Formal Logic Inference Guided Uncertainty Quantification for Personalized Federated Learning

GUOCHENG HE, ZIYAN AN, and MEIYI MA, Vanderbilt University, USA

Federated Learning (FL) enables privacy-preserving model training across heterogeneous distributed systems, such as smart-grid forecasting or traffic-flow prediction from geographically dispersed sensors and devices. A key challenge in such settings is capturing client-specific patterns while addressing data heterogeneity and uncertainty at scale. Existing approaches, including Bayesian Neural Networks (BNNs) and clustering-based methods, struggle with scalability and consistent personalization. We propose **LogiCP**, a novel FL framework that integrates formal logic reasoning with uncertainty quantification (UQ) to support scalable and personalized learning with theoretical guarantees. LogiCP uses Signal Temporal Logic (STL) to extract temporal patterns and form semantically coherent client clusters, controlling intra-cluster heterogeneity. Within each cluster, LogiCP applies decentralized Conformal Prediction (CP) to produce distribution-free prediction intervals with mathematical guarantees that encompass the real value. LogiCP dynamically assigns clients to clusters at runtime without retraining, improving practicality. Evaluations on three real-world datasets—traffic, temperature, and electricity—show that LogiCP consistently outperforms BNN-, clustering-, and CP-based baselines, achieving up to a 95% improvement in client-level MSE while maintaining strong scalability.

JAIR Track: Integration of Logical Constraints in Deep Learning

JAIR Associate Editor: Matteo Zavatteri

JAIR Reference Format:
Guocheng He, Ziyan An, and Meiyi Ma. 2026. Formal Logic Inference Guided Uncertainty Quantification for Personalized Federated Learning. *Journal of Artificial Intelligence Research* 86, Article 21 (July 2026), 27 pages. DOI: 10.1613/jair.1.21429

1 Introduction

Federated learning (FL) (McMahan et al. 2017) is a distributed learning paradigm with widespread applications in real-world, large-scale systems such as smart healthcare and connected vehicular systems (B. Li et al. 2020; Y. Lu et al. 2020; Ma, Preum, et al. 2019; Preum et al. 2021). When applied to safety-critical sequential prediction tasks within such systems, it is essential for FL models to produce reliable and robust outcomes (Pati et al. 2024). For example, in traffic flow prediction, precise forecasts are crucial for effective planning and execution (Kashyap et al. 2022; Medina-Salgado et al. 2022), as inaccuracies can lead to increased driver wait times, traffic congestion, and an overall higher risk of accidents, slowing the operations of intelligent transportation systems (Kaffash et al. 2021; Kong et al. 2025). Applying the standard FL framework to such systems, numerous endpoints (i.e., FL clients) collaboratively contribute to a shared model under the coordination of a central server. FL models aim to leverage client-specific data to enhance shared performance without direct access to private data. However, standard approaches often fail to capture latent personalized features, as a single global model is uniformly shared across all clients. To address this limitation, Personalized FL (PFL) has emerged as a promising approach that

Authors' Contact Information: Guocheng He, ORCID: 0009-0001-7240-9767, guocheng.he@vanderbilt.edu; Ziyan An, ORCID: 0000-0002-1083-0011, ziyan.an@vanderbilt.edu; Meiyi Ma, ORCID: 0000-0001-6916-8774, meiyi.ma@vanderbilt.edu, Vanderbilt University, Nashville, Tennessee, USA.

DOI: 10.1613/jair.1.21429

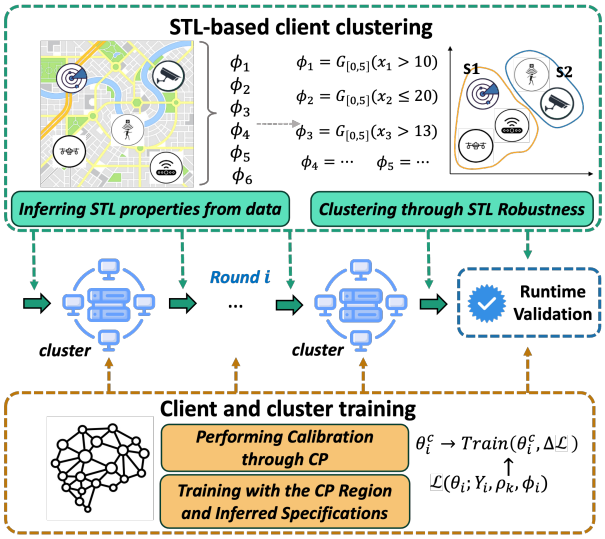

Fig. 1. Overview of LogiCP.

tailors model updates to client-specific data distributions, enhancing local accuracy while still benefiting from global knowledge aggregation (Banabilah et al. 2022).

State-of-the-art approaches to achieve personalization in FL include Bayesian Neural Networks (BNNs) and client clustering. BNN-based methods learn a prior distribution for each client to enable latent feature inference, thereby supporting personalization (Pati et al. 2024). Clustering-based methods, in contrast, group clients with similar data distributions to train cluster-specific models, aiming to balance personalization with generalizability (Caldarola et al. 2021; Ghosh et al. 2020). However, both approaches still face practical challenges when confronted with extreme, large-scale heterogeneity in real-world deployments (Ji et al. 2023; Yan et al. 2023; X. Zhang, Wen, et al. 2024). BNN integration requires appropriate prior distributions for effective training. However, this requirement becomes computationally infeasible with a growing number of clients, due to the substantial overhead involved in learning and maintaining client-specific priors, especially in resource-constrained environments. Meanwhile, as the number of clients increases, data heterogeneity grows significantly (L. Li et al. 2020; Pati et al. 2024). This rising diversity leads to higher intra-cluster variance and greater uncertainty in the shared cluster models. Consequently, it becomes increasingly challenging for clustering-based methods to capture the underlying data characteristics of all participating clients effectively. These limitations underscore the crucial need for a scalable and reliable framework that addresses data heterogeneity and uncertainty while maintaining computational efficiency and personalized learning across diverse clients.

To address these limitations, we develop a novel framework, **LogiCP**, that integrates rigorous uncertainty quantification (UQ) with formal logic inference into the FL framework. Specifically, LogiCP integrates Conformal Prediction (CP) (Shafer and Vovk 2008) with Signal Temporal Logic (STL)-based inference (An et al. 2024; Ji et al. 2023; Ma, Gao, et al. 2020) to address data heterogeneity and model uncertainty. Instead of enforcing system-wide performance goals, LogiCP infers temporal specifications directly from each client in a personalized manner. These specifications are then used to assess semantic alignment among clients, enabling the formation of coherent and efficient clusters. At runtime, LogiCP leverages these specifications to dynamically assign clients to the most semantically aligned clusters, thereby improving adaptability in dynamic environments. In addition, by grouping

clients with similar temporal patterns, LogiCP significantly reduces intra-cluster heterogeneity, which enhances the reliability of localized UQ.

Within each cluster, LogiCP employs CP to generate distribution-free prediction intervals with theoretical guarantees, ensuring the true outcome lies within the predicted interval at a user-defined confidence level. The similarity of temporal patterns among clients within a cluster allows CP intervals to more accurately capture prediction uncertainty. These intervals are further integrated as logical constraints into the model training process, enabling scalable, uncertainty-aware updates that preserve both personalization and efficiency. LogiCP ensures that conformity scores and prediction intervals are computed locally without sharing raw data to preserve data privacy. This design supports rigorous uncertainty estimation while maintaining strict privacy guarantees.

We validate the effectiveness of LogiCP on three real-world datasets, including traffic flow prediction, global city temperature forecasting, and electricity consumption, under different client scales and participation settings. LogiCP achieves up to a 95% improvement in client-level MSE compared to baseline methods, including BNN-based, clustering-based, and CP-based approaches. Additionally, LogiCP incorporates a runtime prediction validation mechanism that leverages STL-guided uncertainty estimates to improve model accuracy at runtime. Our evaluation demonstrates that LogiCP more effectively captures prediction uncertainty than existing UQ methods that lack formal logic reasoning, highlighting the importance of logic-guided inference in managing heterogeneity in FL. Moreover, LogiCP maintains consistent performance across different client scales and outperforms baseline methods. To the best of our knowledge, this is the first work to integrate formal logic-guided uncertainty quantification within the FL framework, specifically designed to address scalability and heterogeneity while ensuring personalization.

In summary, the key contributions of this paper include,

- LogiCP is the first framework to incorporate a formal logic-guided uncertainty quantification method in a federated learning setting. CP-derived regions are incorporated as logical constraints, guiding both local and cluster-level model updates with mathematically rigorous guarantees without solving intensive specifications or centralized data access.
- LogiCP incorporates a runtime validation mechanism that integrates formal logic-guided uncertainty, further improving prediction reliability and robustness in dynamic deployment settings.
- LogiCP is evaluated on three real-world datasets with diverse temporal characteristics, demonstrating substantial performance enhancement over baseline methods across key evaluation metrics. Specifically, LogiCP outperforms baseline methods up to 95% improvement on client-wise MSE, validating strong performance under heterogeneous data distributions. Meanwhile, LogiCP presents consistent performance under different scales of clients and partial participation settings, showcasing its scalability and robustness.

2 Problem Formulation

We consider a FL problem composed of a central server with global model parameters θ^p, a set of N clients $C = \{C_1, C_2, \ldots, C_N\}$, and K clusters $S = \{S_1, S_2, \ldots, S_K\}$. Each client $C_i \in C$ and cluster $S_k \in S$ is associated with trainable model parameters θ_i^c and θ_k^s, respectively.

Each client C_i possesses a distinct local dataset D_i, which consists of M independent and identically distributed (i.i.d.) data points. Each client's local training dataset D_i is partitioned into a private subset D_i^{pri} used exclusively for local training, and a public subset D_i^{pub} shared with the assigned cluster for cluster-level training. To prevent domain skew in the aggregated cluster dataset P_k, random temporal sampling is employed when constructing D_i^{pub}, ensuring that different time periods (e.g., weekdays vs. weekends, peak vs. off-peak hours) are proportionally represented. For each cluster S_k, its dataset P_k is constructed by aggregating the public datasets of the clients assigned to it, i.e., $P_k = \bigcup_{C_i \in S_k} D_i^{pub}$.

We assume that each client follows STL-based patterns ϕ_i defined or inferred from their dataset D_i. Let $g(\theta, Y, \phi)$ denote the robustness function measuring how well a model θ's prediction on data point Y satisfies the STL specification ϕ. For a client C_i and a sequence of predictions $\hat{Y} = \{\hat{Y}_1, \ldots, \hat{Y}_T\}$ generated by cluster model θ_k^s, semantic alignment with the client's specification ϕ_i is measured as:

$$\mathcal{R}_{STL}(\theta_k^s, \hat{Y}, \phi_i) = \frac{1}{T} \sum_{\tau=1}^{T} g(\theta_k^s, \hat{Y}_\tau, \phi_i), \tag{1}$$

where lower values of $\mathcal{R}_{STL}$ indicate stronger semantic alignment.

We further define a CP-based uncertainty quantification coverage function. Let $\alpha \in (0, 1)$ denote the user-specified error rate, such that the target coverage level is $1 - \alpha$ (e.g., $\alpha = 0.1$ corresponds to 90% coverage). The CP region $\rho = C(\theta, D)$ represents the prediction interval computed via conformal prediction. The empirical coverage function $\mathcal{S}_{CP}$ measures the proportion of ground truth values that fall within this region. A high semantic alignment clustering reduces in-cluster heterogeneity, enhancing the accuracy of uncertainty estimates.

$$\mathcal{S}_{CP}(\theta, D) = \frac{1}{|D|} \sum_{Y \in D} \mathbb{1}\{Y \in C(\theta, D)\} \tag{2}$$

To measure the model prediction accuracy, let $f(\theta, Y)$ denote the Euclidean loss between a model's prediction and the ground truth for data point Y. The expected loss over dataset D is:

$$\mathcal{L}(\theta, D) = \frac{1}{|D|} \sum_{Y \in D} f(\theta, Y) = \mathbb{E}_{Y \in D}[f(\theta, Y)] \tag{3}$$

Hence, we have three objectives in total:

(O1) we aim to minimize semantic misalignment across clusters while satisfying the CP logical constraint:

$$\min_{\{S_k\}} \sum_{k=1}^{K} \sum_{C_i \in S_k} \mathcal{R}_{STL}(\theta_k^s, \hat{Y}_k, \phi_i) \quad \text{subject to} \quad \mathcal{S}_{CP}(\theta, D) \geq 1 - \alpha \tag{4}$$

(O2) The objective for each local client C_i aims to find the model parameter θ_i^c that approximates the optimal parameter θ_i^*. The optimal parameter minimizes the loss on the local client's dataset D_i. The client-wise objective is:

$$\min_{\theta} \mathcal{L}(\theta_i^c; D_i^{pri}) \quad \text{for } C_i \in C, \quad \text{subject to} \quad \theta_i^c \approx \theta_i^* \tag{5}$$

(O3) The objective for each cluster S_k aims to find model parameter θ_k^s that approximates the optimal parameter θ_k^*, which minimizes the cluster model's prediction loss on the aggregated cluster dataset P_k. The cluster-wise objective is:

$$\min_{\theta} \mathcal{L}(\theta_k^s; P_k) \quad \text{for } S_k \in S, \quad \text{subject to} \quad \theta_k^s \approx \theta_k^* \tag{6}$$

3 Framework Overview

In this section, we present an overview of LogiCP, which consists of three components: STL-based clustering, client and cluster training, and runtime validation. LogiCP infers STL properties from client data and uses STL robustness to form clusters. In training, LogiCP applies formal logic-guided CP for UQ and updates both client-wise and cluster-wise models with a loss function that incorporates quantified uncertainty regions and inferred logic as constraints. The runtime validation stage then uses the inferred STL specifications and CP regions to validate model predictions. An overview of the proposed method is presented in Fig. 1.

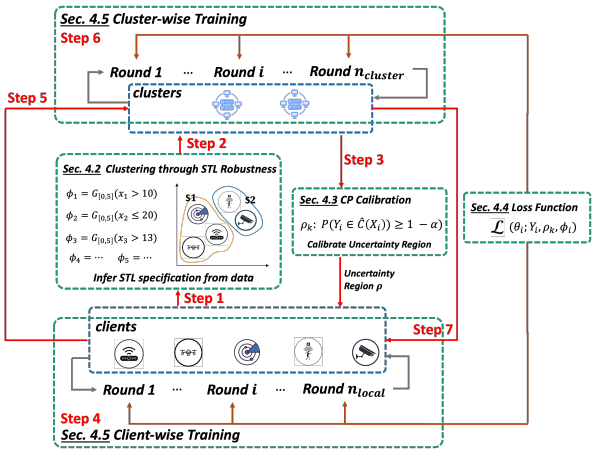

Fig. 2. The training workflow in each communication round to incorporate CP and temporal logic for model updating.

Algorithm 1 summarizes the overall workflow of LogiCP. The training-time optimization begins with a pretraining phase that follows the standard FL framework to initialize client models before STL inference and CP calibration (Algorithm 1, Lines 4–15). Pretraining provides a stable initialization for the subsequent logic-guided and uncertainty-aware training process. The technical details of the pretrain process are summarized in Section 4.1.

Upon completion of the pretraining process, the first round of STL-guided clustering and CP calibration is performed. The workflow of the training process is provided in Fig. 2. In each communication round, LogiCP infers STL specifications to capture client-level temporal properties and forms semantically coherent clusters (Algorithm 1, Lines 13). We define a cluster as *semantically coherent* if its member clients share similar STL specifications, i.e., their temporal patterns (such as peak times, value ranges, and trend behaviors) are aligned. Within each cluster, distributed CP is applied to generate the personalized CP region ρ_k (Algorithm 1, Lines 14). The technical details of STL-guided clustering and CP calibration are provided in Section 4.2 and Section 4.3, respectively.

Following temporal logic inference and CP in each communication round, each client trains locally with a loss function $\mathcal{L}(\theta_i^c; Y_i, \rho_k, \phi_i)$ that jointly accounts for prediction accuracy, STL consistency, and CP-derived uncertainty constraints (Algorithm 1, Lines 18–23). The technical details of the loss function are provided in Section 4.4.

At the last stage of each communication round, client updates are aggregated to update both the global model and the corresponding cluster models, while each cluster further refines its model using the same loss function $\mathcal{L}$ (Algorithm 1, Lines 24–30). This design preserves personalization while allowing clients within the same semantic cluster to benefit from shared training signals.

The communication round in the training process is repeated for n_{tr} times; the technical details of the training process are presented in Section 4.5.

After training, LogiCP further applies a runtime validation and correction mechanism during prediction. The correction is guided by both the CP region ρ_k and the inferred STL specification ϕ_i to validate model outputs and correct violations of inferred temporal properties when needed. when these two regions conflict, LogiCP prioritizes the CP-derived boundary correction target because it provides a distribution-free coverage guarantee. The technical details of runtime validation and correction are provided in Section 4.6.

Algorithm 1 LogiCP Framework

Input: Set of total N clients $\{C_i\}$, each with dataset $\{D_i\}$ containing total M data points
Parameter: θ^p, n_{local}, n_{pre}, n_{tr}, $n_{cluster}$
Initialize θ^p on central server
for $epoch = 1$ to n_{pre} **do** // Section 4.1
  for each client $C_i \in C$ *in parallel* **do**
    Broadcast θ^p to all clients $\{C_i\}$
    for $local\_round = 1$ to n_{local} **do**
      Client C_i pretrain on D_i^{pre} to update θ_i^c
    end for
  end for
  Central server aggregates $\{\theta_i^c\}$ to update θ^p
  if $epoch = n_{pre}$ **then**
    $\{S_k\}_{k=1}^K, \{\phi_i\}_{i=1}^N \leftarrow STL(\{\theta_i^c\}_{i=1}^N)$ // Section 4.2
    $\{\rho_k\}_{k=1}^K \leftarrow CP(\{\theta_i^c\}_{i=1}^N, \{S_k\}_{k=1}^K)$ // Section 4.3
  end if
end for
for $round = 1$ to n_{tr} **do**
  for each client $C_i \in C$ *in parallel* **do** // Section 4.5
    for $local\_round = 1$ to n_{local} **do**
      Client C_i train on D_i^{tr} with $\mathcal{L}(\theta_i^c; Y_i, \rho_k, \phi_i)$ to update θ_i^c // Section 4.4
      Aggregate θ_i^c to update global model θ^p via weighted averaging
      Aggregate θ_i^c from clients in S_k to update each cluster model θ_k^s
    end for
  end for
  for each cluster $S_k \in S$ *in parallel* **do**
    for each $C_i \in S_k$ *in parallel* **do**
      Server constructs $P_k = \bigcup_{C_i \in S_k} D_i^{pub}$
      for $cluster\_round = 1$ to $n_{cluster}$ **do**
        Cluster θ_k^s train on P_k with $\mathcal{L}(\theta_k^s; Y_i, \rho_k, \phi_i)$ to update θ_k^s // Section 4.4
      end for
      For each client $C_i \in S_k$, update shared parameters in θ_i^c with θ_k^s
    end for
    $\{S_k\}_{k=1}^K, \{\phi_i\}_{i=1}^N \leftarrow STL(\{\theta_i^c\}_{i=1}^N)$
    $\{\rho_k\}_{k=1}^K \leftarrow CP(\{\theta_i^c\}_{i=1}^N, \{S_k\}_{k=1}^K)$
  end for
end for
return Final central model θ^p, every cluster model $\{\theta_i^c\}_{i=1}^N$ and every client model $\{\theta_k^s\}_{k=1}^K$

4 Method

In this section, we describe each component of LogiCP in detail. The overarching goal of LogiCP is to mitigate data heterogeneity by clustering clients with similar temporal specifications and to guide UQ during model updates. To achieve this, LogiCP infers each client's temporal logic properties using parameterized STL and computes uncertainty regions through CP. These regions serve as logical constraints, providing rigorous guidance for FL updates and enhancing both prediction accuracy and reliability.

4.1 Pretraining Process

The pretraining phase is introduced to eliminate the confounding influence of poor model initialization, ensuring that uncertainty estimates from CP reflect true predictive uncertainty. This phase follows standard FL setups and provides initial training to each client model. By establishing a baseline level of accuracy, the pretraining phase improves the efficiency of the first CP application and helps lower overall training costs. Importantly, this phase is optional in LogiCP; its primary benefit is to speed up convergence by disentangling the effects of poor initialization from genuine predictive uncertainty, thus improving early-stage stability in heterogeneous settings.

Specifically, consider a set of N clients $C = \{C_1, C_2, \ldots, C_N\}$, where each client C_i possesses its own local pretraining dataset D_i^{pre}. The goal of the pretraining phase is to train a global model θ^p that minimizes the empirical prediction error across all clients' local data, without requiring any raw data exchange. This process aims to provide each client with a modestly initialized model that reduces the impact of poor performance on early-stage uncertainty estimation. The objective is formally defined as:

$$\min_{\theta} \left\{ \mathcal{L}_{pre}(\theta) = \frac{1}{N} \sum_{i=1}^{N} L_{MSE}(\theta_i^c; D_i^{pre}) \right\} \tag{7}$$

Here, $L_{MSE}(\theta_i^c; D_i^{pre})$ denotes the MSE loss computed locally on client C_i's pretraining dataset.

4.2 Clustering Through STL Robustness

Inherent data heterogeneity among clients poses a critical challenge, as each client may exhibit distinct temporal patterns and data distributions. To address data heterogeneity, LogiCP utilizes STL to express and infer the temporal specification of client data formally. These STL specifications are then used to guide the formation of semantically coherent clusters that reduce in-cluster heterogeneity and improve the accuracy of UQ.

We begin by introducing the syntax of STL, referring to (Maler and Nickovic 2004), a logic-based framework that provides a rigorous and expressive way to specify temporal behaviors in sequential data. STL is particularly well-suited for reasoning about time-series data due to its ability to incorporate temporal constraints directly into logical specifications. (A further discussion for using STL over other formalisms like first-order logic (FOL) is provided in Section 6.3.)

Definition 1 (STL Syntax for Temporal Property Inference in LogiCP).
The syntax of an STL formula used in our setting is defined as follows:

$$\phi := \mu \mid \neg\phi \mid \phi \vee \phi \mid \phi \wedge \phi \mid \Diamond_{[a,b]}\phi \mid \Box_{[a,b]}\phi \mid \phi_1 \,\mathcal{U}_{[a,b]}\, \phi_2 \tag{8}$$

Where:

- μ is an atomic predicate of the form $f(x(t)) \sim c$, where f is a real-valued function (e.g., identity, difference), $\sim \in \{\leq, \geq\}$, and $c \in \mathbb{R}$.
- $\neg$ is logical negation, $\vee$ and $\wedge$ are logical disjunction and conjunction.
- $\Diamond_{[a,b]}\phi$ ("eventually") holds if ϕ becomes true at some point in $[a, b]$.
- $\Box_{[a,b]}\phi$ ("always") holds if ϕ remains true throughout $[a, b]$.
- $\phi_1 \,\mathcal{U}_{[a,b]}\, \phi_2$ ("until") holds if ϕ_1 is true until ϕ_2 becomes true in $[a, b]$.

This syntax allows LogiCP to express a wide range of specifications that describe temporal patterns in client data, such as traffic flow consistency, threshold violations, and bounded fluctuations. These formal logics are then used to guide the clustering of clients based on semantic alignment.

Signal Representation. Let $\mathbf{x} = (x_1, x_2, \ldots, x_T)$ denote a discrete-time signal of length T, where $x_\tau \in \mathbb{R}$ represents the signal value at time step τ. In our experiments, x_τ corresponds to the sequential measurement,

such as traffic volume, temperature, or electricity consumption. Normalization is performed locally on each client's data using min-max scaling to $[0, 1]$ to preserve privacy.

Robustness Function. Given an STL formula ϕ, the (violation) robustness function $g(\theta, \mathbf{x}, \phi)$ computes a non-negative violation distance measuring how far the signal $\mathbf{x}$ deviates from satisfying ϕ, adapted from the TeLEx framework (Jha et al. 2017). A robustness value of $g = 0$ indicates that the specification is satisfied, without quantifying the degree of satisfaction. An STL formula ϕ is constructed recursively from atomic predicates combined with Boolean and temporal operators (as defined in Definition 1). An atomic predicate μ takes the form $\mu := x \sim c$, where $c \in \mathbb{R}$ is a threshold parameter embedded in the formula, and $\sim \in \{\leq, \geq\}$ is a comparison operator.

Definition 2 (Robustness Function of STL g).

$$
\begin{aligned}
g(\theta, \mathbf{x}_\tau, \mathbf{x}_\tau \geq c) &= \max(0,\ c - x_\tau) \\
g(\theta, \mathbf{x}_\tau, \mathbf{x}_\tau \leq c) &= \max(0,\ x_\tau - c) \\
g(\theta, \mathbf{x}_\tau, \neg\phi) &= g(\theta, \mathbf{x}_\tau, \phi) \\
g(\theta, \mathbf{x}_\tau, \phi_1 \vee \phi_2) &= \min\{g(\theta, \mathbf{x}_\tau, \phi_1), g(\theta, \mathbf{x}_\tau, \phi_2)\} \\
g(\theta, \mathbf{x}_\tau, \phi_1 \wedge \phi_2) &= \max\{g(\theta, \mathbf{x}_\tau, \phi_1), g(\theta, \mathbf{x}_\tau, \phi_2)\} \\
g(\theta, \mathbf{x}_\tau, \Diamond_{[a,b]}\phi) &= \min_{\tau' \in [\tau+a,\, \tau+b]} g(\theta, \mathbf{x}_{\tau'}, \phi) \\
g(\theta, \mathbf{x}_\tau, \Box_{[a,b]}\phi) &= \max_{\tau' \in [\tau+a,\, \tau+b]} g(\theta, \mathbf{x}_{\tau'}, \phi) \\
g(\theta, \mathbf{x}_\tau, \phi_1 \mathcal{U}_{[a,b]} \phi_2) &= \inf_{\tau' \in [\tau+a,\, \tau+b] \cap \mathbb{T}} \max\Big(g(\theta, \mathbf{x}_{\tau'}, \phi_2),\ \sup_{\tau'' \in [\tau,\, \tau']} g(\theta, \mathbf{x}_{\tau''}, \phi_1)\Big)
\end{aligned}
\tag{9}
$$

We note that this definition differs from classical STL quantitative semantics (Maler and Nickovic 2004) and the original TeLEx robustness (Jha et al. 2017), which use a signed, margin-based measure where positive values encode the degree of satisfaction and negative values encode the degree of violation. In contrast, the function g defined above is a **non-negative violation penalty**: it returns zero for all satisfying trajectories regardless of the satisfaction margin, and returns a positive value proportional to the violation distance otherwise. In LogiCP, we adopt this formulation because the robustness function serves to measure semantic misalignment when assigning clients to clusters. The violation distance directly quantifies how far a cluster model's predictions deviate from a client's inferred specification, where the relevant quantity is the magnitude of deviation from satisfaction rather than the degree to which predictions exceed the satisfaction threshold.

We use the averaged STL robustness $\mathcal{R}_{STL}(\theta_k^s, \hat{Y}, \phi_i)$ as the clustering metric, which quantifies how well a cluster model's predictions align with each client's inferred STL specification. These specifications ϕ_i are mined from templated STL formulas (Donzé and Maler 2010) applied to each client's dataset D_i, using Telex (Jha et al. 2017). Telex approximates the quantitative semantics of STL through differentiable functions, thereby converting the specification inference problem into an optimization problem. Additional details of this process can refer to Jha et al. (Jha et al. 2017).

STL quantitative semantics capture the degree to which a cluster model's predictions satisfy a local client's temporal specification. To compute this robustness, the predicted sequence $\hat{Y}$ is evaluated against the STL formula ϕ_i, where the degree of violations or satisfactions is quantified as real-valued scores.

We define the STL robustness value $\mathcal{R}_{STL}(\theta_k^s, \hat{Y}, \phi_i)$ as the average robustness value across a prediction sequence $\hat{Y} = \{\hat{Y}_1, \ldots, \hat{Y}_T\}$ outputted by cluster model θ_k^s when evaluated on the specification ϕ_i. Formally:

$$
\mathcal{R}_{STL}(\theta_k^s, \hat{Y}, \phi_i) = \frac{1}{T} \sum_{\tau=1}^{T} g(\theta_k^s, \hat{Y}_\tau, \phi_i) \tag{10}
$$

where $g(\cdot,\cdot,\cdot)$ is the robustness of prediction $\hat{Y}_\tau$ with respect to ϕ_i at time τ. A lower value indicates stronger semantic alignment between the cluster model and the client specification. After inferring ϕ_i, each client C_i is assigned to the cluster S_k whose model θ_k^s yields the lowest score, thus forming semantically coherent clusters that support accurate and personalized UQ.

Property Templates and Examples. We employ parameterized STL templates that capture common temporal patterns in sequential data:

- **Bounded range:** $\square_{[0,T]}(x \geq c_1) \wedge \square_{[0,T]}(x \leq c_2)$ — the signal remains within bounds $[c_1, c_2]$ throughout the prediction horizon.
- **Peak detection:** $\diamond_{[a,b]}(x > c)$ — the signal exceeds threshold c at some point within interval $[a, b]$.
- **Trend constraint:** $\square_{[a,b]}(x_{t+1} - x_t < \delta)$ — within interval $[a, b]$, the rate of change is bounded by δ.

Consider a traffic sensor client C_i with hourly volume data. We infer the specification $\phi_i = \square_{[0,5]}(x > 0.3) \wedge \diamond_{[6,10]}(x < 0.2)$, indicating that traffic volume exceeds 0.3 (normalized) during morning hours and drops below 0.2 during mid-day. The parameters $\{0.3, 0.2, 5, 10\}$ are mined from the client's historical data through TeLEx.

It is important to note that the temporal logic inferred from client data is not assumed to represent client behavior precisely. In LogiCP, these STL specifications are utilized as indicators to approximate semantic alignment among clients, thereby guiding the clustering process and mitigating in-cluster heterogeneity. The potential inaccuracy of these logical constraints highlights the importance of incorporating CP, which provides mathematical guarantees to quantify inherent model uncertainty.

4.3 Training-time Optimization: Calibration

The calibration process in LogiCP constitutes a central component of our framework. We detail each step of this process below. CP is an appropriate method for constructing distribution-free prediction sets satisfying that

$$P\left(Y \in \hat{C}(X)\right) \geq 1 - \alpha \tag{11}$$

where $\hat{C}(X)$ represents the prediction interval for a new test point X, and Y is the actual outcome. This guarantees that the true value of Y falls within the computed interval with a probability of at least $1 - \alpha$, thus providing robust predictions against the inherent uncertainty in traffic flow sequences.

As mentioned, the distributed nature of FL keeps data localized but introduces complexities when integrating CP. Hence, we employ a quantile-of-quantile technique (Humbert et al. 2023) to address privacy issues in distributed prediction systems. Specifically, each local client first computes a local quantile of the nonconformity score on its private dataset. Then, each local client shares the computed local quantile to its assigned cluster. Thus, each cluster further computes a quantile of aggregated local quantiles shared by assigned clients to calculate the cluster's CP region ρ_k. Communication between clients and clusters only involves a value of local quantiles and the updated client parameters required by FL. Such a communication process minimizes the exposure of the client dataset while providing rigorous UQ.

In the context of regression tasks relevant to traffic flow prediction, we utilize the fitted absolute residuals as the nonconformity score function $r(X_i, Y_i)$, i.e., $R_i \triangleq r(X_i, Y_i) = |Y_i - \hat{f}(X_i)|$, where R_i is the nonconformity score for i-th data point, and $\hat{f}$ is a predictor trained on historical data.

Definition 3 (Quantile of Quantile Estimator) Given total M data points $D^{(i)} = (X_1^{(i)}, X_2^{(i)}, ..., X_M^{(i)})$ of an individual client i with all total N clients in a distributed system setting, $R^{(i)}$ denotes nonconformity score set calculated on all M data points of i-th client. For any $(\ell, \kappa) \in [[1, M]] \times [[1, N]]$, the Quantile of Quantile Estimator is defined as

$$Q_{(\ell,\kappa)} \triangleq Q_{(\kappa)}\left(Q_{(\ell)}(R^{(1)}), \ldots, Q_{(\ell)}(R^{(N)})\right) \tag{12}$$

where $Q_{(\ell)}(R^{(i)})$ refers to the ℓ-th smallest nonconformity score of data in private calibration dataset of i-th client.

Then, given a confidence level $\alpha \in (0,1)$ for a future time state τ, where $\tau \in \{1,2,...,T\}$, and a trained predictor $\hat{f}$, let $\{(X_j^{(i)}, Y_j^{(i)})\}_{i,j=1}^{N,M}$ and (X,Y) be i.i.d. random variables, where $\hat{Y} = \hat{f}(X)$, the prediction region will be constructed as the following:

$$\hat{C}_{\ell,\kappa}(X) = \left\{Y \in \mathbb{R} : \left\|Y - \hat{Y}\right\| \leq \hat{Q}_{(\ell,\kappa)}\right\} \tag{13}$$

such that

$$P(Y \in \hat{C}_{\ell,\kappa}(X)) \geq 1 - \alpha \tag{14}$$

In words, LogiCP can find the ℓ-th smallest nonconformity score of each local client and the κ-th smallest value of these scores, i.e., (ℓ, κ) such that satisfying Equation (14), where $P(Y \in \hat{C}_{\ell,\kappa}(X))$ is guaranteed to be above $1-\alpha$. The proof and technical details of the solution can be found in Theorem 3.2 in (Humbert et al. 2023).

However, the latest literature has shown that when CP is applied to sequential prediction tasks, the size of the CP region increases along with the length of future time states T (Tibshirani et al. 2019; Zaffran et al. 2022). Consequently, the informativeness derived from CP decreases as the length of the prediction sequence increases. This is because an excessively large CP-derived uncertainty region can negatively impact its guiding effect during model updating for long prediction sequences. To ensure the feasibility of the proposed method for varying prediction sequence lengths, we propose a novel methodology to incorporate the normalized CP region into LogiCP to address this issue.

Similar to the approach detailed in (Zhao et al. 2024), for each local client C_i, we incorporate a normalization term computed from a dedicated normalization dataset D_i^{nor}, which is disjoint from the calibration dataset D_i^{cal}. Both subsets are drawn from the client's private data D_i^{pri}, with $D_i^{nor} \cap D_i^{cal} = \emptyset$. Normalization implementation is designed to consistently establish a CP region with $1-\alpha$ coverage for different prediction lengths, thereby ensuring the feasibility of our proposed algorithm and improving the overall generalizability of LogiCP in various distributed system environments.

Specifically, for each local client i, where $C_i \in C$, suppose there is a set of J sequential data samples $\{Y_\tau^{(j)}\}_{j=1}^{J}$, where $j \in \{1,2,\ldots,J\}$. Let σ_τ be defined over D_i^{nor} as:

$$\sigma_\tau := \max_j \left\|Y_\tau^{(j)} - \hat{Y}_\tau^{(j)}\right\|, \quad \tau \in \{1,2,\ldots,T\}. \tag{15}$$

Moreover, the normalized nonconformity score $R_i^{(j)}$ for each sequential data sample j is calculated as:

$$R_i^{(j)} := \max_{\tau \in \{1,2,\ldots,T\}} \frac{\left\|Y_\tau^{(j)} - \hat{Y}_\tau^{(j)}\right\|}{\sigma_\tau}. \tag{16}$$

For each client i, the local quantile at time τ is computed as:

$$\ell_\tau^{(i)} = \sigma_\tau \cdot Q_{(\ell)}\left(R_i^{(j)}\right), \quad \tau \in \{1,2,\ldots,T\}. \tag{17}$$

Each client C_i transmits a length-T vector of local quantiles $\{\ell_1^{(i)}, \ell_2^{(i)}, \ldots, \ell_T^{(i)}\}$ to its assigned cluster. Each cluster S_k then aggregates these vectors from all assigned clients to compute the cluster-level CP region $\rho_k = \{\hat{Q}_1, \hat{Q}_2, \ldots, \hat{Q}_T\}$.

The CP region $\hat{Q}_\tau$ satisfying the coverage condition at each time τ is then computed as:

$$\hat{Q}_\tau \triangleq Q_{(\kappa)}\left(\ell_\tau^{(1)}, \ell_\tau^{(2)}, \ldots, \ell_\tau^{(N)}\right). \tag{18}$$

In words, the CP region $\hat{C}_{\ell,\kappa}(X)$ for each τ is constructed as follows: (1) for each client i, compute normalized nonconformity scores $R_i^{(j)}$ over its J sequential samples and select the ℓ-th smallest value; (2) scale this value

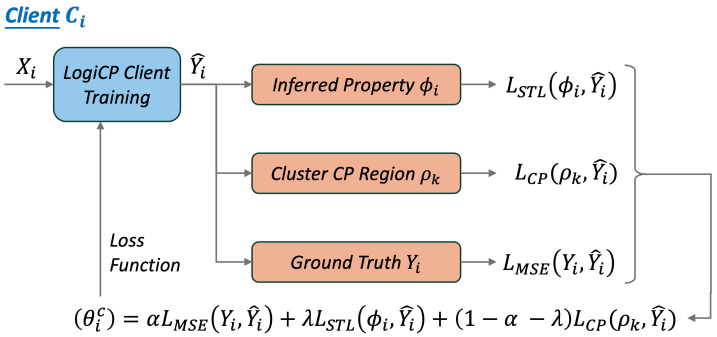

Fig. 3. Client LogiCP model training. (The loss calculation process considers CP regions and inferred properties to optimize both client and cluster model performance.)

by σ_τ to obtain the local quantile $\ell_\tau^{(i)}$; and (3) aggregate the local quantiles across clients by selecting the κ-th smallest value to form the cluster-level CP region at time τ.

4.4 Training-time Optimization: Loss Function

This section outlines the loss function integrated into the model training process. Specifically, it penalizes three critical components: prediction error, error with respect to the quantitative uncertainty range derived from CP, and error with respect to the temporal property inferred from STL. Together, these terms ensure that predictions are not only accurate but also respect uncertainty regions guided by formal inference and conform to client-specific temporal specifications.

The loss function $\mathcal{L}$ for each client C_i is defined as follows:

$$\mathcal{L}(\theta_i^c) := \omega_1 L_{MSE}(Y_i, \hat{Y}_i) + \omega_2 L_{STL}(\phi_i, \hat{Y}_i) + (1 - \omega_1 - \omega_2) L_{CP}(\rho_k, \hat{Y}_i) \tag{19}$$

where:

- $L_{MSE}(Y_i, \hat{Y}_i)$ denotes the Mean Squared Error,
- $L_{STL}(\phi_i, \hat{Y}_i)$ denotes STL alignment loss, which measures the deviation of the predicted sequence $\hat{Y}_i$ from the client's STL specification ϕ_i,
- $L_{CP}(\rho_k, \hat{Y}_i)$ represents the CP loss, which quantifies the deviation between the predicted value $\hat{Y}_i$ and the assigned cluster's CP region ρ_k.

We formally define each loss component below:
MSE Loss:

$$L_{MSE}(Y_i, \hat{Y}_i) = \frac{1}{T} \sum_{\tau=1}^{T} (Y_{i,\tau} - \hat{Y}_{i,\tau})^2 \tag{20}$$

STL Alignment Loss: Using the TeLEx-style robustness function $g(\cdot)$ defined in Section 4.2:

$$L_{STL}(\phi_i, \hat{Y}_i) = \frac{1}{T} \sum_{\tau=1}^{T} g(\phi_i, \hat{Y}_i, \tau) \tag{21}$$

This term penalizes predictions that deviate from the client's inferred STL specification.
CP Loss: Given the cluster's CP region $\rho_k = \{\hat{Q}_1, \ldots, \hat{Q}_T\}$:

$$L_{CP}(\rho_k, \hat{Y}_i) = \frac{1}{T}\sum_{\tau=1}^{T} \max\left(0, |\hat{Y}_{i,\tau} - Y_{i,\tau}| - \hat{Q}_\tau\right) \tag{22}$$

This term penalizes predictions whose errors exceed the CP-derived bounds.

By combining formal logic-guided and uncertainty-aware constraints, this loss function enables LogiCP to produce reliable, personalized predictions. It preserves semantic alignment with client-specific temporal behavior while offering theoretical guarantees on predictive coverage.

4.5 Training-time Optimization: Training Process

The training process of LogiCP integrates all key components introduced in earlier sections to provide formal logic-guided UQ in FL update. At the end of the pretraining phase, each client infers temporal specification ϕ_i and is initially assigned to a cluster based on semantic alignment (Section 4.2). Within each cluster S_k, distributed CP is applied to provide an uncertainty region ρ_k, which captures the ground truth property at a user-defined confidence level (Section 4.3). During training, the CP region ρ_k computed in Section 4.3 is incorporated directly into the loss function $L_{CP}(\rho_k, \hat{Y}_i)$ as a prediction bound constraint. The CP region remains in its original form $\rho_k = \{\hat{Q}_1, \hat{Q}_2, \ldots, \hat{Q}_T\}$ throughout training and inference, without conversion to STL representation.

During each communication round, every client C_i performs n_{local} local update steps using its private dataset D_i^{tr}, minimizing the loss function in Equation (19). This objective aligns model predictions with both CP-derived constraints and STL-inferred specifications. The updated parameters θ_i^c are sent to both the central server and the assigned cluster. The server updates the global model θ^p, and each cluster S_k updates its model θ_k^s through weighted averaging.

After client-wise training, each cluster aggregates public data from its assigned clients P_k and performs $n_{cluster}$ cluster-level training iterations using the same loss function. The initial cluster model θ_k^s is initialized from the weighted average of its assigned clients' parameters during the first clustering. This hybrid training design preserves client personalization while enabling generalization across semantically similar clients. Furthermore, during runtime, LogiCP dynamically assigns clients to semantically aligned clusters using STL inference, eliminating the need for global retraining. This capability enhances both the generalizability and adaptability of FL systems in dynamic, real-world environments.

STL-based clustering and logic-guided CP are repeated in the following communication rounds to adapt to updated client and cluster models. This ensures that clusters remain semantically aligned with updated specifications, and uncertainty regions maintain coverage guarantees. Each communication round, comprising local updates, global and cluster-level aggregation, re-clustering, and CP calibration, is repeated for n_{tr} iterations.

Throughout the entire training process, privacy is rigorously preserved. All client-side learning is conducted exclusively on each client's private dataset D_i^{pri}, and no raw data is shared. The pretraining (D_i^{pre}), normalization (D_i^{nor}), and calibration (D_i^{cal}) data subsets are all derived locally from D_i^{pri}. Both STL inference and distributed CP operate without raw data exchange, where communication between clients and clusters is limited to model parameter updates and scalar quantile values required by the CP calibration process. This design ensures that LogiCP remains fully compliant with standard FL privacy principles while supporting rigorous, logic-guided UQ in a distributed setting.

4.6 Runtime Validation and Correction

After training, we incorporate a runtime validation and correction mechanism into the model's prediction process. This mechanism jointly coordinates quantified uncertainty and inferred property to ensure that the model's

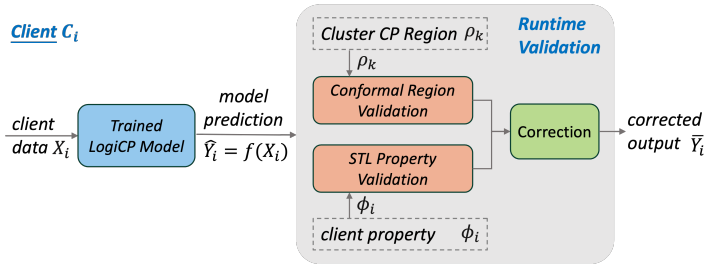

Fig. 4. Runtime validation structure calibrates prediction from the model based on the prediction regions derived from specification mining and CP.

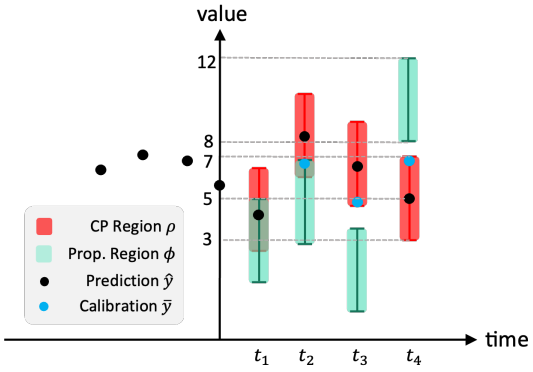

Fig. 5. The visualization of the corrected output $\bar{y}$ from runtime validation structure and the client model prediction $\hat{y}$, represented as blue dot and black dot respectively in the graph. The red region represents the uncertainty region generated from CP, and the green region represents the property region inferred from signal temporal logic.

output is not only within regions that have a marginal guarantee to encompass the real target but also respects the underlying temporal property.

The qualitative semantics of the runtime validation structure are formalized as a function

$$\Psi(\mathcal{Y}; \Phi, T) \mapsto \mathcal{H}, \tag{23}$$

where $\mathcal{H}$ includes the boolean values $\{True\top, False\bot\}$. Specifically, given a sequential predicted result $\hat{y} \in \mathcal{Y}$ and a property $\phi \in \Phi$, if $\hat{y} \models \phi$ at time $t \in T$, then

$$\Psi(\hat{y}; \phi, t) \mapsto \top; \tag{24}$$

otherwise,

$$\Psi(\hat{y}; \phi, t) \mapsto \bot. \tag{25}$$

Other prediction corrections structure, such as one described in An et al. (An et al. 2024), directly correct the prediction within the inferred property region when a violation occurs. These adjustments lack a theoretical guarantee to ensure that the corrected result will have a higher accuracy, particularly when the inferred property is insufficiently precise.

Hence, the goal of our runtime validation structure is to ensure that when prediction $\hat{y}$ violates the inferred property region ϕ, it applies a minimal correction to the prediction to generate a corrected result $\bar{y}$ that satisfies both the inferred property region ϕ and the CP region ρ simultaneously, i.e., $\Psi(\bar{y}; \rho, t) \mapsto \top \wedge \Psi(\bar{y}; \phi, t) \mapsto \top$, where we require the corrected prediction $\bar{y}$ to satisfy both the CP constraint (i.e., $|\bar{y}_t - \hat{y}_t| \leq \hat{Q}_t$, with $\hat{Q}_t$ being

the CP bound derived from region ρ) and the STL specification ϕ simultaneously. As an example in Figure 5, the predicted value $\hat{y}$ violates the inferred property region ϕ at t_2, runtime validation structure calibrated the predicted value to the bound of ϕ.

It is important to note that there exists a situation where the two properties contradict each other, i.e., $\Psi(\bar{y}; \rho, t) \mapsto \top \Leftrightarrow \Psi(\bar{y}; \phi, t) \mapsto \bot$. At t_3 in Figure 5, the predicted value $\hat{y}$ violates the inferred property region, whereas these two regions conflict. In such cases, the runtime validation structure prioritizes the region derived from CP, i.e., $\Psi(\bar{y}; \rho, t) \mapsto \top \wedge \Psi(\bar{y}; \phi, t) \mapsto \bot$. This is because the CP region has a theoretical guarantee to contain the real value, offering a more reasonable direction for calibration. Thus, when two regions have conflict, the runtime validation structure will, at most, calibrate the prediction to the bounds of the CP region.

As a specific example, assuming CP region ρ at t_4 reveals that the deviation between predicted $\hat{y}$ and the real value y is 2. It suggests that for an output $\hat{y} = 5$ from the model, the CP region has a CP region $[3, 7]$. The inferred STL semantics property $G_{[t_4]}(y \geq 8) \wedge G_{[t_4]}(y \leq 12)$ suggests a property region $\phi = [8, 12]$. In this case, when the two regions contradict each other, the runtime validation structure calibrates the prediction to the boundary of the CP region, resulting in $\bar{y} = 7$.

5 Evaluation

Our evaluation focuses on the following primary objectives: (G1) the effectiveness of LogiCP in **improving FL prediction accuracy** under client heterogeneity, in comparison to baseline FL approaches; (G2) the **reliability of the runtime validation structure** in enhancing prediction accuracy in practical scenarios; and (G3) the **scalability** of LogiCP, by evaluating its ability to maintain both predictive accuracy and reliable uncertainty quantification across varying numbers of clients in large-scale distributed environments; (G4) the capability of LogiCP to **capture model-inherent uncertainty under varying scales**.

5.1 Datasets

We evaluate the performance of LogiCP using three real-world datasets that exhibit diverse forms of temporal heterogeneity: a publicly available traffic flow dataset from the Federal Highway Administration (FHWA 2016), a worldwide temperature dataset from the University of Dayton (University of Dayton 2002), and an electricity consumption dataset (Trindade 2015).

The FHWA dataset was selected for its origin from real-world and broad coverage of traffic patterns across different geographic regions and time periods, making it suitable for assessing model performance under temporal and spatial heterogeneity. The data is collected from 384 sensors in 15 states. Sensor types include Video Image Processors, Radar Sensors, and infrared sensors, deployed across highways, interstates, and town roads. The worldwide temperature dataset (denoted as CT) provides average daily temperature records from 157 U.S. cities and 167 international cities, spanning from 1995 to 2002. Unlike the FHWA dataset, which exhibits strong and consistent temporal trends characteristic of structured traffic patterns, the CT dataset contains greater variability and fluctuations in data distribution due to climate diversity and regional anomalies.

The Electric dataset contains electricity consumption records from 370 clients, with measurements recorded at 15-minute intervals. Unlike the FHWA and CT datasets, which reflect traffic dynamics and climate-driven patterns respectively, the Electric dataset captures heterogeneous consumption behaviors across residential and commercial users. These behaviors vary significantly with user types, seasonal factors, and daily routines, introducing a distinct form of temporal heterogeneity.

Together, these three datasets enable a comprehensive evaluation of LogiCP's generalizability across diverse temporal prediction domains, ranging from highly structured sequences to irregular and user-dependent patterns, aligning with our goal of addressing data heterogeneity in practical federated learning settings.

Table 1. Comparison of MSE and inferred specification satisfaction rate on FHWA Dataset.

Method	RNN		GRU		LSTM		Transformer	
	MSE	γ	MSE	γ	MSE	γ	MSE	γ
FedAvg	0.129 ± 0.028	78.58%	0.159 ± 0.027	82.39%	0.116 ± 0.026	85.26%	0.023 ± 0.006	75.83%
IFCA	0.113 ± 0.025	78.76%	0.138 ± 0.026	82.65%	0.115 ± 0.026	85.05%	0.051 ± 0.007	75.14%
IFCA-S	0.105 ± 0.026	81.11%	0.140 ± 0.025	82.66%	0.122 ± 0.030	85.89%	0.056 ± 0.007	74.47%
CP-IFCA	0.121 ± 0.025	80.57%	0.153 ± 0.026	84.08%	0.178 ± 0.026	84.15%	0.039 ± 0.004	76.69%
CP-IFCA-S	0.127 ± 0.028	80.10%	0.146 ± 0.025	84.25%	0.164 ± 0.032	85.07%	0.045 ± 0.007	75.73%
Fed-Bayes	0.051 ± 0.014	86.17%	0.093 ± 0.024	83.67%	0.102 ± 0.026	82.68%	0.596 ± 0.005	74.30%
FedSTL	0.061 ± 0.013	85.95%	0.120 ± 0.025	86.59%	0.184 ± 0.025	86.49%	0.022 ± 0.005	78.80%
FedSTL-S	0.065 ± 0.015	85.80%	0.112 ± 0.024	86.79%	0.170 ± 0.024	86.66%	0.019 ± 0.005	78.24%
FedSTL-T	0.051 ± 0.010	**90.59%**	0.092 ± 0.018	**92.40%**	0.144 ± 0.020	**90.28%**	0.015 ± 0.005	86.20%
LogiCP	0.048 ± 0.009	85.09%	0.109 ± 0.021	87.16%	0.091 ± 0.020	86.94%	0.0027 ± 0.0014	83.63%
LogiCP-S	0.052 ± 0.011	85.08%	0.110 ± 0.022	86.44%	0.094 ± 0.022	86.28%	0.0038 ± 0.0022	78.94%
LogiCP-T	**0.030 ± 0.008**	89.75%	**0.062 ± 0.017**	89.25%	**0.060 ± 0.017**	89.39%	**0.0021 ± 0.0014**	**88.89%**

5.2 Experiment Setup

We design two experimental settings that directly correspond to our primary evaluation objectives [1].

Specifically, to evaluate G1 and G2, we first consider a general multi-client environment consisting of 100 clients applied to both the FHWA dataset and the CT dataset. This configuration is used to evaluate **LogiCP's effectiveness in improving FL prediction accuracy, as well as the reliability of the proposed runtime validation structure under a heterogeneous data setting**. We implement and compare multiple neural architectures, including RNN, GRU, LSTM, and Transformer models, under this multi-client framework to assess **generalizability across model architectures and datasets**.

Second, targeting G3 and G4, we evaluate **scalability and uncertainty quantification performance** using the FHWA dataset with the Transformer model under varying client scales. Specifically, we test LogiCP under environments with 1, 50, 100, and 300 clients. This setting is designed to assess **LogiCP's ability to maintain both prediction accuracy and reliable CP-based uncertainty quantification across different deployment scales**. The CP satisfaction rate (v), introduced in the problem formulation, is used to measure how well the predicted CP regions capture model uncertainty.

In all experiments, each client's local training dataset is divided equally, with 50% allocated to D_i^{pri} and 50% to D_i^{pub}. Models are trained to forecast the future 24 timestamps based on a 120 historical timestamps. All clients participate in each communication round across all settings unless a specific participation rate is specified. Training is performed using Stochastic Gradient Descent (SGD) with a momentum of 0.9, a fixed learning rate of 1×10^{-3}, and a batch size of 64. For all experiments involving CP, we set the error rate $\alpha = 0.1$. Experiments are conducted on 128 GB RAM, AMD Ryzen Threadripper Pro 7975WX, and NVIDIA RTX 6000 Ada.

5.3 Baseline Models

We compare the performance of LogiCP with the following FL baselines: (1) FedAvg (McMahan et al. 2017) serves as a standard baseline representing traditional FL without formal reasoning and uncertainty quantification.

[1]The full implementation of LogiCP is available at https://github.com/AICPS-Lab/LogiCP

The central server iteratively aggregates parameters from all local clients through weighted averaging at each communication round, without accounting for specification alignment or uncertainty. This baseline allows us to isolate the impact of both STL-based clustering and CP-derived uncertainty quantification. (2) IFCA (Ghosh et al. 2020) is included to assess the performance of FL with client clustering but without formal logic inference. Clients are assigned to the cluster whose model minimizes prediction loss on the client's local data, and the cluster models are updated based on assigned clients. Comparing LogiCP with IFCA highlights the enhancement derived from formal logic-guided uncertainty quantification over purely loss-based partitioning. (3) CP-IFCA is a modified version of IFCA where distributed CP is applied within each cluster. This baseline setting evaluates the performance of applying CP-based uncertainty quantification **without** STL-guided clustering. It enables a direct comparison with LogiCP to assess the enhancement of formal logic guidance in uncertainty quantification. (4) FedSTL (An et al. 2024) infers formal properties into FL updates, but does **not** incorporate uncertainty quantification. Comparing LogiCP with FedSTL highlights the importance of combining formal logic inference with rigorous uncertainty quantification to enhance the reliability of predictions. (5) Fed-Bayes (BNN-based personalization). We additionally implement a Bayesian neural network (BNN)–based personalized FL method (denoted as *Fed-Bayes*) following (X. Zhang, Y. Li, et al. 2022). In this setting, each client maintains both a global and a personalized model, with weights represented as variational distributions $w \sim \mathcal{N}(\mu, \text{softplus}(\rho)^2)$. We adopt the hyperparameter settings reported in the authors' open-sourced implementation, including a KL divergence weight $\zeta = 10.0$, weight scale 0.1, and ρ offset -3.0.

5.4 Evaluation Metrics

We primarily employ three metrics to assess LogiCP performance under each experimental setting: **(1) Mean Squared Error (MSE):** MSE measures the overall prediction accuracy by quantifying the average discrepancy between predicted and actual values, providing a straightforward measure of prediction accuracy. **(2) Satisfaction Rate of Inferred Specification (γ):** This metric computes the percentage of model predictions that comply with the STL-inferred specification ϕ_i for each client. It reflects how well the inferred specification captures the client's temporal pattern, helping to assess the consistency of formal inference under heterogeneous conditions. **(3) CP Guarantee Satisfaction Rate (v):** (v) is defined as the percentage of ground truth values that fall within the CP-derived uncertainty region. It assesses the model's ability to capture uncertainty accurately, measuring the reliability of uncertainty quantification. A higher v indicates stronger alignment between the uncertainty region and the actual outcomes, reflecting better representation of the inherent uncertainty of the model.

5.5 Enhanced Prediction Accuracy

The experimental results on the FHWA, CT, and Electric datasets of all FL approaches for each neural network model are presented in Table 1, Table 2, and Table 3, respectively. Here, "-S" represents the evaluation of cluster models, while "-T" refers to predictions calibrated by the runtime validation structure described in Section 4.6.

For the FHWA dataset, LogiCP outperforms all baseline methods in client-level MSE by 50%, 23%, 35%, and 91% on the RNN, GRU, LSTM, and Transformer models, respectively. The most significant improvement is observed with the Transformer model, where LogiCP reduces client-wise MSE by approximately 88%, 93%, 95%, and 88% compared to FedSTL, CP-IFCA, IFCA, and FedAvg, respectively, with 100 participating clients. Similarly, on the CT dataset, LogiCP achieves consistent improvements on client-level MSE across all model types by 33%, 24%, 43%, and 46% on the RNN, GRU, LSTM, and Transformer models, respectively. On the Electric dataset, LogiCP consistently outperforms all baseline methods on client-level MSE across all model types by 32%, 40%, 58%, and 17% on the RNN, GRU, LSTM, and Transformer models, respectively. Specifically, LogiCP outperforms FedAvg by 51%, IFCA by 44%, CP-IFCA by 40%, and Fed-STL by 37% on average across all four model architectures. Compared to Fed-Bayes, LogiCP achieves an average improvement of 11% on RNN, GRU, and LSTM models.

Table 2. Comparison of MSE and inferred specification satisfaction rate on CT Dataset.

Methods	RNN		GRU		LSTM		Transformer	
	MSE	γ	MSE	γ	MSE	γ	MSE	γ
FedAvg	0.150 ± 0.019	74.96%	0.210 ± 0.022	76.69%	0.449 ± 0.041	58.05%	0.359 ± 0.039	66.25%
IFCA	0.118 ± 0.011	76.49%	0.145 ± 0.014	78.73%	0.264 ± 0.026	63.05%	0.157 ± 0.017	66.44%
IFCA-S	0.104 ± 0.015	72.90%	0.133 ± 0.018	75.35%	0.118 ± 0.022	77.50%	0.133 ± 0.016	73.98%
CP-IFCA	0.194 ± 0.016	88.33%	0.127 ± 0.011	93.24%	0.149 ± 0.027	90.61%	0.418 ± 0.024	76.90%
CP-IFCA-S	0.212 ± 0.031	89.00%	0.129 ± 0.016	93.87%	0.146 ± 0.031	91.25%	0.429 ± 0.032	76.04%
Fed-Bayes	0.115 ± 0.013	84.92%	0.132 ± 0.012	83.32%	0.147 ± 0.013	80.79%	0.394 ± 0.026	70.45%
Fed-STL	0.111 ± 0.010	69.67%	0.127 ± 0.013	72.30%	0.127 ± 0.013	71.97%	0.110 ± 0.012	70.40%
Fed-STL-S	0.098 ± 0.008	71.84%	0.115 ± 0.011	74.73%	0.129 ± 0.013	73.95%	0.114 ± 0.012	72.99%
Fed-STL-T	0.127 ± 0.013	81.21%	0.141 ± 0.014	78.16%	0.140 ± 0.014	81.90%	0.114 ± 0.012	73.00%
LogiCP	0.091 ± 0.007	88.65%	0.111 ± 0.010	92.66%	0.110 ± 0.010	91.16%	0.104 ± 0.011	92.11%
LogiCP-S	0.100 ± 0.008	**92.41%**	0.125 ± 0.012	94.88%	0.127 ± 0.013	**94.25%**	0.135 ± 0.013	92.07%
LogiCP-T	**0.090 ± 0.008**	92.38%	**0.111 ± 0.010**	**96.07%**	**0.108 ± 0.010**	93.32%	**0.104 ± 0.011**	**95.29%**

LogiCP-T consistently outperforms Fed-Bayes, achieving an average MSE reduction of 24% across all datasets. The 40% improvement over CP-IFCA demonstrates the significance of STL-guided clustering for CP-based uncertainty quantification in heterogeneous environments. While both methods employ conformal prediction, LogiCP leverages STL specifications to measure distribution alignment among clients and reduce intra-cluster heterogeneity, resulting in more accurate and informative prediction intervals.

Additionally, LogiCP-S achieves an average 54% reduction in MSE compared to CP-IFCA-S and a 37% reduction in MSE compared to Fed-STL-S on FHWA dataset. These improvements validate LogiCP's ability to mitigate heterogeneity and enhance client prediction accuracy significantly while preserving personalization. LogiCP-S achieves an average 37% reduction in MSE compared to CP-IFCA-S and a 39% reduction compared to Fed-STL-S on the Electric dataset. This result further validates LogiCP's ability to mitigate heterogeneity while maintaining personalized prediction accuracy across diverse scenarios.

Furthermore, we observe that the MSE of predictions calibrated by the runtime validation structure (**LogiCP-T**) further improves upon LogiCP, achieving the highest accuracy across all evaluated settings. In terms of specification satisfaction rate (γ), LogiCP consistently outperforms all baselines, except for FedSTL-T on the FHWA dataset, which achieves a comparable or slightly higher γ. This is because FedSTL-T calibrates predictions solely with respect to the inferred STL properties ϕ, which results in higher γ. However, LogiCP-T still achieves lower MSE than FedSTL-T, demonstrating that the inferred specifications alone may not reliably capture personalized features. Hence, the result indicates that by integrating STL-guided UQ in runtime, LogiCP more effectively adapts to individual client characteristics and ensures reliable model performance across diverse settings.

The evaluation results show that LogiCP achieves optimal performance on the FHWA, CT, and Electric datasets, which exhibit distinct temporal characteristics. **These outcomes highlight LogiCP's consistency across diverse data patterns, validating its generalizability under heterogeneous scenarios.** These findings support our core objective to develop a framework that can maintain high personalization and prediction reliability across various practical contexts.

Table 3. Comparison of MSE and inferred specification satisfaction rate on Electric Dataset.

Methods	RNN		GRU		LSTM		Transformer	
	MSE	γ	MSE	γ	MSE	γ	MSE	γ
FedAvg	0.142 ± 0.011	92.69%	0.202 ± 0.009	96.84%	0.321 ± 0.018	98.55%	0.089 ± 0.006	88.13%
IFCA	0.110 ± 0.007	97.84%	0.162 ± 0.008	94.72%	0.322 ± 0.018	98.59%	0.082 ± 0.007	83.44%
IFCA-S	0.116 ± 0.009	96.05%	0.170 ± 0.010	93.95%	0.326 ± 0.019	98.49%	0.085 ± 0.006	98.49%
CP-IFCA	0.102 ± 0.008	98.11%	0.133 ± 0.009	95.21%	0.316 ± 0.021	99.04%	0.082 ± 0.007	87.34%
CP-IFCA-S	0.107 ± 0.010	96.92%	0.136 ± 0.010	95.17%	0.307 ± 0.020	98.79%	0.082 ± 0.006	89.81%
Fed-Bayes	0.074 ± 0.006	96.43%	0.089 ± 0.007	97.68%	0.115 ± 0.012	96.67%	0.892 ± 0.027	93.60%
Fed-STL	0.097 ± 0.007	94.14%	0.139 ± 0.008	98.06%	0.278 ± 0.015	97.00%	0.076 ± 0.006	84.65%
Fed-STL-S	0.104 ± 0.008	92.29%	0.163 ± 0.009	95.91%	0.296 ± 0.019	96.16%	0.083 ± 0.006	84.74%
Fed-STL-T	0.095 ± 0.007	97.74%	0.130 ± 0.008	97.16%	0.246 ± 0.013	96.42%	0.076 ± 0.006	84.86%
LogiCP	0.068 ± 0.006	96.22%	0.081 ± 0.006	96.42%	0.096 ± 0.006	98.10%	0.068 ± 0.006	91.81%
LogiCP-S	0.071 ± 0.006	98.80%	0.083 ± 0.006	96.45%	0.106 ± 0.008	96.07%	0.073 ± 0.006	91.34%
LogiCP-T	**0.068 ± 0.006**	96.63%	**0.081 ± 0.006**	97.11%	**0.096 ± 0.006**	98.54%	**0.067 ± 0.006**	93.04%

Table 4. Comparison of MSE and CP satisfaction rate under the Transformer Model Across Different Client Scales. Note that CP satisfaction rates are not applicable for approaches that do not integrate with CP.

Method	300 Clients		100 Clients		50 Clients		1 Client	
	MSE	v	MSE	v	MSE	v	MSE	v
FedAvg	0.018 ± 0.001	N/A	0.023 ± 0.006	N/A	0.046 ± 0.003	N/A	0.034	N/A
IFCA	0.022 ± 0.001	N/A	0.051 ± 0.007	N/A	0.056 ± 0.012	N/A	0.057	N/A
IFCA-S	0.019 ± 0.001	N/A	0.056 ± 0.007	N/A	0.060 ± 0.014	N/A	N/A	N/A
CP-IFCA	0.038 ± 0.002	96.53%	0.039 ± 0.004	96.83%	0.041 ± 0.009	96.79%	0.039	98.92%
CP-IFCA-S	0.040 ± 0.003	96.50%	0.045 ± 0.007	96.51%	0.049 ± 0.014	96.46%	0.032	100.00%
Fed-STL	0.019 ± 0.001	N/A	0.022 ± 0.005	N/A	0.026 ± 0.009	N/A	0.025	N/A
Fed-STL-S	0.019 ± 0.001	N/A	0.019 ± 0.005	N/A	0.023 ± 0.010	N/A	N/A	N/A
Fed-STL-T	0.019 ± 0.001	N/A	0.015 ± 0.005	N/A	0.018 ± 0.009	N/A	0.020	N/A
LogiCP	0.0034 ± 0.0009	97.20%	0.0027 ± 0.0014	98.14%	0.0026 ± 0.0018	98.54%	0.0084	100.00%
LogiCP-S	0.0034 ± 0.0008	**98.36%**	0.0038 ± 0.0022	98.65%	0.0041 ± 0.0033	98.72%	0.0079	100.00%
LogiCP-T	**0.0024 ± 0.0008**	98.18%	**0.0021 ± 0.0014**	**98.77%**	**0.0021 ± 0.0017**	**98.95%**	**0.0050**	**100.00%**

5.6 Consistency and Scalability of LogiCP

We evaluate LogiCP's performance under varying client scales, with results summarized in Table 4. Across all configurations, LogiCP consistently outperforms baseline models in both prediction accuracy (MSE) and uncertainty quantification (CP satisfaction rate). Notably, in the most challenging 300-client setting, LogiCP improves MSE by 82%, 91%, 85%, and 81%, respectively, compared to baselines. Moreover, LogiCP maintains stable performance as the number of clients increases, with MSE differences remaining within the thousandths range

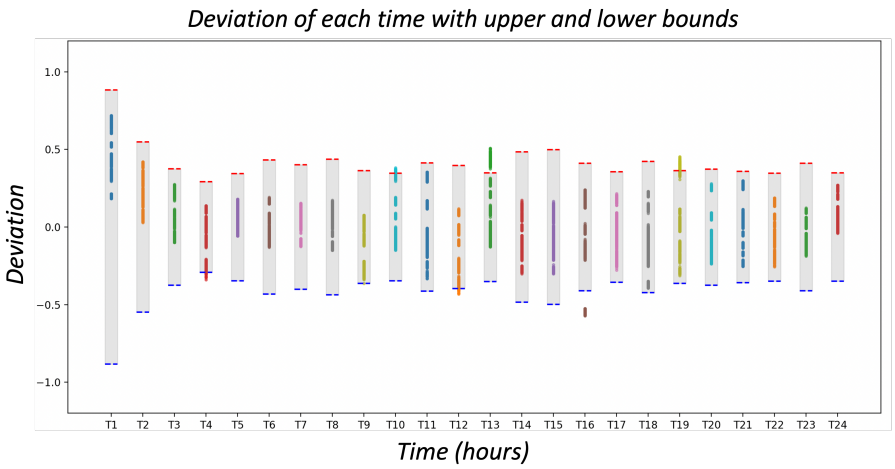

Fig. 6. Visualization of one cluster prediction across all attributes with CP region for RNN model on FHWA dataset.

across different scales. **These results indicate that LogiCP continues to capture personalized features for all clients, demonstrating its scalability and consistency in preserving personalization under increasing client heterogeneity**. Furthermore, LogiCP-T consistently delivers the highest prediction accuracy across all scales, further highlighting the practicality of LogiCP in real-world, large-scale federated environments.

Additionally, LogiCP achieves consistently higher CP satisfaction rates than CP-IFCA across all scale settings. A higher satisfaction rate reflects that the CP region encompasses a larger proportion of true values, indicating a more reliable uncertainty estimation. **This outcome validates LogiCP's core objective of guiding uncertainty quantification through formal inference, reducing intra-cluster heterogeneity to achieve more reliable uncertainty quantification.**

5.7 Partial Participation Analysis

To evaluate LogiCP's effectiveness under realistic FL conditions, we conduct experiments under varying participation rates $\{0.3, 0.5, 0.7, 0.9\}$ on the Electric dataset with 100 clients using the RNN model. The results are presented in Table 5.

LogiCP consistently outperforms all baselines across all participation settings, achieving an average MSE reduction of 31% compared to baseline methods. Notably, even at the lowest participation rate of 0.3, LogiCP maintains its performance advantage, achieving a 29% average MSE improvement over all baselines. **This demonstrates that LogiCP's STL-guided clustering and CP-based uncertainty quantification remain effective under FL partial participation conditions**.

Additionally, LogiCP outperforms CP-IFCA by an average of 31% in MSE across all participation settings, highlighting the significance of STL-guided clustering in CP-based uncertainty quantification. LogiCP outperforms Fed-STL by an average of 28% in MSE across all participation settings, revealing the significance of integrating rigorous uncertainty quantification in model updating.

5.8 Ablation Study

5.8.1 Ablation Study: Runtime Correction Strategies. We compare two runtime correction strategies when the CP region and STL property region conflict: (1) LogiCP-CP, which prioritizes the CP boundary, and (2) LogiCP-STL,

Table 5. Comparison of MSE and STL satisfaction rate of RNN on Electric Dataset with different participation ratio.

Methods	0.3		0.5		0.7		0.9	
	MSE	γ	MSE	γ	MSE	γ	MSE	γ
FedAvg	0.174 ± 0.013	91.07%	0.167 ± 0.013	91.61%	0.156 ± 0.012	92.27%	0.142 ± 0.011	92.67%
IFCA	0.199 ± 0.012	88.74%	0.138 ± 0.009	91.82%	0.130 ± 0.009	92.86%	0.132 ± 0.008	92.27%
IFCA-S	0.119 ± 0.010	91.96%	0.115 ± 0.009	92.50%	0.116 ± 0.009	91.45%	0.114 ± 0.009	92.12%
CP-IFCA	0.103 ± 0.040	95.25%	0.103 ± 0.039	95.33%	0.103 ± 0.040	95.15%	0.102 ± 0.039	94.18%
CP-IFCA-S	0.114 ± 0.059	94.46%	0.113 ± 0.054	94.42%	0.112 ± 0.049	94.72%	0.112 ± 0.061	92.88%
Fed-Bayes	0.076 ± 0.006	95.01%	0.074 ± 0.006	96.34%	0.074 ± 0.006	96.25%	0.074 ± 0.006	96.42%
Fed-STL	0.100 ± 0.007	93.69%	0.099 ± 0.007	93.77%	0.098 ± 0.007	93.94%	0.098 ± 0.007	94.05%
Fed-STL-S	0.106 ± 0.008	91.95%	0.105 ± 0.007	91.93%	0.104 ± 0.008	92.11%	0.104 ± 0.008	92.15%
Fed-STL-T	0.098 ± 0.007	97.42%	0.097 ± 0.007	**97.48%**	0.096 ± 0.007	**97.63%**	0.096 ± 0.007	**97.66%**
LogiCP	0.075 ± 0.006	96.61%	0.071 ± 0.006	96.34%	0.071 ± 0.006	95.42%	0.068 ± 0.006	95.95%
LogiCP-S	0.077 ± 0.006	**97.78%**	0.075 ± 0.006	97.08%	0.073 ± 0.006	97.10%	0.071 ± 0.006	96.70%
LogiCP-T	**0.075 ± 0.006**	96.98%	**0.071 ± 0.006**	96.73%	**0.071 ± 0.006**	95.87%	**0.068 ± 0.006**	96.30%

Table 6. Ablation study of runtime correction strategies on FHWA dataset with 100 clients. LogiCP-STL prioritizes the STL property boundary; LogiCP-CP prioritizes the CP boundary.

Methods	RNN		GRU		LSTM		Transformer	
	MSE	γ	MSE	γ	MSE	γ	MSE	γ
LogiCP	0.052 ± 0.007	84.39%	0.101 ± 0.022	80.28%	0.081 ± 0.019	83.15%	0.0035 ± 0.0008	83.20%
LogiCP-STL	0.034 ± 0.006	**89.39%**	0.070 ± 0.017	85.56%	0.058 ± 0.015	**88.64%**	0.0026 ± 0.0007	**88.63%**
LogiCP-CP	**0.034 ± 0.006**	89.37%	**0.070 ± 0.017**	**85.56%**	**0.058 ± 0.015**	88.62%	**0.0026 ± 0.0007**	88.47%

which prioritizes the STL property boundary. Table 6 presents the results across all four model architectures on the FHWA dataset with 100 clients.

The two strategies achieve comparable performance, with MSE differences occurring at the fifth decimal place and nearly identical specification satisfaction rates (γ). A likely explanation is that conflicts between the CP region and STL property region—where the two do not overlap—are infrequent in practice. Given this empirical equivalence, LogiCP adopts the CP-priority strategy as it provides correction grounded in a mathematically guaranteed coverage bound.

5.8.2 Ablation Study: Calibration Analysis. To provide deeper insights into the CP calibration of LogiCP, we compare LogiCP (STL-based clustering) with CP-IFCA (loss-based clustering with CP) under varying target coverage rates α. This comparison evaluates how different clustering strategies affect calibration quality, coverage reliability, and interval tightness across different confidence levels. The detailed results are presented in Table 7.

α-sweep Analysis. We evaluate LogiCP under varying CP error rates $\alpha \in \{0.10, 0.15, 0.20, 0.25, 0.30\}$, corresponding to target coverage levels of 90%, 85%, 80%, 75%, and 70%, respectively. **LogiCP consistently achieves empirical coverage exceeding the target $1 - \alpha$ across all settings.** For example, at a high target coverage of 90% ($\alpha = 0.10$), LogiCP attains 98.01% empirical coverage, while at a lower target of 70% ($\alpha = 0.30$), it maintains 95.32%—comfortably above the threshold. These results demonstrate the reliability of our distributed CP implementation and confirm that LogiCP provides valid prediction intervals with rigorous coverage guarantees.

Per-client and Per-cluster Coverage. We analyze the coverage distribution across individual client models and cluster models by examining the standard deviation of coverage rates. The standard deviation reflects the uniformity of uncertainty quantification: lower standard deviation indicates more consistent coverage guarantees across participants.

Results show that LogiCP-S (cluster models) achieves more uniform coverage, with standard deviations ranging from 0.30% to 0.65%. This indicates that LogiCP's cluster-level aggregation effectively reduces coverage variability. Compared to CP-IFCA-S (standard deviation 0.30%–0.72%), LogiCP-S achieves comparable uniformity while maintaining significantly higher coverage rates and lower MSE. **The reduced variance in LogiCP-S suggests that STL-guided clustering effectively groups clients with similar temporal patterns, leading to more homogeneous intra-cluster behavior and more reliable uncertainty quantification.**

Interval Width Analysis. The evaluation results reveal that LogiCP produces significantly narrower prediction intervals while maintaining coverage guarantees. At the 90% target coverage level, LogiCP achieves an average interval width of 0.129, compared to 0.396 for CP-IFCA—a 67% reduction in interval width. This substantial improvement persists across all coverage levels, with LogiCP achieving an average width reduction of 69% compared to CP-IFCA. Narrower intervals indicate more informative uncertainty quantification, where the model can provide tighter bounds while still guaranteeing valid coverage. This improvement stems from **LogiCP's STL-guided clustering, which reduces intra-cluster heterogeneity and allows CP to produce more precise, client-specific prediction regions** rather than overly conservative intervals that must accommodate diverse client behaviors.

Table 7. Calibration analysis across different target coverage levels of 50 clients on FHWA dataset with Transformer model.

		Target Coverage Level $(1-\alpha)$				
Method	**Metric**	**90%**	**85%**	**80%**	**75%**	**70%**
CP-IFCA	MSE ↓	0.0416 ± 0.0324	0.0416 ± 0.0324	0.0421 ± 0.0357	0.0421 ± 0.0357	0.0415 ± 0.0324
	Coverage ↑	97.43 ± 1.67%	96.78 ± 1.70%	97.10 ± 1.62%	96.51 ± 1.62%	93.99 ± 1.95%
	Width ↓	0.396 ± 0.049	0.383 ± 0.049	0.392 ± 0.049	0.380 ± 0.050	0.341 ± 0.046
CP-IFCA-S	MSE ↓	0.0439 ± 0.0235	0.0439 ± 0.0234	0.0447 ± 0.0191	0.0438 ± 0.0197	0.0438 ± 0.0236
	Coverage ↑	97.22 ± 0.72%	96.50 ± 0.59%	96.96 ± 0.30%	96.24 ± 0.50%	93.43 ± 0.40%
	Width ↓	0.396 ± 0.049	0.383 ± 0.049	0.392 ± 0.049	0.380 ± 0.050	0.341 ± 0.046
LogiCP	MSE ↓	**0.0029 ± 0.0020**	**0.0029 ± 0.0020**	**0.0029 ± 0.0020**	**0.0029 ± 0.0020**	**0.0028 ± 0.0020**
	Coverage ↑	98.01 ± 3.98%	97.45 ± 4.66%	97.59 ± 4.52%	97.27 ± 4.86%	**95.32 ± 6.73%**
	Width ↓	**0.129 ± 0.005**	**0.119 ± 0.007**	**0.121 ± 0.008**	**0.117 ± 0.009**	**0.099 ± 0.008**
LogiCP-S	MSE ↓	0.0037 ± 0.0007	0.0037 ± 0.0007	0.0037 ± 0.0007	0.0037 ± 0.0007	0.0036 ± 0.0006
	Coverage ↑	**98.35 ± 0.30%**	**97.78 ± 0.43%**	97.98 ± 0.39%	**97.63 ± 0.43%**	95.06 ± 0.65%
	Width ↓	0.129 ± 0.005	0.119 ± 0.007	0.121 ± 0.008	0.117 ± 0.009	0.099 ± 0.008

5.8.3 Ablation Study: Clustering Strategy. We replace STL-guided clustering with two alternative strategies while retaining all other components: (1) Loss-based Clustering, where clients are assigned to the cluster whose model minimizes prediction loss on the client's local data; and (2) Random Clustering, where clients are randomly assigned to clusters without considering data characteristics. Results are presented in Table 8.

LogiCP achieves substantially lower MSE than both baseline clustering strategies across all target coverage levels, reducing MSE by 83% compared to both Random and Loss-based Clustering. Additionally, LogiCP achieves an average 73% reduction in interval width while maintaining empirical coverage above the target threshold. **These results demonstrate that STL-based semantic clustering effectively reduces intra-cluster heterogeneity, enabling CP to produce precise, client-specific prediction regions rather than overly conservative intervals.**

Table 8. Ablation study: Impact of clustering strategies on CP. Comparison of Random Clustering, Loss-based Clustering, and LogiCP (STL-based Clustering) across different target coverage levels on FHWA dataset with Transformer model.

		Target Coverage Level $(1-\alpha)$				
Method	**Metric**	**90%**	**85%**	**80%**	**75%**	**70%**
Random-Clustering	MSE ↓	0.0172 ± 0.0274	0.0171 ± 0.0275	0.0172 ± 0.0276	0.0171 ± 0.0276	0.0169 ± 0.0274
	Coverage ↑	97.81 ± 2.85%	97.03 ± 3.54%	96.79 ± 3.63%	96.02 ± 4.17%	92.55 ± 6.11%
	Width ↓	0.486 ± 0.034	0.451 ± 0.027	0.443 ± 0.028	0.423 ± 0.029	0.359 ± 0.020
Loss-based-Clustering	MSE ↓	0.0179 ± 0.0305	0.0167 ± 0.0266	0.0174 ± 0.0336	0.0165 ± 0.0271	0.0171 ± 0.0331
	Coverage ↑	97.76 ± 3.29%	97.38 ± 3.51%	**97.60 ± 3.54%**	96.39 ± 3.93%	93.18 ± 4.91%
	Width ↓	0.491 ± 0.015	0.469 ± 0.026	0.467 ± 0.035	0.435 ± 0.033	0.368 ± 0.027
LogiCP	MSE ↓	**0.0029 ± 0.0020**	**0.0029 ± 0.0020**	**0.0029 ± 0.0020**	**0.0029 ± 0.0020**	**0.0028 ± 0.0020**
	Coverage ↑	**98.01 ± 3.98%**	**97.45 ± 4.66%**	97.59 ± 4.52%	**97.27 ± 4.86%**	**95.32 ± 6.73%**
	Width ↓	**0.129 ± 0.005**	**0.119 ± 0.007**	**0.121 ± 0.008**	**0.117 ± 0.009**	**0.099 ± 0.008**

Table 9. Ablation study evaluating the contribution of CP-aware training on FHWA dataset with 100 clients. No-CP-Loss: trained without CP; Runtime-CP-Only: trained without CP but CP applied at runtime; LogiCP: CP-aware training; LogiCP-T: CP-aware training and runtime validation.

Methods	**RNN**		**GRU**		**LSTM**		**Transformer**	
	MSE	γ	MSE	γ	MSE	γ	MSE	γ
No-CP-Loss	0.096 ± 0.019	81.54%	0.115 ± 0.020	82.80%	0.155 ± 0.026	81.70%	0.033 ± 0.005	77.33%
Runtime-CP-Only	0.072 ± 0.016	**97.39%**	0.079 ± 0.017	87.65%	0.099 ± 0.021	87.52%	0.020 ± 0.005	80.84%
LogiCP	0.052 ± 0.007	84.39%	0.101 ± 0.022	80.28%	0.081 ± 0.019	83.15%	0.0035 ± 0.0008	83.20%
LogiCP-T	**0.034 ± 0.006**	89.37%	**0.070 ± 0.017**	**87.65%**	**0.058 ± 0.015**	**88.62%**	**0.0026 ± 0.0007**	**88.47%**

5.8.4 Ablation 2: CP-Aware Training. We remove the CP loss term (L_{CP}) from the training objective while retaining prediction loss (L_{MSE}), property loss (L_{STL}), and STL-based clustering. We evaluate two settings: (1) No-CP-Loss, where no CP is incorporated during training or runtime; and (2) Runtime-CP-Only, where the model is trained without CP loss but CP calibration is applied at runtime. Results are presented in Table 9.

Runtime-CP-Only achieves an average 33% MSE reduction compared to No-CP-Loss, confirming that incorporating CP at runtime provides meaningful improvements. **LogiCP achieves an average 49% MSE reduction compared to No-CP-Loss, demonstrating that CP-aware training provides greater benefits than post-hoc correction alone.** LogiCP-T further improves upon LogiCP by 30% through uncertainty-aware runtime validation, confirming that CP-aware training and runtime correction are complementary.

6 Discussion

In this section, we discuss several important aspects of LogiCP, including privacy considerations, partial participation, the choice of STL over other formalisms, dataset heterogeneity, and the rationale behind runtime correction priorities.

6.1 Privacy and Data Leakage in Clustering

We assume that the STL-based clustering process in LogiCP does not introduce data leakage, as no raw data is shared between clients at any point in the protocol. LogiCP adheres to standard practices in FL, communicating only model parameters and a length-T quantile vector used for CP calibration.

We clarify that transmitting the quantile vector does not compromise client privacy. The quantile values $\ell_\tau^{(i)}$ are aggregated statistics derived from prediction errors (nonconformity scores), not raw data points. These values

Table 10. LogiCP performance under varying participation rates on FHWA dataset with $\alpha = 0.1$ using the Transformer model. Results show MSE and CP coverage rate (v) at both client and cluster levels.

Level	Metric	frac = 0.3	frac = 0.5	frac = 0.7	frac = 0.9	frac = 1
LogiCP (Client)	MSE	0.0093 ± 0.0225	0.0057 ± 0.0145	0.0033 ± 0.0069	0.0029 ± 0.0036	0.0026 ± 0.0018
	v	97.20%	98.16%	98.71%	98.68%	98.54%
LogiCP-S (Cluster)	MSE	0.0090 ± 0.0042	0.0058 ± 0.0030	0.0041 ± 0.0019	0.0035 ± 0.0009	0.0041 ± 0.0033
	v	98.16%	98.96%	99.05%	99.10%	98.72%

capture the model's prediction uncertainty at each time step, which is a characteristic of model behavior rather than client data. **While the quantile vector has length T, its values represent error bounds rather than signal values.** Two clients with vastly different data distributions (e.g., high-traffic urban sensor vs. low-traffic rural sensor) could produce similar quantile vectors if their models have comparable prediction accuracy, and vice versa. Therefore, the temporal structure of the quantile vector reflects how uncertainty evolves over the prediction horizon—a general feature of sequential forecasting—rather than revealing the shape of a client's private trajectory.

Importantly, communication within each cluster involves no sharing of raw inputs, labels, or intermediate representations. This is aligned with recent FL studies where clustering is commonly employed to improve personalization while maintaining privacy (Ghosh et al. 2020; B. Liu et al. 2024; J. Lu et al. 2024). Furthermore, our framework adheres to privacy-preserving principles also emphasized in prior works (McMahan et al. 2017; Vahidian et al. 2023), which strictly prohibit raw data sharing across clients or with the central server. Hence, we believe the design of LogiCP sufficiently mitigates the risk of data leakage.

6.2 Partial Participation and Coverage Validity

In practical FL deployments, not all clients participate in every communication round. LogiCP preserves its effectiveness under partial participation through a round-synchronized design, where clustering and CP calibration are performed at the start of each round using only participating clients. Specifically, (1) **client-to-cluster assignments remain fixed within each communication round** during both CP calibration and model training, and (2) only participating clients are clustered, ensuring that **no client uses a CP region from a cluster it does not belong to**. This design preserves the exchangeability required for valid CP coverage.

Empirical results confirm that LogiCP maintains valid coverage guarantees under partial participation: as shown in Table 10, the empirical coverage consistently exceeds the target coverage of 90% ($\alpha = 0.1$) across all participation rates, ranging from 97.20% to 99.10%. Meanwhile, under varying participation rates from 0.3 to 0.9, LogiCP consistently outperforms baselines, achieving an average 31% reduction in MSE (see Table 5). These results demonstrate LogiCP's reliability and effectiveness under realistic partial participation scenarios.

6.3 STL Specifications

STL is particularly well-suited for reasoning about time-series data due to its ability to incorporate timing constraints directly into logical specifications. Unlike FOL, which lacks native constructs for specifying temporal intervals, STL enables the expression of properties such as threshold violations, sustained trends, and bounded temporal events in a compact and interpretable form. This expressiveness is especially valuable in FL scenarios where client datasets consist of sequential, time-indexed signals (e.g., traffic volumes).

While both STL and FOL are capable of representing temporal properties, we choose STL for its greater convenience and scalability in our context. For example, a temporal condition such as "eventually within the next 10 time steps, the signal x falls below 10" can be concisely written in STL as $\diamond_{[0,10]}(x < 10)$. In contrast,

FOL would require enumerating conditions over each time point, leading to a specification whose length scales linearly with the trace length L. Thus, STL enables more efficient and readable representations of time-dependent behaviors, making it a practical and natural choice for inferring temporal properties in FL.

6.4 Dataset Heterogeneity

Our experimental design assumes that the employed traffic flow dataset captures data heterogeneity, as it is derived from real-world measurements across multiple locations and time periods. This assumption is based on that traffic patterns are influenced by diverse spatial, temporal, and behavioral factors, such as geographic location, commuting trends, or weather conditions, which vary significantly across monitoring stations.

By utilizing this dataset, we simulate realistic heterogeneous environments without manually inducing variation. This provides a meaningful justification for evaluating LogiCP's ability to generalize and scale across diverse client distributions. While data with synthetic heterogeneity allows for controlled experiments, real-world data offers more practical insights into the challenges of FL in large-scale, distributed systems.

6.5 CP Boundary Priority in Runtime Correction

We clarify that the CP region ρ at runtime is a relative region centered at the prediction $\hat{y}$, so the prediction itself never violates the CP region. The correction mechanism operates as follows: (1) when the prediction violates the STL property region ϕ, but the CP region ρ and STL region ϕ overlap, the correction is applied to the STL property boundary; (2) when the prediction violates the STL property region ϕ, and the CP region ρ and STL region ϕ do not overlap, LogiCP prioritizes the CP boundary for correction. This is because the CP region provides a mathematical guarantee that the true value lies within the interval with probability at least $1 - \alpha$, whereas the STL-inferred property region is an empirical approximation derived through STL mining from parameterized templates. The conflict scenario (Case 2) is relatively rare in practice, as evidenced by our ablation study showing negligible performance differences between CP-priority and STL-priority correction strategies (Table 6). Given this empirical equivalence, LogiCP prioritizes the CP boundary when conflicts occur, as it offers correction based on a mathematically guaranteed coverage bound rather than an empirically derived constraint.

7 Related Work

There have been related literature aiming to enhance the performance of the FL framework in the context of traffic flow prediction while minimizing the exchanged data (Sepasgozar and Pierre 2022; C. Zhang, Dang, et al. 2021). The solution includes implementing a consortium blockchain-based FL framework to ensure secure learning without a centralized model, or conducting an FL-based gated recurrent unit neural network algorithm in forecasting task (Y. Liu et al. 2020; Qi et al. 2021). BNNs are a common approach to incorporate in FL to enhance the performance (P. Li et al. 2025; X. Zhang, Y. Li, et al. 2022). Zhang et al. (X. Zhang, Y. Li, et al. 2022) propose to address the FL overfitting problem via Bayesian variational inference by introducing weight uncertainty. An et al. (An et al. 2024) propose FedSTL, which introduces pattern-aware clustering and property-loss enforcement for personalized FL. However, the predefined STL templates may not perfectly describe all possible temporal behaviors, and such inflexibility can introduce additional uncertainty in dynamic settings. Moreover, Graph Neural Networks (GNN) are another common way to protect data privacy while retaining the topological information of the transportation network (Xia et al. 2022; C. Zhang, S. Zhang, et al. 2021). Yuan et al. (Yuan et al. 2022) propose a federated deep learning-based algorithm to capture long-term and short-term hidden spatial-temporal information and combine it with the semantics from the semantic capture network to increase traffic flow prediction efficiency. Besides the Quantile-of-Quantile (Humbert et al. 2023) method, there are other approaches to conduct CP to incorporate the uncertainty by utilizing importance weighting to address the label shift between agents (Plassier, Makni, et al. 2023). Plassier et al. (Plassier, Kotelevskii, et al. 2024) also

propose an approach to generate confidence sets for non-exchangeable data distributions. Moreover, other works apply CP in practical scenarios. Lu et al. (C. Lu et al. 2023) propose a weaker notion of partial exchangeability to deal with data heterogeneity across the clients in the post-processing step for the computer vision task of medical imaging. Zhu et al. (Zhu et al. 2024) utilize the statistical information derived from CP to calibrate a decision interval according to the loss of local device data to enhance the quality of the inference decision at the server.

8 Conclusion

In this paper, we present LogiCP, a novel framework that integrates STL-based formal inference with CP to enable personalized and reliable uncertainty quantification in FL. To address the fundamental challenge of data heterogeneity and its impact on model uncertainty, LogiCP infers client-specific temporal properties using STL and leverages them to form semantically coherent clusters. This logic-guided clustering reduces intra-cluster heterogeneity, enhancing the reliability of uncertainty estimation. To preserve privacy, LogiCP employs a decentralized CP implementation that avoids raw data sharing. In addition, LogiCP incorporates a runtime validation that calibrates predictions using STL-guided uncertainty regions, further improving accuracy and reliability in dynamic, real-world FL deployments. Comprehensive evaluations on three real-world datasets with diverse temporal characteristics demonstrate LogiCP's generalizability across varying data patterns. Across multiple model architectures and client scales, LogiCP consistently outperforms baseline methods in both prediction accuracy and provides more reliable uncertainty quantification. It achieves up to a 95% improvement in client-level MSE and maintains stable performance as the number of clients increases. These results underscore the effectiveness of integrating formal logic inference with UQ to manage heterogeneity and improve personalized performance in FL. Overall, LogiCP offers a scalable, generalizable, and reliable solution for accurate and personalized federated learning with rigorous uncertainty quantification, effectively addressing the core challenges of heterogeneity and uncertainty in large-scale, real-world applications while preserving data privacy.

Acknowledgments

This work was supported in part by the National Science Foundation under grants 2443803 and 2220401. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Z. An, T. T. Johnson, and M. Ma. 2024. "Formal logic enabled personalized federated learning through property inference." In: *Proceedings of the AAAI Conference on Artificial Intelligence* 10. Vol. 38, 10882–10890.

S. Banabilah, M. Aloqaily, E. Alsayed, N. Malik, and Y. Jararweh. 2022. "Federated learning review: Fundamentals, enabling technologies, and future applications." *Information processing & management*, 59, 6, 103061.

D. Caldarola, M. Mancini, F. Galasso, M. Ciccone, E. Rodolà, and B. Caputo. 2021. "Cluster-driven graph federated learning over multiple domains." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2749–2758.

A. Donzé and O. Maler. 2010. "Robust satisfaction of temporal logic over real-valued signals." In: *International conference on formal modeling and analysis of timed systems*. Springer, 92–106.

FHWA. 2016. *Highway Performance Monitoring System Field Manual*. https://www.fhwa.dot.gov/policyinformation/hpms/fieldmanual/hpms_field_manual_dec2016.pdf. Accessed: 2024-01-24. (2016).

A. Ghosh, J. Chung, D. Yin, and K. Ramchandran. 2020. "An efficient framework for clustered federated learning." *Advances in neural information processing systems*, 33, 19586–19597.

P. Humbert, B. Le Bars, A. Bellet, and S. Arlot. 2023. "One-shot federated conformal prediction." In: *International Conference on Machine Learning*. PMLR, 14153–14177.

S. Jha, A. Tiwari, S. A. Seshia, T. Sahai, and N. Shankar. 2017. "Telex: Passive stl learning using only positive examples." In: *International Conference on Runtime Verification*. Springer, 208–224.

J. Ji, J. Wang, C. Huang, J. Wu, B. Xu, Z. Wu, J. Zhang, and Y. Zheng. 2023. "Spatio-temporal self-supervised learning for traffic flow prediction." In: *Proceedings of the AAAI conference on artificial intelligence* 4. Vol. 37, 4356–4364.

S. Kaffash, A. T. Nguyen, and J. Zhu. 2021. "Big data algorithms and applications in intelligent transportation system: A review and bibliometric analysis." *International journal of production economics*, 231, 107868.

A. A. Kashyap, S. Raviraj, A. Devarakonda, S. R. Nayak K, S. KV, and S. J. Bhat. 2022. "Traffic flow prediction models–A review of deep learning techniques." *Cogent Engineering*, 9, 1, 2010510.

W. Kong, Y. Ju, S. Zhang, J. Wang, L. Huang, and H. Qu. 2025. "Graph enhanced spatial–temporal transformer for traffic flow forecasting." *Applied Soft Computing*, 170, 112698.

B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao. 2020. "DeepFed: Federated deep learning for intrusion detection in industrial cyber–physical systems." *IEEE Transactions on Industrial Informatics*, 17, 8, 5615–5624.

L. Li, Y. Fan, M. Tse, and K.-Y. Lin. 2020. "A review of applications in federated learning." *Computers & Industrial Engineering*, 149, 106854.

P. Li, Q. Hu, and X. Wang. 2025. "Federated learning meets Bayesian neural network: Robust and uncertainty-aware distributed variational inference." *Neural Networks*, 185, 107135.

B. Liu, Y. Ma, Z. Zhou, Y. Shi, S. Li, and Y. Tong. 2024. "Casa: Clustered federated learning with asynchronous clients." In: *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1851–1862.

Y. Liu, J. Q. James, J. Kang, D. Niyato, and S. Zhang. 2020. "Privacy-preserving traffic flow prediction: A federated learning approach." *IEEE Internet of Things Journal*, 7, 8, 7751–7763.

C. Lu, Y. Yu, S. P. Karimireddy, M. Jordan, and R. Raskar. 2023. "Federated conformal predictors for distributed uncertainty quantification." In: *International Conference on Machine Learning*. PMLR, 22942–22964.

J. Lu, Y. Chen, S. Cao, L. Chen, W. Wang, and Y. Xin. 2024. "LEAP: optimization hierarchical federated learning on non-IID data with coalition formation game." In: *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, 4660–4668.

Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang. 2020. "Federated learning for data privacy preservation in vehicular cyber-physical systems." *IEEE Network*, 34, 3, 50–56.

M. Ma, J. Gao, L. Feng, and J. Stankovic. 2020. "STLnet: Signal temporal logic enforced multivariate recurrent neural networks." *Advances in Neural Information Processing Systems*, 33, 14604–14614.

M. Ma, S. M. Preum, M. Y. Ahmed, W. Tärneberg, A. Hendawi, and J. A. Stankovic. 2019. "Data sets, modeling, and decision making in smart cities: A survey." *ACM Transactions on Cyber-Physical Systems*, 4, 2, 1–28.

O. Maler and D. Nickovic. 2004. "Monitoring temporal properties of continuous signals." In: *International symposium on formal techniques in real-time and fault-tolerant systems*. Springer, 152–166.

B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. 2017. "Communication-efficient learning of deep networks from decentralized data." In: *Artificial intelligence and statistics*. Pmlr, 1273–1282.

B. Medina-Salgado, E. Sánchez-DelaCruz, P. Pozos-Parra, and J. E. Sierra. 2022. "Urban traffic flow prediction techniques: A review." *Sustainable Computing: Informatics and Systems*, 35, 100739.

S. Pati et al.. 2024. "Privacy preservation for federated learning in health care." *Patterns*, 5, 7.

V. Plassier, N. Kotelevskii, et al.. 2024. "Efficient conformal prediction under data heterogeneity." In: *International Conference on Artificial Intelligence and Statistics*. PMLR, 4879–4887.

V. Plassier, M. Makni, A. Rubashevskii, E. Moulines, and M. Panov. 2023. "Conformal prediction for federated uncertainty quantification under label shift." In: *International Conference on Machine Learning*. PMLR, 27907–27947.

S. M. Preum, S. Munir, M. Ma, M. S. Yasar, D. J. Stone, R. Williams, H. Alemzadeh, and J. A. Stankovic. 2021. "A review of cognitive assistants for healthcare: Trends, prospects, and future directions." *ACM Computing Surveys (CSUR)*, 53, 6, 1–37.

Y. Qi, M. S. Hossain, J. Nie, and X. Li. 2021. "Privacy-preserving blockchain-based federated learning for traffic flow prediction." *Future Generation Computer Systems*, 117, 328–337.

S. S. Sepasgozar and S. Pierre. 2022. "Fed-NTP: A federated learning algorithm for network traffic prediction in VANET." *IEEE Access*, 10, 119607–119616.

G. Shafer and V. Vovk. 2008. "A tutorial on conformal prediction." *Journal of machine learning research*, 9, 3.

R. J. Tibshirani, R. Foygel Barber, E. Candes, and A. Ramdas. 2019. "Conformal prediction under covariate shift." *Advances in neural information processing systems*, 32.

A. Trindade. 2015. *ElectricityLoadDiagrams20112014*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C58C86. (2015).

University of Dayton. Nov. 2002. *Temperature Data from Around the World Attracts Web Visitors to University of Dayton Site*. News Releases, University of Dayton. News Releases. 9948. (Nov. 2002). https://ecommons.udayton.edu/news_rls/9948.

S. Vahidian, M. Morafah, W. Wang, V. Kungurtsev, C. Chen, M. Shah, and B. Lin. 2023. "Efficient distribution similarity identification in clustered federated learning via principal angles between client data subspaces." In: *Proceedings of the AAAI conference on artificial intelligence* 8. Vol. 37, 10043–10052.

M. Xia, D. Jin, and J. Chen. 2022. "Short-term traffic flow prediction based on graph convolutional networks and federated learning." *IEEE Transactions on Intelligent Transportation Systems*, 24, 1, 1191–1203.

R. Yan, L. Qu, Q. Wei, S.-C. Huang, L. Shen, D. L. Rubin, L. Xing, and Y. Zhou. 2023. "Label-efficient self-supervised federated learning for tackling data heterogeneity in medical imaging." *IEEE Transactions on Medical Imaging*, 42, 7, 1932–1943.

X. Yuan, J. Chen, J. Yang, N. Zhang, T. Yang, T. Han, and A. Taherkordi. 2022. "Fedstn: Graph representation driven federated learning for edge computing enabled urban traffic flow prediction." *IEEE Transactions on Intelligent Transportation Systems*, 24, 8, 8738–8748.

M. Zaffran, O. Féron, Y. Goude, J. Josse, and A. Dieuleveut. 2022. "Adaptive conformal predictions for time series." In: *International conference on machine learning*. PMLR, 25834–25866.

C. Zhang, S. Zhang, J. James, and S. Yu. 2021. "FASTGNN: A topological information protected federated learning approach for traffic speed forecasting." *IEEE Transactions on Industrial Informatics*, 17, 12, 8464–8474.

C. Zhang, S. Dang, B. Shihada, and M.-S. Alouini. 2021. "Dual attention-based federated learning for wireless traffic prediction." In: *IEEE INFOCOM 2021-IEEE conference on computer communications*. IEEE, 1–10.

X. Zhang, Y. Li, W. Li, K. Guo, and Y. Shao. 2022. "Personalized federated learning via variational bayesian inference." In: *International Conference on Machine Learning*. PMLR, 26293–26310.

X. Zhang, S. Wen, L. Yan, J. Feng, and Y. Xia. 2024. "A hybrid-convolution spatial–temporal recurrent network for traffic flow prediction." *The Computer Journal*, 67, 1, 236–252.

Y. Zhao, B. Hoxha, G. Fainekos, J. V. Deshmukh, and L. Lindemann. 2024. "Robust conformal prediction for STL runtime verification under distribution shift." In: *2024 ACM/IEEE 15th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 169–179.

M. Zhu, M. Zecchin, S. Park, C. Guo, C. Feng, and O. Simeone. 2024. "Federated inference with reliable uncertainty quantification over wireless channels via conformal prediction." *IEEE Transactions on Signal Processing*, 72, 1235–1250.

Received 21 December 2025; accepted 30 April 2026