t-1}\right), \tag{9}$$

The hidden state of the backward GRU is

$$\overleftarrow{h}_i^t = GRU\left(x_i^t, \overleftarrow{h}_i^{t+1}\right). \tag{10}$$

Two-way hiding is represented as

$$h_i^t = \left[\overrightarrow{h}_i^t; \overleftarrow{h}_i^t\right]. \tag{11}$$

Attention score is calculated as follows

$$e_i^t = v_a^\top \tanh\left(W_a h_i^t + b_a\right), \tag{12}$$

Normalized attention weights are

$$\alpha_i^t = \frac{\exp\left(e_i^t\right)}{\sum_{k=1}^{l} \exp\left(e_i^k\right)}. \tag{13}$$

The context vector is

$$z_i = \sum_{t=1}^{l} \alpha_i^t h_i^t, \tag{14}$$

The final predicted output is

$$\widehat{y}_i = W_o z_i + b_o. \tag{15}$$

This structure highlights key time periods and features, giving the model stronger expressive power during lunch/dinner peaks, weekend fluctuations, and promotions. Time-series deep learning and multi-factor ML models have been used for restaurant sales and demand prediction [7, 8]; this study further adds federated training and privacy protection.

2.2.3. *Dynamic Weighted Federated Aggregation*

Standard FedAvg aggregates mainly by sample-size weights, but catering data quality, demand stability, and communication reliability vary significantly by store, so relying only on sample size lets abnormal traffic or unstable nodes hurt the global model. This study therefore designs dynamic weighted federated aggregation, using validation error and communication stability as reliability signals to adjust weights round by round rather than fixing them by sample size alone.

In the r-th round of communication, the local model parameters of node C_i are θ_i^{r+1}, and the global model is updated according to Eq. (16).

$$\theta^{r+1} = \sum_{i=1}^{N} \bar{w}_i^r \theta_i^{r+1}. \tag{16}$$

Dynamic weight is defined as

$$w_i^r = \lambda_1 \frac{|D_i|}{|D|} + \lambda_2 \frac{1}{RMSE_i^r + \varepsilon} + \lambda_3 s_i^r, \tag{17}$$

Where $RMSE_i^r$ represents the local validation error of node $C_i$in the r round, s_i^r represents the communication stability score, and ε is a constant to prevent the denominator from being zero. The weights are normalized as follows:

$$\bar{w}_i^r = \frac{w_i^r}{\sum_{j=1}^{N} w_j^r}, \quad \sum_{i=1}^{N} \bar{w}_i^r = 1. \tag{18}$$

Because secure aggregation (Section 2.2.4) exposes only the masked sum of updates, the reliability signals $RMSE$ and s needed for Eq. (17) cannot be read off an individual update and must instead be reported through a separate, lighter channel. Each round, node Ci reports these two statistics outside the masked-aggregation step; to limit what they reveal about its local data distribution, both are quantized to a small number of discrete levels and perturbed with additive noise under a small privacy budget al.located independently of the gradient-level budget ε. The server computes the normalized weight from these quantized, perturbed reports and returns it to each client, which rescales its own clipped-and-noised update locally before applying its secure-aggregation mask — scaling and masking commute, so the server still observes only the single weighted sum, never any client's individual weight or update. This bounds, but does not eliminate, what the reliability channel can leak; a tighter, formally accounted joint privacy budget across the gradient and reliability channels is left for future work (Section 3.3).

This method increases the contribution of nodes with sufficient samples, low prediction error, and stable communication, while reducing the impact of abnormal or unstable nodes. Figure 2 shows the privacy-preserving federated training process.

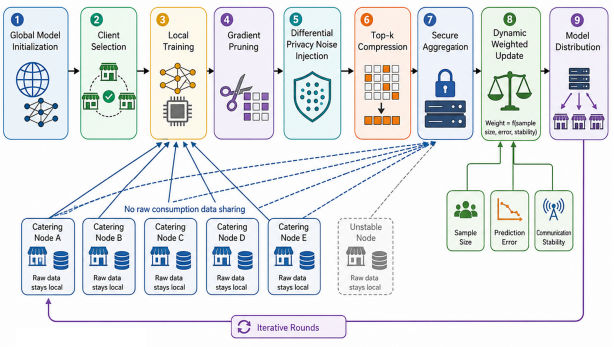

Fig. 2. Privacy-Preserving Federated Training and Dynamic Aggregation Process

As shown in Figure 2, SPFed-RDF's training process includes global model initialization, client selection, local training, gradient clipping, DP noise injection, Top-k compression, secure aggregation, dynamic weighted update, and model distribution — letting catering nodes collaboratively train without sharing raw consumption data.

2.2.4. *Privacy Protection Mechanism*

The gradient of a local node in the r-th round is denoted as g_i^r.

(1) Gradient clipping. To bound the sensitivity of a

single local update — an L2-norm clipping operation, not a pruning or sparsification step — its norm is clipped to a threshold C:

$$\bar{g}_i^r = \frac{g_i^r}{\max\left(1, \frac{\|g_i^r\|_2}{C}\right)}, \tag{19}$$

Where C is the clipping threshold.

(2) Gaussian perturbation. Gaussian noise calibrated to C and a noise multiplier σ is then added to the clipped gradient:

$$\tilde{g}_i^r = \bar{g}_i^r + \mathcal{N}\left(0, \sigma^2 C^2 I\right), \tag{20}$$

Where σ is the noise multiplier. A smaller privacy budget ε results in stronger noise and more comprehensive privacy protection, but the prediction error may increase. Differential privacy can be formally expressed as: for any adjacent datasets D and D', the random mechanism M satisfies.

$$\Pr\left[\mathcal{M}(D) \in S\right] \leq e^{\epsilon} \Pr\left[\mathcal{M}\left(D'\right) \in S\right] + \delta. \tag{21}$$

The clip-and-noise mechanism gives each client a record-level (sample-level) local DP guarantee: it bounds what an observer of a node's uploaded update — including an honest-but-curious server — can infer about any single training example in that node's dataset. It does not constitute client-level (participant-level) DP, since it does not hide whether the node participated in a given round; that stronger guarantee is left for future work (Section 3.3).

(3) Privacy accounting over multiple rounds. A single Gaussian perturbation (Eq. 21) does not certify the guarantee across the E local steps per round and R = 100 rounds actually used. With local sampling rate $q = B/|Di|$, the standard Rényi differential-privacy (RDP)/moments-accountant analysis for iterative DP-SGD gives, for order $\alpha > 1$, a per-step bound of

$$\rho(\alpha) \leq q^2 \alpha / \sigma^2. \tag{22}$$

RDP composes additively over the $T = R \cdot E$ steps per client; converting the composed bound to (ε, δ)-DP gives

$$\varepsilon = \min_{\alpha > 1} \left[T\rho(\alpha) + ln(1/\delta)/(\alpha - 1)\right]. \tag{23}$$

Table 2's budgets ($\varepsilon \in \{0.5, 1, 2, 4, 8\}$) fix $\delta = 1/|Di|$, R = 100, E = 5, and q from Table 2's batch size, then solve Eq. (23) for σ at each target ε. This RDP/moments-accountant composition is the accountant used throughout; the reported ε is an end-to-end guarantee over all 100 rounds, not a per-round figure.

(4) Secure aggregation. Clipping and DP noise protect an update against inference from its own value, but a server observing updates individually could still statistically distinguish across nodes. Secure aggregation removes this by exposing only the sum: each pair of clients agrees on a shared pseudorandom mask that cancels on summation, so the server recovers the aggregate but no individual update. Dropout is handled by threshold secret-sharing of the mask seeds, following dropout-resilient designs [5]: surviving clients can jointly remove a disconnected client's mask contribution without learning its update. The protocol assumes a semi-honest (honest-but-curious) server and tolerates collusion below the reconstruction threshold; an actively malicious or majority-colluding server is out of scope (Section 2.1.3), as are poisoning and Byzantine attacks. Because implementing this cryptographic layer at 100 clients × 100 rounds is heavy to simulate, Section 3 idealizes it: the simulator restricts the server-visible signal to the masked sum when computing accuracy and attack-success metrics, but does not implement or measure the masking and dropout-recovery communication itself. The secure-aggregation entries in Table 4 and the overhead in Section 3.1.6 therefore reflect the compressed gradient payload only, not a benchmarked cryptographic cost. The server receives the aggregated update:

$$G^r = \sum_{i=1}^{N} \tilde{g}_i^r, \tag{24}$$

The aggregated update reveals only the sum G^r, never the plaintext $\tilde{g}_i^r$ of each node, which prevents direct analysis of the consumption behavior distribution of a particular store even when the server participates in training.

2.2.5. *Communication Compression Mechanism*

A scalable catering information system needs many store nodes; uploading complete model updates each round would scale communication cost linearly with clients and parameters. This paper adopts Top-k gradient sparsity, uploading only the fraction of parameters with the largest absolute values.*Type equation here.*

$$\Delta_i^r = TopK\left(\tilde{g}_i^r, k\right). \tag{25}$$

The communication complexity of standard FedAvg is

$$Comm_{FedAvg} = O(RNP), \tag{26}$$

Where R is the number of communication rounds, N is the number of clients, and P is the number of model parameters. After Top-k compression, the dominant communication volume is proportional to the number of rounds, clients, and selected gradient entries; the per-round client upload is reduced from all parameters to the selected Top-k subset.

$$Comm_{SPFed} = O(RNkP), \quad 0 < k < 1. \tag{27}$$

Uploading only ~30% of key gradient elements can significantly reduce communication volume. Combining compression with secure aggregation improves deployment efficiency, but excessive compression risks losing effective update information, requiring a balance between overhead and performance. Research on communication-efficient and distillation-based federated learning confirms compression and sparse uploading are key to large-scale federated system usability [9–11].*Type equation here.*

2.3. Experimental Setup

2.3.1. Simulation Dataset Construction

This study constructs a multi-node catering demand simulation dataset referencing real operation patterns, with 50 basic nodes expanding to 100, each generating 180 days of hourly demand records including order volume, revenue, timing, weekday/holiday, weather, promotion intensity, takeout ratio, consumption behavior, and platform traffic. Client counts of 10, 20, 30, 50, and 100 test scalability, and four partitions (IID, mild/moderate/severe Non-IID) test heterogeneity impact, with parameters referencing related federated learning and demand-prediction research [12]. Simulated records combine deterministic calendar cycles with stochastic demand shocks from weather, promotion, and traffic; non-IID partitions vary customer mix, peak-hour intensity, and promotion sensitivity across nodes.

2.3.2. Comparison Methods

To comprehensively verify the effectiveness of SPFed-RDF, this study compared traditional time-series models, machine learning models, local deep learning models, centralized deep learning models, and federated learning models.

2.3.3. Evaluation Metrics

Predictive performance was evaluated using MAE, RMSE, and MAPE.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|, \tag{28}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}, \tag{29}$$

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|. \tag{30}$$

System performance is evaluated via training time, communication overhead, convergence rounds, and client scalability; privacy protection via member-inference and gradient-inversion attack success rates; and communication overhead via cumulative upload volume over 100 rounds. The same train-validation-test split, forecasting horizon, and communication-round setting apply to all baselines for consistent comparison.

3. Results and discussion

3.1. Experimental Results and Analysis

3.1.1. Comparison of Predictive Performance of Different Models

Figure 3 compares MAE, RMSE, and MAPE across models for 24-hour catering demand prediction. ARIMA has the highest error, since linear models struggle with nonlinear fluctuations from promotions, weather, and traffic; SVR improves somewhat but has limited long-term dependency modeling; XGBoost captures multi-feature interactions, reducing error further; Local-BiGRU captures store-specific patterns but generalizes poorly from limited single-store data; FedAvg improves accuracy via multi-node training but retains non-IID update bias; FedProx is more stable via drift limiting; FedAvg+DP has slightly higher error from privacy noise. SPFed-RDF, combining attention-based temporal modeling, dynamic weighted aggregation, and privacy-preserving training, performs best across all three metrics, achieving RMSE of 20.91 — 18.35% below FedAvg's 25.61 and 11.92% below FedProx's 23.74 — reflecting the combined effect of these mechanisms rather than any single module.

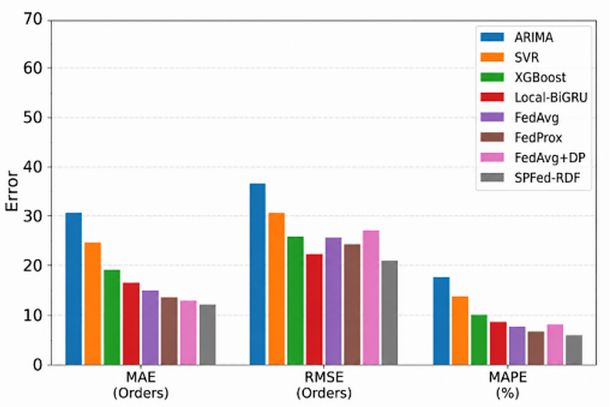

Fig. 3. Comparison of Catering Demand Prediction Errors of Different Models

3.1.2. Federated Training Convergence Performance Analysis

Figure 4 shows convergence curves after 100 rounds: FedAvg decreases quickly early but fluctuates later from heterogeneous node update inconsistency; FedAvg+DP converges slowest due to DP noise increasing gradient variance; FedProx is more stable than FedAvg but still has higher final error; SPFed-RDF stabilizes after ~60 rounds with the lowest validation error, since dynamic aggregation

Table 2. Simulation Dataset and Federated Training Parameter Settings

Project	Set up
Basic number of catering nodes	50, expanded experiment to 100
Time span	180 days
Time Granularity	1 hour
Total Sample Size	216,000 records
Input Window	24 hours
Prediction Step Size	1 hour vs. 24 hours
Data Distribution	IID, mild non-IID, moderate non-IID, severe non-IID
Client Size	10, 20, 30, 50, 100
Number of Local Training Rounds	5
Number of Communication Rounds	100
Batch Size	64
Optimizer	Adam
Learning Rate	0.001
Privacy Budget	0.5, 1, 2, 4, 8
Client Disconnection Rate	0%, 10%, 20%, 30%, 40%, 50%
Gradient Upload Ratio	10%, 30%, 50%, 70%, 100%

Table 3. Comparison Model Settings

Model	Method Description	Training methods
ARIMA	Classical statistical time series forecasting models	Centralized
SVR	Support Vector Regression Based on Kernel Functions	Centralized
XGBoost	Tree-based nonlinear regression method	Centralized
Local-BiGRU	Each catering node was trained independently using a BiGRU.	Local training
Centralized-BiGRU	BiGRU is trained by pooling all the data.	Centralized training
FedAvg	Standard Federal Average Algorithm	Federal training
FedProx	Federated learning methods with proximal regularization	Federal Training
FedAvg+DP	FedAvg with differential privacy noise	Federal Training
SPFed-RDF	This study proposes a privacy-preserving and scalable federated prediction method.	Federal Training

reduces weights of high-error, unstable nodes, stabilizing the global update direction.

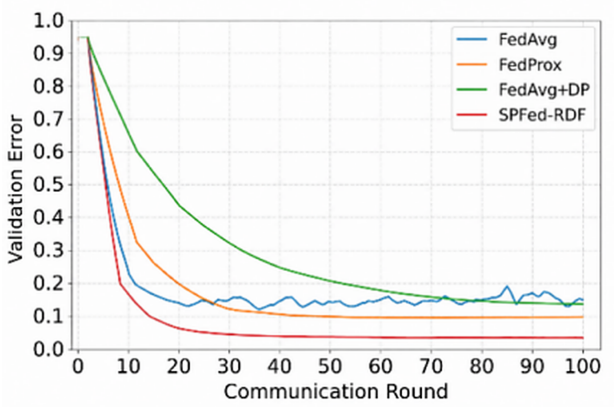

Fig. 4. Convergence Curves of Different Federated Learning Methods

3.1.3. *Visualization of Prediction Curves for Representative Stores*

Figure 5 shows actual versus predicted demand for a representative store over 72 hours, with peaks at lunch/dinner varying by date. FedAvg tends to underestimate promotional/weekend peaks and overestimate low-traffic periods; FedProx reduces some bias but peak fitting remains insufficient; SPFed-RDF fits lunch peaks, dinner peaks, and nighttime troughs better, showing attention highlights key time steps while federated training improves generalization across store demand patterns.

3.1.4. *Robustness Analysis of Non-IID Data*

Figure 6 compares RMSE across heterogeneity levels: error rises for all federated methods from IID to severe non-IID, with FedAvg most sensitive (RMSE rising to 31.82); FedAvg+DP rises further from noise and data shift; FedProx mitigates drift via proximal regularization but remains limited under high heterogeneity; SPFed-RDF grows slowest, reaching RMSE 24.36 under severe non-IID — a 23.44% reduction versus FedAvg — showing dynamic weighted aggregation mitigates bias from differing customer struc-

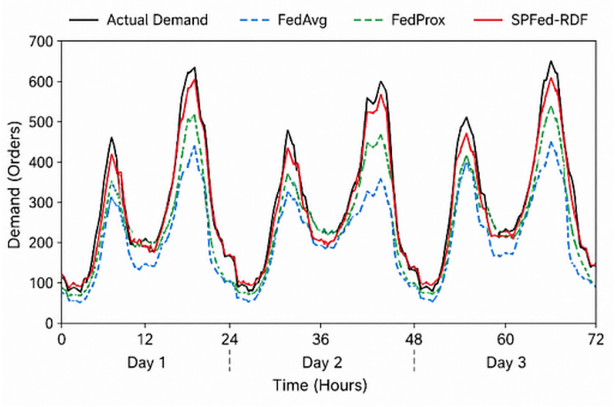

Fig. 5. Actual Demand vs. Predicted Demand Curves for Representative Catering Nodes

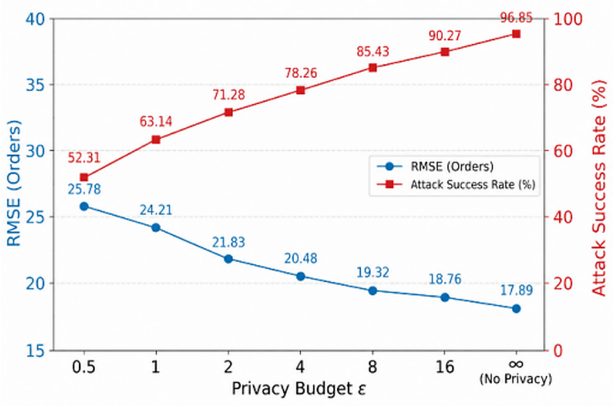

Fig. 7. Trade-off between prediction accuracy and attack success rate under different privacy budgets

ture, location, and promotion strategy across nodes.

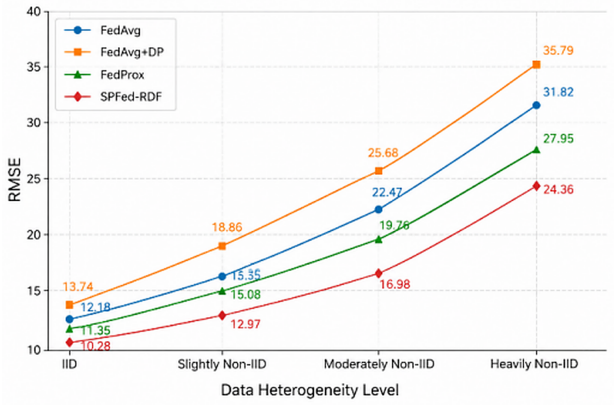

Fig. 6. Comparison of prediction errors under different levels of Non-IID

3.1.5. *Trade-off between privacy budget and prediction accuracy*

Figure 7 shows RMSE and attack success rate across privacy budgets: smaller budgets add stronger noise, reducing member-inference success but raising RMSE — at ϵ=0.5, attack success drops to 52.31% but RMSE rises to 25.78; increasing ϵ from 2 to 4 significantly lowers RMSE while keeping attack success below the unprotected model, making ϵ=2 or 4 a reasonable balance. In deployment, the privacy budget can vary by feature sensitivity: smaller for consumption frequency, food preference, and payment amount; larger for weather and calendar features, choosing smaller budgets only when accuracy loss remains acceptable for inventory and staffing decisions.

3.1.6. *Communication Overhead and Client Scalability*

Figure 8 shows communication overhead as clients grow from 10 to 100: FedAvg and FedAvg+DP grow roughly linearly, since DP perturbs gradients without reducing uploaded parameter count; SPFed-RDF without compression is slightly higher due to added privacy/stability information, but Top-k compression significantly reduces it — at 100 clients, FedAvg's 100-round cumulative volume is 1280 MB versus SPFed-RDF's 752 MB (a 41.27% reduction), confirming compression significantly improves large-scale deployment capability. Privacy perturbation and communication compression solve different bottlenecks and should be jointly configured in large client networks.

Fig. 8. Comparison of communication overhead under different numbers of clients

3.1.7. *Robustness to Client Disconnection*

Figure 9 shows RMSE across client disconnection rates: error rises for all methods, with FedAvg most sensitive (RMSE rising to 34.16 at 50% disconnection); FedProx shows some robustness but still increases significantly at high disconnection; SPFed-RDF reaches RMSE 26.84 at 50% disconnection, better than both, since the communication stability score in aggregation weights reduces long-term un-

stable nodes' impact, improving availability in edge catering networks.

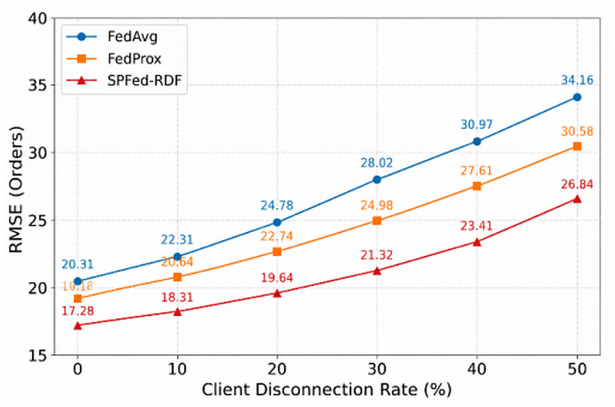

Fig. 9. Comparison of Model Robustness under Different Client Disconnection Rates

3.1.8. *Analysis of Inference Attack Protection Capability*

Figure 10 compares member-inference and gradient-inversion attack success under different protections: unprotected federated learning has 78.46% member-inference success; DP alone reduces this but the server may still observe single-node updates; secure aggregation alone hides updates but doesn't reduce single-record distinguishability; combining both further reduces success; SPFed-RDF, using gradient clipping, DP, secure aggregation, and Top-k compression together, achieves the lowest success rate under both attack types, showing stronger consumption-data protection.

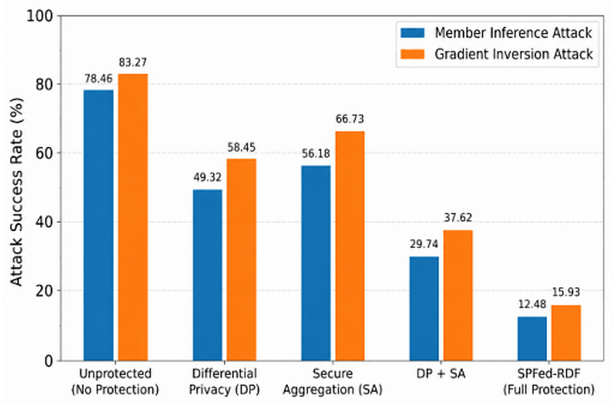

Fig. 10. Comparison of inference attack success rates under different privacy protection strategies

3.1.9. *Model Size and System Efficiency Analysis*

Figure 11 compares training time and prediction error across model sizes: large models slightly reduce RMSE but increase training time significantly; small models train fast with low communication cost but can't fully express complex temporal relationships; the medium BiGRU-Attention model balances error and training time best, so this study uses it as default — showing model selection in scalable systems must weigh communication cost and edge computing power, not just error.

Fig. 11. Comparison of training time and prediction error under different model sizes

3.1.10. *Visualization of t-SNE Feature Distribution*

Figure 12 shows t-SNE demand-representation distributions: Local-BiGRU's low/medium/high-demand samples overlap significantly since single-node training struggles for stable discriminative features; FedAvg improves class separation but medium/high-demand samples still overlap under non-IID data; FedProx improves intra-cluster compactness; SPFed-RDF forms the clearest separation boundaries, showing privacy-preserving federated training learns more discriminative demand representations — consistent with error results, indicating improvement comes from representation quality, not just parameter fitting.

3.1.11. *Sensitivity Analysis of Gradient Compression Ratio*

Figure 13 shows RMSE and communication overhead across gradient upload ratios: reducing from 100% to 30% significantly cuts overhead with only slight RMSE increase; reducing further to 10% minimizes overhead but increases error significantly, showing excessive compression loses key gradient information. A Top-k upload ratio of ~30% best balances performance and communication cost, further verifying SPFed-RDF's scalability for large-scale catering nodes.

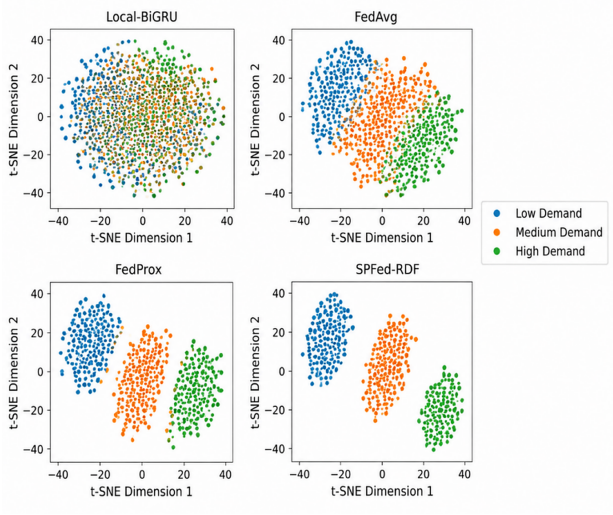

Fig. 12. Comparison of t-SNE feature distributions for different algorithms

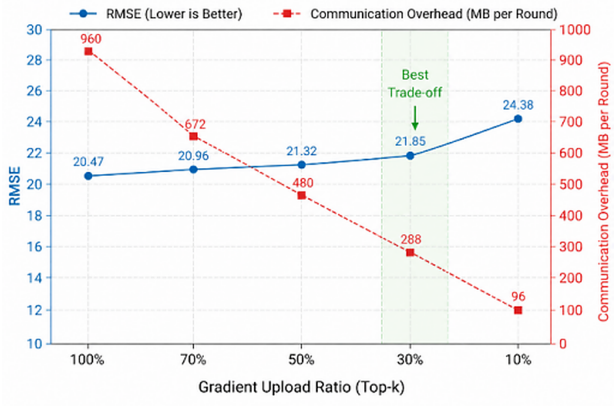

Fig. 13. Impact of Gradient Upload Ratio on Prediction Error and Communication Overhead

3.2. Security and Scalability Analysis

3.2.1. Data Privacy Protection Analysis

SPFed-RDF protects consumption data at three levels: at the data level, orders, amounts, preferences, arrival times, delivery areas, and member behaviors stay local and never reach a central server; at the gradient level, clipping bounds single-sample sensitivity and Gaussian DP noise, accounted via the Rényi-DP composition in Section 2.2.4, gives each client a record-level (ϵ, δ) guarantee over the full 100-round training process; at the aggregation level, pairwise-masked secure aggregation (idealized in simulation, Section 2.2.4) ensures the server only ever sees the weighted sum of updates, not any individual store's update, while the dynamic-aggregation reliability signals are quantized and separately perturbed before leaving the client (Section 2.2.3). No raw order identifier, delivery address, payment amount, membership trajectory, or dish-preference sequence is transmitted to the aggregation server.

3.2.2. System Scalability Analysis

SPFed-RDF supports 10–100 restaurant nodes with communication costs controlled via Top-k compression; original data stays local, avoiding centralized storage pressure; dynamic weighted aggregation reduces the impact of abnormal or offline nodes; and each node only performs local training, privacy-update generation, and compressed uploading, suiting store edge servers or enterprise private cloud deployment, with the server-side aggregator in enterprise private cloud and store nodes running lightweight training with scheduled communication windows.

3.2.3. Comprehensive Comparison of Security and Scalability

Table 4 compares security and scalability: centralized training enables unified modeling but offers weaker privacy; local training protects privacy but can't leverage cross-store data value; FedAvg provides basic federated training but doesn't address gradient privacy, communication overhead, or heterogeneity; SPFed-RDF comprehensively combines dynamic aggregation, differential privacy, secure aggregation, and compression, showing superior overall privacy, scalability, and non-IID robustness.

3.3. Discussion

This study shows restaurant demand forecasting must shift from single-point models to scalable distributed information-system design. The core issue is not whether a single model fits historical sales, but how to achieve collaborative modeling across stores, platforms, and data owners without leaking raw consumption data. Centralized training uses more complete data but carries higher privacy/compliance risk; local training protects data but can't leverage cross-store patterns; standard federated learning enables collaboration but still faces non-IID, communication, and gradient-leakage issues.

SPFed-RDF's advantage lies in unifying predictive modeling, privacy mechanisms, and system scalability: BiGRU-Attention captures temporal patterns and key demand drivers; dynamic weighted aggregation adapts to inter-store data differences; differential privacy and secure aggregation reduce consumer-data leakage risk; and Top-k compression reduces multi-round communication overhead. t-SNE visualization further shows SPFed-RDF's learned representations are more focused and interpretable for restaurant management decisions.

Limitations: experimental data is simulation-based on real operational patterns, requiring future verification with

Table 4. Comparison of Security and Scalability of Different Methods

Method	Original data protection	Gradient privacy	Secure aggregation	Communication compression	Non-IID robustness	Client scalability
Intensive training	Low	Low	No	No	Middle	Low
Local training	High	High	No	No	Low	Middle
FedAvg	Medium	Low	No	No	Low	Middle
FedAvg+DP	High	Medium	No	No	Low	Middle
FedProx	Medium	Low	No	No	Medium	Middle
SPFed-RDF	High	High	Yes	Yes	High	High

larger real chain-restaurant data; the privacy budget is accounted per client via the Rényi-DP composition in Section 2.2.4 but uses a fixed configuration, with adaptive per-feature allocation and a formally accounted joint budget across the gradient and reliability channels (Section 2.2.3) left for future work; secure aggregation is idealized in simulation rather than benchmarked as a concrete cryptographic implementation, and its computation/communication cost under an actual pairwise-masking protocol needs separate measurement; this study considers honest-but-curious servers, member inference, and gradient inversion, with poisoning, backdoor, Byzantine attacks, and an actively malicious or majority-colluding server left for future consideration; and the current centralized aggregation server could be enhanced via blockchain auditing or decentralized aggregation. Regulatory validation under jurisdiction-specific data-protection rules and deployment tests on real POS, membership, and delivery-platform interfaces remain necessary before production use.

4. Conclusions

This study proposes SPFed-RDF, a scalable privacy-preserving federated learning demand prediction framework letting restaurant nodes collaboratively train without uploading raw consumption data. A BiGRU-Attention local model captures temporal demand fluctuations and key drivers; dynamic weighted federated aggregation mitigates non-IID bias; and gradient clipping, differential privacy, secure aggregation, and Top-k compression reduce privacy leakage and communication costs. Simulations show SPFed-RDF outperforms traditional, local deep learning, and standard federated models in RMSE, MAE, MAPE, communication overhead, and attack protection. Future work will pursue validation on real multi-city, multi-brand data, adaptive differential privacy by feature sensitivity, and robust aggregation with trusted auditing against malicious clients.

References

[1] A. Schmidt, M. W. U. Kabir, and M. T. Hoque, (2022) *"Machine Learning Based Restaurant Sales Forecasting"* **Machine Learning and Knowledge Extraction *4***(1): 105–130. DOI: 10.3390/make4010006.

[2] B. (Chae, C. Sheu, and E. O. Park, (2024) *"The Value of Data, Machine Learning, and Deep Learning in Restaurant Demand Forecasting: Insights and Lessons Learned from a Large Restaurant Chain"* **Decision Support Systems *184***: 114291. DOI: 10.1016/j.dss.2024.114291.

[3] M. S. Hossain and F. Parvin, (2025) *"A Comparative Study of Various Statistical and Machine Learning Models for Predicting Restaurant Demand in Bangladesh"* **PLOS ONE *20***(6): e0325449. DOI: 10.1371/journal.pone.0325449.

[4] C. Baek, S. Kim, D. Nam, and J. Park, (2021) *"Enhancing Differential Privacy for Federated Learning at Scale"* **IEEE Access *9***: 148090–148103. DOI: 10.1109/ACCESS.2021.3124020.

[5] Z. Liu, J. Guo, K.-Y. Lam, and J. Zhao, (2023) *"Efficient Dropout-Resilient Aggregation for Privacy-Preserving Machine Learning"* **IEEE Transactions on Information Forensics and Security *18***: 1839–1854. DOI: 10.1109/TIFS.2022.3163592.

[6] T. Qi, F. Wu, C. Wu, L. He, Y. Huang, and X. Xie, (2023) *"Differentially Private Knowledge Transfer for Federated Learning"* **Nature Communications *14***(1): 3785. DOI: 10.1038/s41467-023-38794-x.

[7] S. Kim, (2025) *"Development of an AI-Based Restaurant Menu Demand Prediction Model Utilizing Sales and Meteorological Data"* **Food Science and Biotechnology *34***(15): 3597–3606. DOI: 10.1007/s10068-025-01956-2.

[8] F. Haselbeck, J. Killinger, K. Menrad, T. Hannus, and D. G. Grimm, (2022) *"Machine Learning Outperforms Classical Forecasting on Horticultural Sales Predictions"* **Machine Learning with Applications *7***: 100239. DOI: 10.1016/j.mlwa.2021.100239.

[9] C. Wu, F. Wu, L. Lyu, Y. Huang, and X. Xie, (2022) *"Communication-Efficient Federated Learning via Knowledge Distillation"* **Nature Communications *13***(1): 2032. DOI: 10.1038/s41467-022-29763-x.

[10] Y. Liu, Y. Kang, T. Zou, Y. Pu, Y. He, X. Ye, Y. Ouyang, Y.-Q. Zhang, and Q. Yang, (2024) *"Vertical Federated Learning: Concepts, Advances, and Challenges"* **IEEE Transactions on Knowledge and Data Engineering *36***(7): 3615–3634. DOI: 10.1109/TKDE.2024.3352628.

[11] P. Qi, D. Chiaro, A. Guzzo, M. Ianni, G. Fortino, and F. Piccialli, (2024) *"Model Aggregation Techniques in Federated Learning: A Comprehensive Survey"* **Future Generation Computer Systems *150***: 272–293. DOI: 10.1016/j.future.2023.09.008.

[12] H. Wang, F. Xie, Q. Duan, and J. Li, (2022) *"Federated Learning for Supply Chain Demand Forecasting"* **Mathematical Problems in Engineering *2022***: 4109070. DOI: 10.1155/2022/4109070.