Uncertainty-Aware Online Time Series Multi-Step Forecasting Framework in Cloud Systems

Jiadong Chen, Yang Luo, Xiuqi Huang, Fuxin Jiang, Yangguang Shi, Tieying Zhang, and Xiaofeng Gao, *Senior Member, IEEE*

***Abstract*—Accurate resource planning in large-scale systems relies on reliable predictions of future workloads, a task inherently challenged by their variability and dynamism. Previous prediction methods are either ineffective to deal with the changing dynamics of the series, or are highly black-boxed and unable to conduct effective theoretical analysis. To address these issues, we design an effective ensemble framework, Interval Prediction with Online Chasing (IPOC), tailored for multi-step interval forecasting in real-time systems. Theoretically, by formulating the task as a Dynamic Deterministic Markov Decision Process (Dd-MDP), an advanced theoretical framework is introduced to analyze problem solvability and derive conditions for the existence of feasible solutions. Incorporating the proposed Adaptive Copula Conformal Inference (ACCI) module and a well-designed Chasing Oracle, IPOC captures the changing dynamics and temporal dependencies to enable multi-step forecasting. We organically integrate advanced online learning theories with time series forecasting tasks to construct a forecasting framework that is both theoretically rigorous and practically effective. Theoretical analysis underpins IPOC's effectiveness, demonstrating sublinear regret and adherence to confidence interval specifications. The chasing regret of the Chasing Oracle is $O(L_c)$, and the overall regret of IPOC is $O(\sqrt{L_c T \log |\mathcal{F}|})$. Empirically, IPOC is validated through extensive experiments on five real-world datasets, including public datasets and different types of workload collected from Bytedance Cloud, with comparisons to 25 baselines and 4 forecasting horizons (1/5/10/30). Specifically, IPOC achieves an average reduction of over 20% in RMSE/MAE/SMAPE/ρ-risk compared to baselines across five datasets. Besides, we apply our model to a case study on predictive auto-scaling tasks in actual large-scale cloud systems to validate its utility.**

***Index Terms*—Online learning, sub-linear regret, time series, interval forecasting, ensemble learning, conformal inference.**

Received 28 March 2025; revised 25 February 2026; accepted 3 March 2026. Date of publication 16 March 2026; date of current version 9 April 2026. This work was supported by the National Key R&D Program of China under Grant 2024YFF0617700, in part by the National Natural Science Foundation of China under Grant U23A20309, Grant 62272302, Grant 62302273, in part by the Science Fund Program of Shandong Province for Distinguished Oversea Young Scholars under Grant 2023HWYQ-006, and in part by the Bytedance Research Project under Grant CT20241217115379. Recommended for acceptance by D.-N. Yang. *(Corresponding author: Xiaofeng Gao.)*

Jiadong Chen, Yang Luo, and Xiaofeng Gao are with the Shanghai Key Laboratory of Scalable Computing and Systems, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: chenjiadong998@sjtu.edu.cn; floatingdream@sjtu.edu.cn; gaoxiaofeng@sjtu.edu.cn).

Xiuqi Huang is with the State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310027, China (e-mail: huangxiuqi@zju.edu.cn).

Fuxin Jiang and Tieying Zhang are with Bytedance Incorporated, Beijing 100098, China (e-mail: jiangfuxin@bytedance.com; tieying.zhang@bytedance.com).

Yangguang Shi is with the School of Computer Science and Technology, Shandong University, Jinan 250100, China (e-mail: shiyangguang@sdu.edu.cn).

Digital Object Identifier 10.1109/TKDE.2026.3674583

I. Introduction

SERVERLESS computing is a cloud computing model where the cloud provider dynamically manages the allocation of machine resources and allows clients to build and run services without thinking about servers [1], [2], as shown in Fig. 1, whose primary challenge is to elastically manage resources for such applications facing the difficulty of dealing with cold starts, low latency, and scaling efficiency. Recent research shows that with effective auto-scaling, service quality can improve by 20% while resource waste decreases by 15% [3]. Workload forecasting is essential to address these challenges and achieve predictive auto-scaling. Since scaling operations involve physical resource scheduling and take time to complete, prediction needs to be both accurate and sufficiently long-term, which ensures enough time for subsequent scaling actions. A sample of workload data from a real-world large-scale system is shown in Fig. 2. We summarize two common workload patterns and two typical concept drifts based on studies [4] and industrial practice. *Concept drift* is defined as the changes in the distribution of streams over time, in which the distribution of workloads changes during the drifts. Fig. 2(a) and (b) show two common workload patterns, where the growth pattern depicts a steady increase in workload (e.g., more and more stored data) and the periodic pattern depicts a cyclical growth and reduction in workload (e.g. periodic query requests). Fig. 2(c) and (d) show two typical concept drifts. The sudden drift indicates a drastic change in workload (e.g., a new version release), while the recurring drift indicates a change in workload followed by a return to the original distribution for a period of time (e.g., the impact of holidays).

Due to the complexity of online load, accurate and effective workload forecasting is particularly difficult. The previous forecasting methods such as statistical [5], machine learning [6], and ensemble methods [4], [7] used in cloud or database systems only focus on point forecasting. Meanwhile, the current probabilistic forecasting models [8], [9], [10] do not provide theoretical guarantees, which makes their reliability questioned and contribution limited to real-world large-scale systems. Furthermore, deep learning models [8], [9], [11], [12], [13] that consume considerable time and computing resources to train are also difficult to adapt to the changing dynamics of the online environment. Based on this discussion, we outline the essential requirements and challenges for the workload forecasting task in proactive auto-scaling systems.

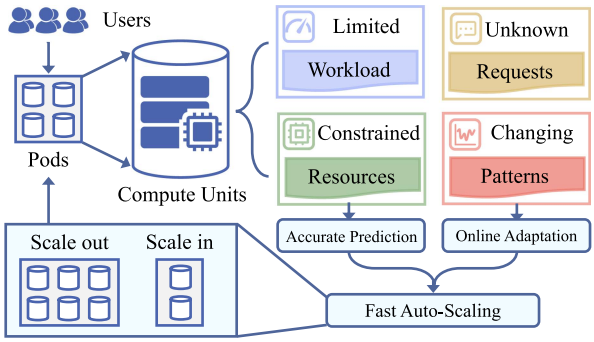

Fig. 1. With the proposed forecasting framework **IPOC**, serverless systems can achieve rapid online adaptation and handle cold starts, thereby enabling predictive auto-scaling and offering improved quality of service.

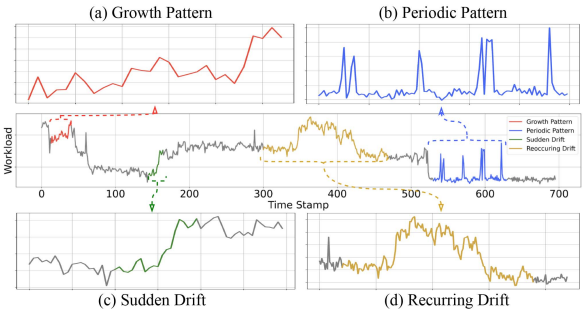

Fig. 2. Workload Example. A piece of workload data from a real-world large-scale system contains two common workload patterns and two typical concept drifts: growth pattern, periodic pattern, sudden drift, and recurring drift.

R1. Continuously adapt to the changing dynamics: Cloud workloads exhibit online variability, driven by fluctuating user request patterns and continuous platform updates. Their non-stationary behavior demands forecasting models with real-time adaptation capabilities to maintain predictive accuracy.

R2. Accurately quantify the uncertainty of the series: For auto-scaling systems, it is essential to accurately quantify the uncertainty associated with future workloads—such as by determining lower and upper bounds—in order to enhance resource utilization without compromising service quality.

R3. Theoretically analyze the robustness of the forecasts: For cloud systems, robustness constitutes one of the most critical requirements; thus, the forecasting algorithm must provide rigorous guarantees and analytical support for predictions.

As far as we know, no existing work has fully met the above requirements. In this paper, we address the challenges of online multi-step interval forecasting, which aims to predict intervals that contain future values with a specified probability. We propose an ensemble strategy to integrate various base forecasting models, adapting to diverse series patterns. Our system, **IPOC**, is a lightweight ensemble framework that leverages a dynamic deterministic Markov decision process (Dd-MDP) for online learning, offering theoretical guarantees. It converts point predictions to confidence intervals using adaptive conformal inference and selects a target model through a model selector. The chasing oracle within **IPOC** enables the ensemble to track the target model, ensuring accurate confidence intervals.

We have theoretically established that **IPOC**'s chasing regret is $O(L_c)$, validating its effectiveness. **IPOC** guarantees not only the validity of prediction intervals but also the sub-linear regret performance of the model ensemble, even in the worst-case scenario. Extensive experiments on five real-world datasets and comparisons with 25 baselines, along with a case study on capacity recommendation in HDFS, further validate **IPOC**'s contributions to practical tasks. The main contributions of this paper can be summarized as follows:

This manuscript is a journal extension to our previous conference paper [14]. This journal version involves several improvements to enhance the previous model from the following aspects. First, we have extended the scope of the forecasting problem from single-step to multi-step forecasting as shown in Section III. In Section IV, we elaborate on the theoretical underpinnings of our approach and designed the **Adaptive Copula Conformal Inference (ACCI)**, which captures temporal dependencies to generate confidence intervals for multi-step forecasts. In Section V, we expand the design of our ensemble module, introducing an online ensemble strategy based on the Exponentiated Gradient Descent (EGD) algorithm. Furthermore, in Section VI, we provide theoretical proof of the reliability of the IPOC framework, including the ACCI method. Additionally, we collect new workload data from Bytedance Cloud's public IaaS platform for experiments and perform more detailed analyses in Section VII.

1) We formally define the multi-step online time series interval forecasting (MOTSF-Int) and model it as a dynamic deterministic Markov decision process instance.
2) We design the Adaptive Copula Conformal Inference (ACCI) to quantify the uncertainty of multi-step forecasts. We construct an online ensemble framework, **IPOC**, to solve the MOTSF-Int task. To our knowledge, we are the first to introduce the online learning theory into the time series analysis task.
3) We theoretically analyze the effectiveness of each part of **IPOC**, such as the effectiveness of ACCI and the regret bound of IPOC. The regret of **IPOC** is $O(\sqrt{L_c T \log |\mathcal{F}|}) = O(\sqrt{T})$ with fixed parameters $L_c, |\mathcal{F}|$, which is sublinear asymptotically.
4) We conduct extensive experiments on five real-world datasets and a case study of predictive auto-scaling in Kubernetes system to verify the effectiveness of **IPOC**.
5) In real-world Kubernetes horizontal pod proactive auto-scaling tests, compared with the built-in Naïve auto-scaling module, **IPOC** demonstrates significant improvements in resource utilization and latency reduction.

II. Related Works

Time Series for Systems: Long-term time series forecasting is vital for strategic planning and decision-making, enabling organizations to anticipate future trends and patterns. Traditional statistical methods [15], [16], [17] stationary data using autoregressive and moving average components. Machine learning models such as Random Forest [18], [19], [20] leverage non-linear fitting and ensemble methods for robust predictions.

Deep learning models [11], [12], [13], [21], [22], [23], [24], [25] can capture long-term dependencies and complex patterns. In large-scale systems, workloads are usually modeled and processed in the form of time series and workload forecasting has begun to be applied to system load balancing, parameter optimization, capacity scaling, etc [26]. LoadDynamics [6] employs a combination of LSTM and Bayesian optimization for handling the dynamic fluctuations. WGAN-gp Transformer [2] adopts a Transformer as a generator and a multi-layer perception as a critic to predict cloud workload. QueryBot [4] combines linear regression and RNNs for various database workload patterns. DBAugur [7] uses adversarial neural networks to predict the trends of database workloads and shows the superiority of index selection and data region migration tasks. AHPA [1] and PASS [3] are designed for auto-scaling based on long-term time series forecasting techniques.

Conformal Inference: Conformal inference is a powerful tool for quantifying the uncertainty around predictions made by black-box models. ACI [27], SAOCP [28], and NexCP [29] effectively handle distribution shifts via online parameter updates or weighted quantiles, but they treat forecasting steps independently, neglecting the joint dependencies across the multi-step horizon. CopulaCPTS [30] addresses this by modeling inter-step dependencies via copulas, yet its reliance on static calibration renders it brittle to real-time concept drifts. Similarly, AcMCP [31] accounts for serial dependence in online settings by calibrating residual sequences, whereas our proposed **IPOC** employs empirical copulas to model the high-dimensional joint distribution and integrates a regret-minimized ensemble strategy for robust online update.

III. Preliminary

A. Online Time Series Forecasting

In large-scale systems, workloads (e.g. query per second, CPU usage, etc.) appear as potentially unbounded streams of continuous data in real (or near-real) time, reflecting system statuses, as described in Definition 1.

Definition 1 (Data Stream): Data streams are the continuous flow of data elements ordered in a sequence, which is processed in real-time or near-real-time to gather valuable insights. We denote by x_t the signal recorded at time stamp t, and a data stream can be denoted as $\mathcal{X} = \{x_1, x_2, \cdots x_t, \cdots\}$.

Data streaming is a modern approach to processing and analyzing data in real-time, as opposed to batch processing methods. Key features of data streams include their **continuous** flow, **infinite length**, **unbounded nature**, **high velocity**, and potentially **high variability**. Unlike traditional batch data processing, where data is collected and processed in batches, data streams are continuously collecting data, making it possible to process data as soon as they are created. This provides large-scale systems with the ability to monitor and succeed in day-to-day operation. Data streams are widely used in large scale systems like Apache Flink, Apache Spark Streaming, Apache Kafka, and Google Dataflow.

With the operation of the large-scale system, new signals are continuously recorded, and the stream is continuously longer. Due to the need for real-time and efficiency, we cannot use unlimited memory to record all histories and then predict the future. Therefore we are considering an *multi-step online time series forecasting* problem over data stream $\mathcal{X}$, abbreviated as MOTSF problem. Next, let us discuss the influence of limited memory and parameter updating for MOTSF.

Firstly, there is no separation of training and evaluation in an online setting. Instead, learning occurs over a sequence of time-steps [32]. Hence, At each time stamp t, we can define a length parameter L, a prediction horizon H, and use data sequence $X_t = x_{t-L:t-1}$ to produce a multi-step point prediction $\hat{Y}_t = f(X_t; \theta_t^f) = \hat{x}^f_{t:t+H-1}$ for X_t with prediction model f and associated parameter set θ_t^f. X_t is a fixed-length observed series history, focusing on the latest and limited observation in online scenarios. We use loss function $\ell(\hat{Y}_t, Y_t) = \ell(\hat{x}^f_{t:t+H-1}, x_{t:t+H-1})$ to measure the gap between $\hat{x}^f_{t:t+H-1}$ and $x_{t:t+H-1}$. Specifically, ℓ maps the prediction and the truth into a real number in [0,1], and the more accurate the prediction is, the smaller the loss is. Depending on the requirements, there are many choices for ℓ.

Secondly, since a data stream is recorded continuously, the learning parameter θ should be updated at each time t, which derives the formal definition of MOTSF, denoted as Definition 2.

Definition 2 (Multi-Step Online Time Series Forecasting, MOTSF): At time stamp t, given an L-length history stream X_t and the prediction horizon H, the MOTSF task requires a point prediction $f(X_t; \theta_t^f) = \hat{X}_t^f$. After making the prediction, the prediction model f then receives the true answer Y_t and generates a related loss $\ell(\hat{Y}_t, Y_t)$. Whenever the loss is nonzero, f updates the parameter set from θ_t^f to θ_{t+1}^f by applying some update strategy on the training example pair (X_t, Y_t). By running the online learning T time-steps, the goal is to minimize the cumulative loss, i.e., $\sum_{t=1}^{T} \ell(\hat{Y}_t, Y_t)$.

The performance of an online model is measured by comparing its cumulative loss with the minimum loss of the offline algorithm, and its difference is denoted as *regret* [32]:

$$regret = \sum_{t=1}^{T} \ell(\hat{Y}_t, Y_t) - \inf_{\theta} \sum_{t=1}^{T} \ell(f(X_t; \theta), Y_t). \quad (1)$$

The second term in (1) is the loss suffered by the model with the best parameter setting θ^*, which can only be known in hindsight after seeing the full data stream. An online model's regret is *sub-linear* as a function of T, if and only if $regret = o(T)$, which implies $\lim_{T\to\infty} \frac{regret}{T} \to 0$ and thus on average the model performs almost as well as setting the best parameter in hindsight. We always expect an online learner achieving a sub-linear regret.

B. Online Time Series Interval Prediction

As discussed in Section I, for real management in large-scale systems, such as load balancing or resource scaling, the results of the point prediction are often insufficient. Predicting the potential range of a true value Y_t under a given miscoverage rate α is more meaningful [8], [9], and such range is named as *confidence interval*, whose definition is denoted as Definition 3.

TABLE I
SYMBOLS AND NOTATIONS

Symbols	Definitions
$\mathcal{X}$	data stream $\mathcal{X} = \{x_1, x_2, \cdots x_t, \cdots\}$
X_t	L-length history stream at time t, $X_t = x_{t-L:t-1}$
Y_t	H-length target series at time t, $Y_t = x_{t:t+H-1}$
$\hat{Y}_t$	point prediction from f at time stamp t, $\hat{Y}_t = \hat{x}_{t:t+H-1}$
C_t	prediction interval at time stamp t, $C_t = c_{t:t+H-1}$
α	miscoverage rate $\alpha \in [0, 1]$
ℓ	loss function
f, g	point/interval prediction model
$\mathcal{F}$	model pool with $\lvert\mathcal{F}\rvert$ base models, $\mathcal{F} = \{f^1, \cdots, f^{\lvert\mathcal{F}\rvert}\}$
$\overline{f}_t$	model selected at time stamp t, $\overline{f}_t \in \mathcal{F}$
$\overline{Y}_t, \overline{C}_t$	ensemble result
α_t	adaptive miscoverage rate α_t used in ACCI
$\mathcal{L}$	loss set of a given model
Q	fitted quantile used in ACI
$\mathcal{O}$	chasing oracle
$\mathcal{A}$	target model selector

Definition 3 (Confidence Interval): Given a miscoverage rate $\alpha \in [0, 1]$, the confidence interval for a set of variables $[x_1, \ldots, x_n]$ is an set of intervals $C = [c_1, \ldots, c_n]$, where $c_i = [\underline{c_i}, \overline{c_i}] \forall i \in [1, n]$ satisfying the probability inequality:

$$\mathrm{P}(x_t \in c_t) \geq 1 - \alpha, \quad \forall x_t \in \mathcal{X}. \tag{2}$$

Now we can extend the MOTSF problem into a new version, named as *multi-step online time series interval prediction*, abbreviated as MOTSF-Int problem. At time stamp t, given α, the model g can use X_t to produce confidence intervals $g(X_t, \alpha; \theta_t^g) = C_t^g$, where $C_t^g = [c_t^g, \ldots, c_{t+H-1}^g]$. According to the discussions in [8], [9], [10], there is no "truth interval" for MOTSF-Int problem, while we can still use a designed loss function $\ell(g(X_t, \alpha; \theta_t^g), Y_t) = \ell(C_t^g, Y_t)$ to measure the prediction quality. Definition 4 summarizes the above discussions.

Definition 4 (Multi-Step Online Time Series Interval Forecasting, MOTSF-Int): At each time stamp t, given the miscoverage rate α, the task of a model g is to produce H confidence intervals $g(X_t, \alpha; \theta_t^g) = C_t^g$, guaranteeing (2) and minimize the cumulative losses, i.e.,

$$\min_{\theta_t^g} \sum_{t=1}^{T} \ell(g(X_t, \alpha; \theta_t^g), x_{t:t+H-1})$$
$$s.t. \ \mathrm{P}\left(\forall\, k \in \{0, \ldots, H-1\}, x_{t+k} \in c_{t+k}^g\right) \geq 1 - \alpha. \tag{3}$$

Majority of previous works discussing interval prediction, or sometimes referred to as *probabilistic prediction* [8], [9], [10], only consider the minimization objective of (3), without the validation of (2). However, in reality, need for such validity is essential. To simplify the description, in the following sections we use $\hat{Y}_t^f$ to represent $f(X_t; \theta_t^f)$, and use C_t^g to represent $g(X_t, \alpha; \theta_t^g)$.

Table I summarizes symbols mainly used in this paper, which are defined by their appearances.

IV. Formulation and Modeling

In this section, we address the decomposition of the MOTSF problem, offering a formal description and model. Initially, we introduce a tool to convert point predictions into interval predictions, acknowledging that most existing research is centered on point prediction. Subsequently, we extend single-step prediction to multi-step prediction by modeling the dependencies within multi-step time series. Additionally, we incorporate the concept of model ensemble to bolster the robustness. Lastly, we frame the problem within the context of a *dynamic deterministic Markov decision process (Dd-MDP)* to account for its dynamic and stateful nature.

A. From Points to Intervals

Adaptive Conformal Inference (ACI) [27] provides a new perspective for generating confidence intervals with a theoretical guarantee. In this paper, we adapt ACI to generate confidence interval c_t based on the output $\hat{x}_t$ from a point prediction model f.

At time stamp t, we define a *loss set* for a point prediction model f, which collects the losses from its previous L_c training time-steps, denoted as $\mathcal{L}_t = \{\ell_{t-L_c}, \ldots, \ell_{t-1}\}$, where L_c is the size of loss set, satisfying $L_c \leq L$, since recording all history losses is also unrealistic. To deal with the occurred concept drifts for a time series, as illustrated in Section I, ACI converts the static α in Definition 3 into an *adaptive miscoverage rate* α_t, which is dynamically changed overtime w.r.t. different models. Following the *quantile*'s definition from Statistics, ACI calculates the *fitted* α_t*-th quantile* of $\mathcal{L}_t$, denoted as $Q(\mathcal{L}_t; \alpha_t)$, in (4):

$$Q(\mathcal{L}_t; \alpha_t) \triangleq \inf_b \left\{ b \;\middle|\; \left(\frac{1}{L_c} \sum\nolimits_{\ell \in \mathcal{L}_t} sign(\ell \leq b) \right) \geq 1 - \alpha_t \right\}, \tag{4}$$

where the $sign(\cdot)$ function in (4) returns 1 if the inner predicate is true, and 0 vice versa.

ACI further generates the confidence interval c_t from the point prediction model f, as described in (5).

$$c_t \triangleq \left[\hat{x}_t - Q(\mathcal{L}_t; \alpha_t), \hat{x}_t + Q(\mathcal{L}_t; \alpha_t)\right]. \tag{5}$$

When the true value x_t is revealed, the adaptive miscoverage rate α_t will be updated in an online manner, as shown in (6).

$$\alpha_{t+1} = \alpha_t + \gamma \left(\alpha - sign(x_t \notin \mathbf{c}_t)\right), \tag{6}$$

with a pre-defined parameter $\gamma \geq 0$. If c_t does not cover the true value x_t, α_{t+1} will be smaller than α_t, generating a larger $Q(\mathcal{L}_{t+1}; \alpha_{t+1})$, which means that at time stamp $t+1$ the confidence interval c_{t+1} will expand. The validation of (2) is thus guaranteed theoretically.

B. From Single-Step to Multi-Step

To address the limitations of ACI in capturing the dependencies in multi-step predictions, we draw inspiration from CopulaCPTS [30] and design the Adaptive Copula Conformal Inference (ACCI) method. While CopulaCPTS uses copulas to

Algorithm 1: Adaptive Copula Conformal Inference.

Input: Loss set $\mathcal{L}_t$, history stream X_t, miscoverage rate α, point prediction model f
Output: Confidence interval sequence $\mathcal{C}_t$.
1: Initialize the adaptive miscoverage rate $\alpha_0 \leftarrow \alpha$.
2: Randomly split the Loss set $\mathcal{L}_t$ into $\mathcal{L}_{cal}$ and $\mathcal{L}_{cop}$;
3: Calculate $\hat{F}_1, \ldots, \hat{F}_H$ according to (7);
4: Calculate $C_{\text{empirical}}(\cdot)$ according to (8);
5: Calculate $\mathbf{u}^*$ according to (9) with α_t;
6: $s_j^* = \hat{F}_j^{-1}(\mathbf{u}_j^*)$ for $j = 1, \ldots, H$;
7: $\hat{y} \leftarrow f(X_t)$;
8: $C_t \leftarrow \{y : \|y - \hat{y}_j\| \le s_j^*\}$ for $j = 1, \ldots, H$;
9: Calculate loss ℓ, Update loss set $\mathcal{L}_{t+1}$;
10: Update α_{t+1} according to (6);

analyze the joint distribution of multiple variables, it is constrained by a fixed confidence level, limiting its adaptability in dynamic online scenarios. Our ACCI method overcomes this by incorporating an adaptive algorithm for online dynamic adjustments, enhancing its applicability in real-time systems.

For the multi-step MOTSF-Int, the loss set $\mathcal{L}_t = \{\ell\}_{t-L_c}^{t-1}$, where $\ell_t = \{\ell_t^1, \ldots, \ell_t^H\}$ is the multi-step prediction loss. At each time t, we randomly split the loss set $\mathcal{L}_t$ into two sets, $\mathcal{L}_{cal}$ and $\mathcal{L}_{cop}$. $\mathcal{L}_{cal}$ is used to calculate the cumulative distribution function as follows:

$$\hat{F}_j(s) := \frac{1}{|\mathcal{L}_{cal}| + 1} \left(\tau + \sum_{\ell \in \mathcal{L}_{cal}} sign(l^j < s) \right) \tag{7}$$

where $\tau \sim (0, 1)$, for $j \in \{1, \ldots, k\}$. Next, for every data point in $\mathcal{L}_{cop}$, we evaluate the cumulative probability of the loss metric with the estimated conformal predictive distributions: $\mathcal{U} = \{\mathbf{u}^i\}_{i \in \mathcal{L}_{\text{cop}}}$, $\mathbf{u}^i = (u_1^i, \ldots, u_H^i) = (\hat{F}_1(s_1^i), \ldots, \hat{F}_k(s_H^i))$.

We adopt the empirical copula for modeling in this work. The empirical copula is a non-parametric method of estimating marginals directly from observation, and hence does not introduce any bias. For the joint distribution of H time steps, we construct $Q_{\text{empirical}} : [0, 1]^H \to [0, 1]$ as (8).

$$Q_{\text{empirical}}(\mathcal{L}_{cop}; \mathbf{u}) = \frac{1}{|\mathcal{L}_{\text{cop}}| + 1} \sum_{i \in \mathcal{L}_{\text{cop}} \cup \{\infty\}} \prod_{j=1}^{H} sign(\mathbf{u}_j^i \ge \mathbf{u}_j). \tag{8}$$

Here boldface ∞ is a k-dimensional vector with each $\infty_j = \infty$ for $j = 1, \ldots, H$. To fulfill the full-horizon validity condition, we only need to find appropriate $\mathbf{u}^*$ such that $C_{\text{empirical}}(\mathbf{u}^*) \ge 1 - \alpha$. Lastly, We obtain $(s_1^*, \ldots, s_H^*)$ by $\hat{F}_j^{-1}(\mathbf{u}_j^*)$ and construct the confidence intervals.

$$\arg\min_{\mathbf{u}^*} \sum_{j=1}^{H} \mathbf{u}_j^* \quad \text{s.t. } Q_{\text{empirical}}(\mathbf{u}^*) \le \alpha_t. \tag{9}$$

The adaptive miscoverage rate α_t will be updated in the same manner as shown in (6). Algorithm 1 summarizes the procedure of the proposed Adaptive Copula Conformal Inference (ACCI) algorithm.

C. From Single to Ensemble

Due to the complex and changeable dynamic changes, it is difficult for a single model to perform well for various time series. Therefore, adopting an ensemble strategy to combine different prediction models is a reasonable choice. We adopt the selection-based strategy because of its simplicity and convenience to the modeling and theoretical analysis. Specifically, let $\mathcal{F}$ be a model pool with $|\mathcal{F}|$ base point prediction models $\{f^1, f^2, \ldots, f^{|\mathcal{F}|}\}$. At each time stamp t, Given α, the ensemble model utilizes a model selector $\mathcal{A}$ to choose a model $\overline{f}_t$ from $\mathcal{F}$. Based on $\overline{f}_t(X_t)$ and the loss set $\mathcal{L}_t$ of $\overline{f}_t$, the corresponding confidence intervals $C_t^{\overline{f}}$ is computed by Eqn (4)-(6). For clarity, we use $\overline{f}$ to denote the ensemble model itself and $\overline{f}_t$ to denote the specific model selection at each time step t. The goal of ensemble model is to minimize the cumulative loss, i.e., $\sum_{t=1}^{T} \ell(C_t^{\overline{f}}, x_{t:t+h})$.

The statefulness of model selection for MOTSF-Int: According to (5), the calculation of $C_t^{\overline{f}}$ does not only rely on the point prediction $f(X_t; \theta_t^f)$, but also related to the loss set $\mathcal{L}_t$ of $\overline{f}$. This means that the confidence intervals $C_t^{\overline{f}}$ generated by the ensemble model and C_t^f generated by a base model f may be different, even if f is the chosen model by $\mathcal{A}$ and their point prediction results are the same, i.e., $\overline{f}_t = f$ and $\overline{f}(X_t) = f(X_t; \theta_t^f)$. That means the ensemble learning on MOTSF-Int problem is a *stateful* online learning problem.

To address the statefulness of ensemble learning on MOTSF-Int, we model this problem from a new perspective. A *dynamic deterministic Markov decision process (Dd-MDP)* [33] is defined over a set of states and a set of actions. Each state at time t is associated with a subset of actions called the feasible actions at t. A *state transition function* g maps each state and action to a new state. The Dd-MDP also includes a loss function ℓ that maps each state-action pair to a real value in [0,1]. We show ensemble learning for MOTSF-Int task can be modeled as a Dd-MDP.

State set: We first consider the state of model f at time t to be the limited history series X_t, or the previous prediction sequence $\{\hat{Y}\}_{t-L}^{t-1}$ as used in EA-DRL [34]. However, since we are modeling the state of a model running on a time-evolving stream, the state should reflect both the performance of f and the dynamics of the time-series until time t. Therefore, we identify the state with the past L prediction losses of the model, denoted as $\mathcal{S}_t = \{\ell(C^f, Y)\}_{t-L}^{t-1}$, since it reflects the result of the model selection $\overline{f}_t$, and also, captures the temporal information of $\mathcal{X}$.

Action set and loss function: We identify the action set with the base model set $\mathcal{F}$. At each time t, ensemble model selects a model $\overline{f}_t$ from $\mathcal{F}$ and produces corresponding confidence interval $C_t^{\overline{f}}$. Specifically, the loss function ℓ is the quantile loss:

$$\ell(c_t, x_t) = \begin{cases} \alpha(\overline{c_t} - x_t), & x_t \le \overline{c_t} \\ (1 - \alpha)(x_t - \overline{c_t}), & x_t > \overline{c_t}, \end{cases}$$

and the cumulative loss for multi-step prediction is $\ell(C_t, x_{t:t+h}) = 1/h \sum_{k=0}^{H-1} \ell(\mathbf{c}_{t+j}, x_{t+j})$.

State transition function: The state transition function is deterministic as selecting a model only leads to one possible next state, i.e., $g(\{\ell(C^f, Y)\}_{t-L}^{t-1}, \overline{f}_t) = \{\ell(C^f, Y)\}_{t-L+1}^{t}$.

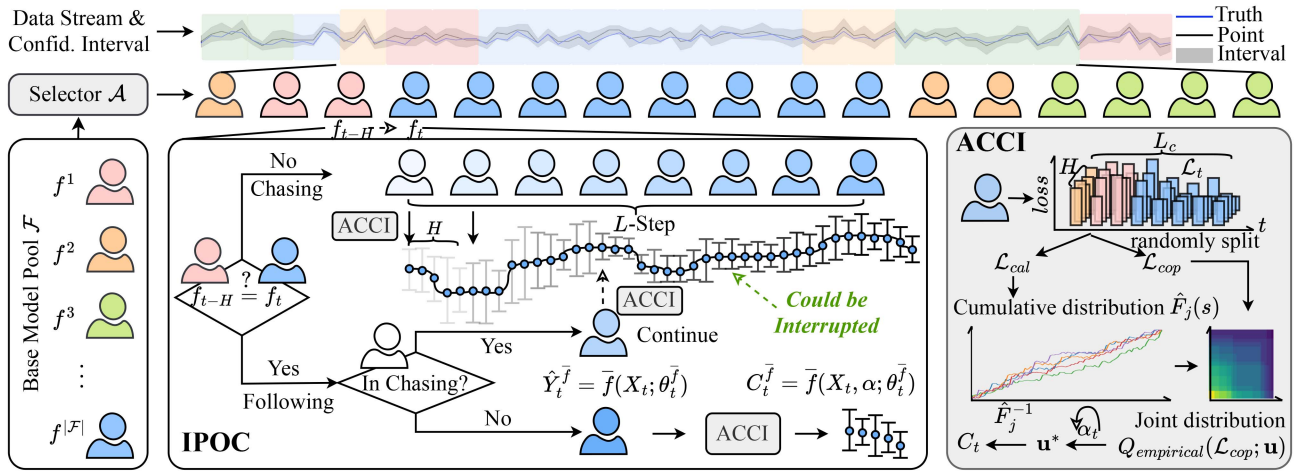

Fig. 3. **IPOC** Framework. Given a miscoverage rate, our goal is to produce the valid confidence interval. Given a base model pool, **IPOC** utilizes a model selector to choose a target model $\bar{f}_t$ and runs a chasing oracle to trace it and produce a corresponding confidence interval $\mathbf{c}_t^{\bar{f}}$ via adaptive conformal inference.

For a stateful online learning problem, we can define its *Chasability condition*, as described in Definition 5, and determine whether there exists an effective algorithm to solve it.

Definition 5 (Chasability condition for MOTSF-Int): A MOTSF-Int instance is called σ-chasable for some $\sigma > 0$ if it admits an ongoing chasing oracle $\mathcal{O}$ that works as follows for any given target model $f_{target} \in \mathcal{F}$. The chasing oracle is invoked at t_{init} and provided with an initial state $s_{init} \in \mathcal{S}$; this invocation is halted at $t_{final} \geq t_{init}$. At each time $t_{init} \leq t \leq t_{final}$, the chasing oracle generates a action $\hat{a}_t$, i.e., choose a base model $\hat{f}_t$. The main guarantee of $\mathcal{O}$ is that its chasing regret (CR) satisfies

$$CR \triangleq \sum_{t=t_{init}}^{t_{final}} \ell(C_t(\hat{f}_t), x_t) - \sum_{t_{init}}^{t_{final}} \ell(C_t(\hat{f}_{target}), x_t) \leq \sigma$$

After modeling the MOTSF-Int task as a Dd-MDP, we can draw support from stateful online learning algorithms [33] to solve it. Lemmas 1 and 2 tells us that for a Dd-MDP instance corresponding to ensemble learning on the MOTSF-Int problem, as long as we can conceive an effective chasing oracle, we can design an online algorithm to solve it.

Lemma 1 (Existence of OLSC Algorithm [33]): For any T-round Dd-MDP instance that admits an effective chasing oracle and any pool $\mathcal{F}$ of base models, there is online learning with switch cost algorithms with respect to the in-hindsight best model in $f \in \mathcal{F}$ is sublinear (and optimal) in T.

Lemma 2 (Regret Bound [35]): Online Learning with switching cost with upper bound Δ on the expected switching cost admits an online algorithm with regret $O(\sqrt{\Delta \cdot T \log |\mathcal{F}|})$. E.g., the Following The Perturbed Leader algorithm $\mathcal{A}$.

Therefore, the focus of the later part is to design a chasing oracle with relatively lower chasing regret, and on this basis, design an online learning algorithm to solve MOTSF-Int problem by model selection. Specifically, at each time stamp t, given a pool of base models $\mathcal{F}$, the ensemble model is required to select a target model that may perform best in the next step, track the target model and generate each confidence interval. The chasing process is to simulate the target model from any initial state, and there will be some ultra cost incurred every time the ensemble model switches from one target model from to another, which is called *switching cost* [35].

V. Framework Design

As discussed in Section IV-C, the challenge mainly comes from two aspects: the first is how to implement an effective chasing process, and the second is how to select the best-performing base model at each step. To meet the first challenge, in Section V-A, we design an effective algorithm, called chasing oracle, denoted as $\mathcal{O}$. It is surprisingly simple and intuitive but has a good theoretical guarantee. For the second challenge, we try different online learning algorithms. We choose the Following The Perturbed Leader (FTPL) [35] algorithm for **IPOC**, as shown in Fig. 3, whose regret is *sub-linear* even in the worst case. In Section V-B, we show the algorithm design of IPOC and prove its effectiveness theoretically.

A. Conception of Chasing Oracle

Inspired by [33], we design a chasing oracle for ensemble-based MOTSF-Int task with theoretical guarantee. The running goal of chasing oracle $\mathcal{O}$ is to ensure the ensemble model can reach the same state as the target model after chasing, in which case the ensemble model can produce the same confidence interval as the target model. The specific implementation of $\mathcal{O}$ is shown in Algorithm 2.

Given a target model f with the chasing procedure starting from t_{init}, the chasing process continues L time-steps. For each time t in $[t_{init}, t_{init+L-1}]$, the ensemble model continues to select $\bar{f} = f$ and set $\hat{Y}_t^{\bar{f}} = \hat{Y}_t^f$ (Line 2–3) and force ensemble model to set $\alpha_t^{\bar{f}}$ equal to α_t^f (Line 4). After that, the ensemble model produces confidence intervals (Line 5–6). Because of the different states of the two models, even if the same point prediction results are used by ensemble model and target model,

Algorithm 2: Chasing Oracle $\mathcal{O}$.

Input: Target model f, Initial time t_{init}.
Output: Confidence interval sequence.
1 **for** $t \in [t_{init}, t_{init+L-1}]$ **do**
2 Target model produces point forecasting $\hat{Y}_t^f$;
3 Set the point prediction of $\overline{f}$: $\hat{Y}_t^{\overline{f}} = \hat{Y}_t^f$;
4 Set the effective miscoverage rate $\alpha_t^{\overline{f}} = \alpha_t^f$;
5 Update quantile $Q(\mathcal{L}_t)$ according to Eqn. (4);
6 Calculate $c_t^{\overline{f}}$ according to Eqn. (5);
7 Calculate ℓ, and update $\alpha_{t+1}^{\overline{f}}$ according to Eqn. (6);

Algorithm 3: IPOC.

Input: Data stream $\mathcal{X} = \{x_1, x_2, \cdots x_T\}$, Pool of base models $\mathcal{F} = \{f^1, \cdots, f^{|\mathcal{F}|}\}$, Initial states of base models and ensemble model , Target model selector $\mathcal{A}$, Chasing oracle $\mathcal{O}$.
1 **for** $1 \le t \le T$ **do**
2 Invoke $\mathcal{A}$ to select a target model $\overline{f}$ from $\mathcal{F}$;
3 **if** *Target model at time t differs from that at* $t-1$ **then**
4 Invoke $\mathcal{O}$ with target model $\overline{f}$;
5 Calculate $C_t^{\overline{f}}$ with Alghrithm 1;
6 **else**
7 Continue the existing run of Chasing Oracle $\mathcal{O}$ and calculate corresponding $C_t^{\overline{f}}$;
8 **foreach** *base model* $f \in \mathcal{F}$ **do**
9 Simulate f on X_t and get C_t^f;
10 Calculate loss $\ell(C_t^f, Y_t)$;
11 Update the state of each model;

their confidence intervals are likely to be different. After Y_t is revealed, the ensemble model updates its parameter α_t^f to α_{t+1}^f (Line 7). The chasing oracle ensures that the ensemble model moves in the direction of the target model, but this process will bring ultra losses, which is defined as *chasing regret*, and calculated by the following formula:

$$CR \triangleq \sum_{t=t_{init}}^{t_{final}} \mathbb{E}[\ell(C_t^{\overline{f}}, Y_t)] - \sum_{t=t_{init}}^{t_{final}} \ell(C_t^f, Y_t)$$

where t_{init}, t_{final} are the starting and finishing time.

B. Design of *IPOC* Algorithm

The framework of **IPOC** is shown in Fig. 3, and the specific complement of **IPOC** is shown in Algorithm 3. **IPOC** combines the chasing oracle $\mathcal{O}$ with a model selector $\mathcal{A}$. At the beginning, the first L predictions are used to initialize the initial states and loss sets of each base model and the ensemble model. For each time stamp $t \in [1, T]$, **IPOC** first utilizes the model selector to determine a target model (Line 2), and then invokes the chasing oracle to produce confidence intervals (Line 3–7). According to the loss and the state transition function, **IPOC** calculates loss and update state (Line 8–10). Theoretically, **IPOC** with chasing oracle $\mathcal{O}$ and target selector FTPL algorithm $\mathcal{A}$ has a *sub-linear* regret bound.

Algorithm 4: FTPL-CI.

Input: Pool of base model $\mathcal{F} = \{f^1, \cdots, f^{|\mathcal{F}|}\}$, Time Stamp t, Distribution of perturbation $\mathbb{D}_{per}$, Loss function ℓ, Distribution of loss $\mathbb{D}_{l_t^i}$ for each model $f^i \in \mathcal{F}$, Repeat Times m.
Output: Target model f.
1 **foreach** *base model* $f^i \in \mathcal{F}$ **do**
2 Sample the loss $\widehat{l_t^i} \sim \mathbb{D}_{l_t^i}$;
3 **for** $j = 1, \ldots, m$ **do**
4 **foreach** *base model* $f^i \in \mathcal{F}$ **do**
5 Sample the perturbation $\sigma_{t,i} \sim \mathbb{D}_{per}$;
6 $f_{t,j} \leftarrow \text{argmin}_{f^i \in \mathcal{F}} \widehat{l_t^i} - \sigma_t^i + \sum_{k=t-L}^{t-1} l(C_k^{f^i}, Y_k)$;
7 Sample f from the empirical distribution over $\{f_{t,1}, f_{t,2}, \cdots, f_{t,m}\}$ and select it as the target model;

C. Further Discussion About the Ensemble Module

Following The Perturbed Leader (FTPL) [35] is a representative algorithm for Online Learning with Switching Cost problem. FTPL incorporates perturbation terms to make the worst-case regret bound $O(T^{1/2})$, which is widely used in many fields of online learning [36], [37]. Recent works [38], [39] demonstrate that a simple modification of FTPL can achieve better regret guarantees when the sequence of loss functions is predictable while retaining the optimal worst-case regret guarantee for unpredictable sequences. Based on the above works, we design Following The Perturbed Leader with Conformal Inference (FTPL-CI). FTPL-CI uses loss function statistics (specifically, we construct an empirical distribution using the recent partial loss functions and sample from it to obtain an estimate of the next loss function) to estimate the loss function at the next moment, as depicted in Algorithm 4.

VI. Theoretical Analysis

A. The Effectiveness of ACCI

Theorem 1: The average miscoverage ratio of confidence intervals $\{\mathcal{C}_t\}_{t=1}^T$ will converge to α, i.e.

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \prod_{j=1}^{h} (x_t^j \notin \mathcal{C}_t^j) \overset{a.s.}{=} \alpha.$$

Proof: We first demonstrate that for a fixed α, the coverage rate will be equal or less than α, and then further prove that for the adaptive version as using adaptive α_t instead of α, the actual miscoverage rate will gradually converge to α. The proof is inspired by that of conformal prediction in [27], [30], [40]. The following lemma shows the validity of the conformal predictive distributions, whose cumulative distribution function is constructed as (7).

Lemma 3 ([41]): Given a loss function $\ell : \mathcal{Z} \to \mathbb{R}$ and a data sample $z \sim \mathcal{Z}$, calculate the loss as $l = \ell(z)$. Then, the cumulative distribution $\hat{F}_j(\cdot)$ constructed as (7) is valid in the sense that $\mathbb{P}_{\mathcal{Z}}[\hat{F}_j(s_j) \le 1-\alpha] = 1-\alpha$, for any $0 < \alpha < 1$.

Given a test data sample $z_t = (X_{t-L:t-1}, Y_{t:t+H-1}) \sim \mathcal{Z}$, we want to prove that the confidence intervals $C_t =$

$\{c_t, \ldots, c_{t+H-1}\}$ output by ACCI satisfies:

$$\mathrm{P}\left[y_j \in C_j\right] \geq 1-\alpha, \quad \forall j \in \{t+1, \ldots, t+H\}$$

Define a partial order for k-dimensional vectors $\preceq$ as $\mathbf{u} \preceq \mathbf{v}$ i.f.f. $\forall j \in \{1, \ldots, h\}, \mathbf{u}_j \leq \mathbf{v}_j$. For every data point in $\mathcal{L}_{cop}$, we evaluate the cumulative probability of the loss metric with the estimated conformal predictive distributions: $\mathcal{U} = \{\mathbf{u}^i\}_{i \in \mathcal{L}_{\text{cop}}}$, $\mathbf{u}^i = (u_1^i, \ldots, u_H^i) = (\hat{F}_1(s_1^i), \ldots, \hat{F}_k(s_H^i))$.

We define an empirical multivariate quantile function for $\mathcal{U}$, a set of k-dimensional vectors, based on the partial order:

$$Q(\mathcal{U}; \alpha) \triangleq \inf_{\mathbf{u}^*} \left\{ \mathbf{u}^* \;\middle|\; \left(\frac{1}{|\mathcal{U}|} \sum_{\mathbf{u} \in \mathcal{U}} sign(\mathbf{u} \preceq \mathbf{u}^*) \right) \geq 1-\alpha \right\}.$$

We first calculate $\mathbf{u}_j = \hat{F}_j(\ell(z_t)_j)$ for $j \in \{1, \ldots, h\}$. Let $\mathbf{u}^* = Q(\mathcal{U} \cup \{\infty\}; \alpha), \mathbf{u}^* \in [0,1]^h$. An important observation for the conformal prediction proof is that if $\mathbf{u}^* \preceq \mathbf{u}_t$, then

$$Q(\mathcal{U} \cup \{\infty\}; \alpha) = Q\left(\mathcal{U} \cup \{\mathbf{u}_t\}; \alpha\right),$$

the quantile remains unchanged. This fact can be re-written as

$$\mathbf{u}_t \preceq Q(\mathcal{U} \cup \{\infty\}; \alpha) \Longleftrightarrow \mathbf{u}_t \preceq Q\left(\mathcal{U} \cup \{\mathbf{u}_t\}; \alpha\right)$$

The above describes the condition where $\mathbf{u}_t$ is among the $\lceil (1-\alpha)t \rceil$ smallest of $\mathcal{U}$. By exchangability, the probability of $\mathbf{u}_t$'s rank among $\mathcal{U}$ is uniform. Therefore,

$$\mathrm{P}\left[\mathbf{u}_t \preceq Q(\mathcal{U} \cup \{\infty\}; \alpha)\right] = \frac{\lceil (1-\alpha)(|\mathcal{U}|+1) \rceil}{(|\mathcal{U}|+1)} \geq 1-\alpha \tag{10}$$

Note again that:

1) $\mathbf{u}^* = Q(\mathcal{U} \cup \{\infty\}; \alpha) = (\hat{F}_1(s_1^*), \ldots, \hat{F}_t(s_H^*))$
2) $\mathbf{u}_t = (\hat{F}_1(s_t), \ldots, \hat{F}_1(s_{t+H}))$
3) The confidence intervals are constructed as (Algorithm 1, line 9):

$$C_j \leftarrow \left\{x : \|\mathbf{x} - \hat{\mathbf{x}}_j\| < s_j^*\right\} \tag{11}$$

By definition of $\preceq$, we have

$$\begin{aligned} \mathbf{u}_t \preceq \mathbf{u}^* &\Longleftrightarrow \forall j \in \{0, \ldots, H-1\}, (\mathbf{u}_t)_j \leq \mathbf{u}_j^* \\ &\overset{\text{Lemma 3}}{\Longrightarrow} \forall j \in \{0, \ldots, H-1\}, (s_t)_j \leq s_j^* \\ &\overset{(11)}{\Longleftrightarrow} \forall j \in \{0, \ldots, H-1\}, x_{t+j} \in C_j \end{aligned} \tag{12}$$

Combining (10) and (12), we have

$$\begin{aligned} \mathrm{P}\left[X_t \in C_t\right] &\geq \mathrm{P}\left[\mathbf{u}_t \preceq Q(\mathcal{U} \cup \{\infty; \alpha\})\right] \\ &\geq 1-\alpha \end{aligned} \tag{13}$$

It should be noted that the above proof is based on the assumption that the dataset is *exchangeable*. Further, we discuss that replacing the fixed α with the adaptive miscoverage rate α_t, and updating it in an online manner will keep the validity of the conformal prediction without relying on this assumption.

Lemma 4: With probability one we have that $\forall t \in \mathbb{N}, \alpha_t \in [-\gamma, 1+\gamma]$.

Proof: Assume by contradiction that with positive probability $\{\alpha_t\}_{t \in \mathbb{N}}$ is such that $\inf_t \alpha_t < -\gamma$ (the case where $\sup_t \alpha_t > 1+\gamma$ is identical). Note that $\sup_t |\alpha_{t+1} - \alpha_t| = \sup_t \gamma |\alpha - \mathrm{err}_t| < \gamma$. Thus, with positive probability we may find $t \in \mathbb{N}$ such that $\alpha_t < 0$ and $\alpha_{t+1} < \alpha_t$. However,

$$\begin{aligned} \alpha_t < 0 &\Longrightarrow \hat{Q}_t(1-\alpha_t) = \infty \Longrightarrow \mathrm{err}_t = 0 \\ &\Longrightarrow \alpha_{t+1} = \alpha_t + \gamma(\alpha - \mathrm{err}_t) \geq \alpha_t \end{aligned}$$

and thus $\mathbb{P}(\exists t \text{ such that } \alpha_{t+1} < \alpha_t < 0) = 0$. We have reached a contradiction. With probability one we have that for all $T \in \mathbb{N}$,

$$\left| \frac{1}{T} \sum_{t=1}^{T} err_t - \alpha \right| \leq \frac{\max\{\alpha_1, 1-\alpha_1\} + \gamma}{T\gamma}.$$

In particular, $\lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathrm{err} \overset{\text{a.s.}}{=} \alpha$. □

B. The Effectiveness of Chasing Oracle $\mathcal{O}$

Theorem 2: With $L_c \leq L$, the chasing regret generated by the chasing oracle $\mathcal{O}$ corresponding to MOTSF-Int is $O(L_c)$.

Proof of Theorem 2 According to the definition of state s_t^f and loss set $\mathcal{L}_t$ for the model f, if $L_c < L$, we can find that after chasing L_c time-steps as Algorithm 2, the state of ensemble model $\overline{f}$ will be completely consistent with the state of target model f. Therefore, the chasing regret of $\mathcal{O}$ is

$$\begin{aligned} CR &\triangleq \sum_{t_{init}}^{t_{final}} \ell(C_t^{\overline{f}}, Y_t) - \sum_{t_{init}}^{t_{final}} \mathbb{E}[\ell(\mathbf{c}_t^f, Y_t)] \\ &= \sum_{t=t_{init}}^{t_{init}+L_c} l(C_t^{\overline{f}}, Y_t) - l(C_t^f, Y_t). \end{aligned}$$

As $l(\cdot,\cdot) \in [0,1]$, the chasing regret of $\mathcal{O}$ is obviously $O(L_c)$□

C. The Effectiveness of *IPOC*

Theorem 3: The regret of **IPOC** with chasing oracle $\mathcal{O}$ and target selector FTPL algorithm $\mathcal{A}$ is $O(\sqrt{L_c T \log|\mathcal{F}|})$.

Proof: We reduct the analysis of proposed IPOC to the analysis of OLSC Algorithm. When the chasing regret is verified to be sublinear, the performance analysis of Algorithm 3 can draw support from FTPL algorithm $\mathcal{A}$. The key idea of the technique here is to utilize FTPL to make decisions in model selection over data stream $\mathcal{X}$ and map the chasing regret incurred by the chasing oracle $\mathcal{O}$ to a switching cost that can be accommodated by $\mathcal{A}$.

With the model selector $\mathcal{A}$ and a chasing oracle $\mathcal{O}$, our main algorithm **IPOC** operates as follows. The selector $\mathcal{A}$ chooses a proper target model sequence $\{\overline{f}_t\}_{t=1}^T$ and $\mathcal{O}$ produces a confidence interval consequence $\{C_t^{\overline{f}}\}_{t=1}^T$ based on the selected target model at each time stamp. The regret of **IPOC** with σ-chasing regret can be analyzed with the help of $\mathcal{A}$ satisfying the configurations as follows. Recall the definition of OLSC under the full information setting. Given a finite expert set $\mathcal{F}$ and T time-steps, the decision maker is required to pick and chase an expert sequence $\{\overline{f}_t\}_{t=1}^T$ in an online mode to minimize the regret, where the expert loss function $\ell : \mathcal{F} \mapsto [0,1)$ is revealed at the end of each step and an extra switching cost is incurred whenever switching from one expert to another. The regret of

OLSC can be defined as:

$$\min_{f\in\mathcal{F}}\left(\Delta\sum_{t=2}^{T} sign(\overline{f}_t \neq \overline{f}_{t-1}) + \sum_{t=1}^{T}\mathbb{E}[\ell(C_t^{\overline{f}}, Y_t)]\right) - \sum_{t=1}^{T}\ell(C_t^{f}, Y_t)$$

The FTPL algorithm $\mathcal{A}$ is utilized as a strategy selector with the following conversions in a similar way with [33]:

1) the expert set of $\mathcal{A}$ is identified with the base model pool $\mathcal{F}$ in the ensemble-based MOTSF-Int problem;
2) the number of steps of $\mathcal{A}$ equals to that in the ensemble-based MOTSF-Int problem, denoted by T;
3) the switching cost of $\mathcal{A}$ is set to $\Delta = \sigma$.
4) The strategy selector $\mathcal{A}$ helps us to establish the following guarantee on the final regret, which can be proved in an analogous way to the corresponding result in [33].

Next, we prove that the regret of **IPOC** with a chasing oracle $\mathcal{O}$ of chasing regret σ for ensemble-based MOTSF-Int instances is $O(\sqrt{\sigma \cdot T\log|\mathcal{F}|})$. Referring to [33], we split the T time-steps into episodes $\{1, 2, \cdots\}$. Each episode λ represents a maximal contiguous step sequence during which $\mathcal{A}$ keeps the model selection $\overline{f} = f^{\lambda}$ unchanged. Assuming that t_λ and t'_λ denote the first and last step of episode λ respectively, **IPOC** follows the point prediction $x_t^{\overline{f}}$ and confidence interval $\mathbf{c}_t^{\overline{f}}$ generated by $\mathcal{O}$ during $t \in [t_\lambda, t'_\lambda]$ and the chasing regret of $\mathcal{O}$ is upper bounded by $\sigma = \Delta$. The chasing regret follows that

$$\sum_{t=t_\lambda}^{t'_\lambda}\mathbb{E}[\ell(C_t^{\overline{f}}, Y_t)] - \sum_{t=t_\lambda}^{t'_\lambda}\ell(C_t^{f^\lambda}, Y_t) \leq \Delta.$$

Therefore, for each strategy $f \in \mathcal{F}$, we have

$$\sum_{t=1}^{T}\mathbb{E}[\ell(C_t^{\overline{f}}, Y_t)] - \sum_{t=1}^{T}\ell(C_t^{f}, Y_t) \leq \left(\Delta\sum_{t=2}^{T} sign(\overline{f}_t \neq \overline{f}_{t-1}) + \sum_{t=1}^{T}\mathbb{E}[\ell(C_t^{\overline{f}}, Y_t)]\right) - \sum_{t=1}^{T}\ell(C_t^{f}, Y_t)$$

By Lemma 2, the regret of **IPOC** with FTPL $\mathcal{A}$ is at most

$$O(\sqrt{\Delta \cdot T\log|\mathcal{F}|}) = O(\sqrt{\sigma \cdot T\log|\mathcal{F}|}).$$

With the conclusion and Theorem 2, we can prove Theorem 3. □

VII. Empirical Study

A. Experimental Settings

1) Dataset: We use 5 real-world data streams, in which 4 are private and 1 is open source, representing the majority of scenarios in large-scale systems. Table II shows the details.

1) *HDFS:* A highly fault-tolerant system that contains the **daily** total usage stream with not-stationary, sudden drifts.
2) *Abase:* A distributed key-value storage system holds the **hourly** query streams with various patterns and drifts.
3) *FaaS:* A serverless computing system for executing application functions includes the **minutely** request stream.
4) *IaaS:* A cloud computing service model for delivering fundamental computing resources, which includes the **high-frequency fluctuating** computing request stream.
5) *Alibaba-cluster-trace-v2018:* The trace sampled from real production clusters with many missing values. We take the CPU usage at **second** level as the data stream.

TABLE II
Statistics of the Datasets for Experiment

Dataset	Observations Per Sequence	Number of Sequences	Sampling Frequency	Workload Type
HDFS	221	10	Day	Capacity Usage
Abase	696	10059	Hour	Queries
FaaS	23041	1014	Minute	Requests
IaaS	69491	622	Minute	CPU Usage
Alibaba	6928	1154	Second	CPU Usage

2) Metrics: We use root mean square error (RMSE) and Symmetric Mean Absolute Percentage Error (SMAPE) to evaluate the quality of point predictions, and for the quality of confidence intervals, we utilize ρ-risk, average coverage ratio (ACR), and median interval width as metrics.

B. Baseline Setting

1) Single Base Model Set-Up: In **IPOC**, we constructs a pool of base models $\mathcal{F}$ containing $|\mathcal{F}|$ independently trained predictors. The pool is organised into three families according to the underlying modelling paradigm:

- *Statistical:* **ETS** [16], **ARIMA** [15], **Prophet** [17].
- *Machine-learning:* **GPR** [42], **SVR** [43], **RFR** [19], **ETR** [44], **GBR** [18], **DTR** [45], **Bagging** and **Adaboost** models [19], [20] (using either GPR or DTR as the base regressor).
- *Deep-learning:* **RNN** [46], **LSTM** [22], **GRU** [21], **PatchTST** [11], **TimeMixer** [13], **DLinear** [12].

In total, the experimental pool comprises 25 concrete base predictors obtained by instantiating the above families with different hyper-parameter settings, enabling the proposed method to cope with diverse data-stream characteristics.

2) Baseline Methods: We compare the performance of **IPOC** with the following methods. First of all, three ensemble strategies are implemented with the same base model setting of **IPOC**. **FFORMS:** An ensemble strategy training a meta-learner to select the prediction of the proper base model as the ensemble output [47]. **FFORMA:** An ensemble strategy which outputs continuous weights for different base models by a meta-learner to ensemble their forecasts [48]. In our experimental settings, both FFORMS and FFORMA take LightGBM [49] as the meta-learner. ARIMA, GPR, RFR, ETR, Prophet, ETS are also used as baselines in our experiment.

*3) Flexible Ensemble Strategy Choice for **IPOC**:* In Chapters IV-VI, we design a ensemble framework based on the FTPL algorithm and conduct theoretical analyses. In practice,

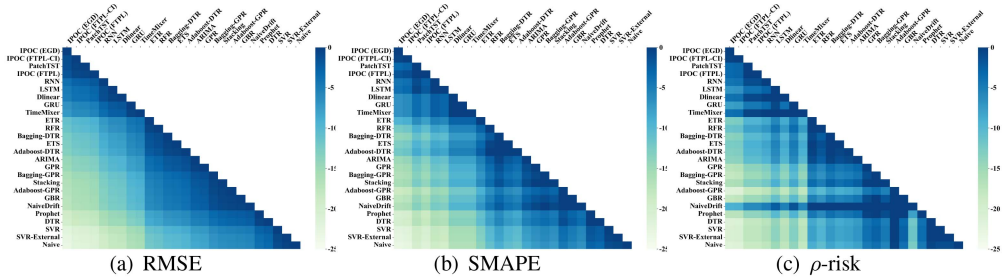

Fig. 4. Comparison of Prediction Accuracy Ranking between **IPOC** and Base Models on multi-step forecasting. We compare them on RMSE, SMAPE, and ρ-risk. The rank of each model is the average of its ranks on all datasets.The model with a darker grid performs better in a column of the three subfigures.

thanks to the flexibility of **IPOC**, it can be easily integrated with other ensemble modules to achieve better performance. Exponentiated Gradient Descent (EGD) [50] is a commonly used ensemble method. Specifically, the decision space $\triangle$ is a d-dimensional simplex, i.e. $\triangle = \{\mathbf{w}_t \mid w_{t,i} \geq 0 \text{ and } \|\mathbf{w}_t\|_1 = 1\}$, where t is the time step indicator and we omit the subscript t for simplicity when it's not confusing. Given the online data stream X_t, its forecasting target Y_t, and $|\mathcal{F}|$ forecasting experts $\{f_i(X_t)\}_{i=1}^d$, the player's goal is to minimize the forecasting error as $\min_{\mathbf{w}} \| \sum_{i=1}^{|\mathcal{F}|} w_i f_i(X_t) - Y_t \|^2 \quad \text{s.t.} \quad \mathbf{w} \in \triangle$. According to EGD, choosing $\mathbf{w}_1 = [w_{1,i} = 1/d]_{i=1}^d$ as the center point of the simplex, the updating rule for each w_i is:

$$w_{t+1,i} = \frac{w_{t,i} \exp\left(-\eta \|f_i(X_t) - Y_t\|^2\right)}{Z_t} \tag{14}$$

where $Z_t = \sum_{i=1}^d w_{t,i} \exp(-\eta \|f_i(X_t) - Y_t\|^2)$ is the normalizer. In theory, the algorithm can achieve the regret bound sublinear in T, and weighted summation has more flexibility.

C. Evaluation Results

1) Comparison With Base Models: In this part, we compare **IPOC** with its base models as shown in Fig. 4. For types with multiple base models, the best rank among them is chosen as the representative. Each grid in a subfigure represents the difference between the model rankings on the X-axis and Y-axis. From the first columns of Fig. 4(a)-(b), **IPOC** achieves the best performance in all three metrics (RMSE, SMAPE, ρ-risk), indicating its reasonable ensemble of different base models, including the newly added deep learning models PatchTST, TimeMixer, and DLinear, for better predictions. To explore **IPOC**'s ensemble mechanism, we visualize the selected rates of various base models on five datasets in Fig. 5. It shows that **IPOC** has different proportions of selecting base models across datasets. For instance, DLinear is most selected in FaaS but rarely in Abase, suggesting **IPOC** can adaptively select suitable models according to dataset characteristics.

2) Comparision With Baselines: We compare **IPOC** with six traditional statistical methods, three state-of-the-art deep learning baselines, and two classic ensemble strategies. To enable interval prediction for all baselines, we implement the Adaptive Copula Conformal Inference (ACI) module with a significance level $\alpha = 0.1$. Table III summarizes the experimental results across four key metrics: RMSE, MAE, SMAPE(%), and ρ-risk,

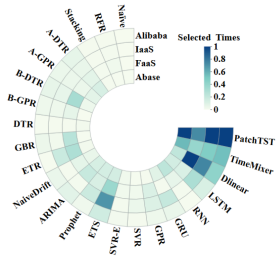

Fig. 5. Selectivity of IPOC. Cells show the selection frequency of models in each dataset after Min-Max normalization.

where the top two performances for each metric in each dataset are bolded. Across all datasets and metrics, **IPOC** consistently achieves the lowest or near-lowest values. It outperforms not only traditional statistical methods and classic ensemble strategies but also the newly introduced deep learning baselines. For example, on the Abase dataset, **IPOC**-E achieves an RMSE of 0.1103, which is 22.9% lower than the best deep learning baseline and 19.3% lower than the best classic ensemble. Notably, single models—whether traditional statistical methods or deep learning baselines—exhibit significant performance instability across datasets. For instance, ARIMA achieves a relatively low RMSE on Alibaba but a poor RMSE on IaaS. In contrast, **IPOC** effectively combine the strengths of diverse base models, maintaining robust and accurate predictions across different datasets.

To explore the performance of **IPOC** on interval prediction further, we compare **IPOC** with two methods that can produce interval predictions. We still set the miscoverage rate $\alpha = 0.1$. Table IV exhibits the results on 5 datasets. In single-step forecasting tasks, such as on HDFS, the traditional statistical methods performs well, but their performance will be greatly degraded in multi-step prediction tasks, while ARIMA and GPR fail to guarantee to contain the data stream at nearly 90%, which means their confidence intervals are unreliable. In contrast, we note that the coverage ratio of **IPOC** keeps close to 90% in all datasets. We attribute such phenomenon to the effective ACCI algorithm which is used to improve the adaptation when the data stream is highly shifted.

D. Efficiency Analysis

Compared to other similar ensemble methods, the advantage of IPOC lies in its almost non-parametric nature (considering

TABLE III
PERFORMANCE COMPARISON BETWEEN **IPOC** AND BASELINES: SINGLE-STEP PREDICTION ON HDFS AND MULTI-STEP PREDICTION (AVERAGE $h = 1/5/10/30$) ON ABASE, FAAS, IAAS, AND ALIBABA

Datasets	Metrics	ARIMA*	GPR*	RFR*	ETR*	Prophet*	ETS*	PatchTST*	TimeMixer*	DLinear*	FFORMS*	FFORMA*	**IPOC**-F	**IPOC**-C	**IPOC**-E
HDFS	RMSE	**0.0365**	0.0794	0.0668	0.0505	0.1225	0.0402	0.1064	0.1142	0.0852	0.0421	0.0399	0.0588	**0.0393**	0.0379
	MAE	0.0157	0.0310	0.0263	0.0203	0.0448	0.0157	0.0441	0.0398	0.0185	0.0154	**0.0151**	0.0186	**0.0156**	0.167
	SMAPE(%)	2.2940	4.5250	3.7755	2.7795	6.5750	2.2605	6.3974	5.8208	4.4733	2.2479	**2.1905**	2.7205	**2.2438**	2.4145
	ρ-risk	0.0058	0.0090	0.0071	0.0064	0.0090	**0.0054**	0.0097	0.0113	0.0085	0.0077	0.0074	0.0128	0.0063	**0.0056**
Abase	RMSE	0.2024	0.2463	0.2057	0.2059	0.2645	0.2062	0.1425	0.1704	0.1500	0.1370	0.1490	0.1151	**0.1124**	**0.1103**
	MAE	0.1419	0.1722	0.1498	0.1501	0.1886	0.1458	0.1076	0.1230	0.1166	0.1115	0.1043	0.0835	**0.0805**	**0.0786**
	SMAPE(%)	8.2936	10.055	8.6032	8.2201	11.072	8.3968	6.2688	7.1390	6.8182	6.4144	6.0308	4.8808	**4.6158**	**4.5563**
	ρ-risk	0.1894	0.2163	0.2021	0.1916	0.2525	0.1916	0.1277	0.1430	0.1304	0.1342	0.1279	0.1000	**0.1002**	**0.0976**
FaaS	RMSE	0.4686	0.4594	0.4500	0.4400	0.5079	0.4541	0.3863	0.4588	0.4106	0.4014	0.4193	0.2945	**0.2894**	**0.2861**
	MAE	0.2903	0.3106	0.2928	0.2915	0.3489	0.2857	0.2345	0.2899	0.2899	0.2407	0.2447	0.1777	**0.1741**	**0.1743**
	SMAPE(%)	16.970	18.134	16.810	15.965	20.484	16.455	13.671	16.823	16.957	13.850	14.152	10.393	**9.9808**	**10.105**
	ρ-risk	0.1482	0.1529	0.1301	0.1315	0.1682	0.1410	0.1058	0.0968	0.1070	0.1032	0.0989	0.0866	**0.0801**	**0.0776**
IaaS	RMSE	1.9035	0.6361	0.6811	0.6547	1.4417	1.8449	0.4475	0.5641	0.5141	0.4191	0.5299	0.3271	**0.3225**	**0.3183**
	MAE	1.4883	0.4039	0.4123	0.3938	1.2594	1.4432	0.2977	0.3355	0.3016	0.2723	0.2850	**0.2066**	0.2089	**0.2044**
	SMAPE(%)	24.495	11.793	11.836	10.784	21.965	23.564	8.6754	9.7332	8.8229	7.8340	8.2417	6.0406	**5.9892**	**5.9266**
	ρ-risk	0.3294	0.2849	0.2623	0.2864	0.3161	0.3043	0.1924	0.2184	0.1701	0.1789	0.1752	0.1371	**0.1298**	**0.1267**
Alibaba	RMSE	0.5062	0.3737	0.5177	0.5123	0.5682	0.6531	0.3733	0.4731	0.4544	0.3656	0.4132	0.2912	**0.2829**	**0.2779**
	MAE	0.3788	0.2828	0.3960	0.3908	0.4364	0.4929	0.3324	0.3379	0.3870	0.3176	0.2964	0.2653	**0.2556**	**0.2563**
	SMAPE(%)	22.140	16.512	22.741	21.403	25.619	28.390	19.373	19.605	22.642	18.272	17.140	15.516	**14.653**	**14.361**
	ρ-risk	0.1700	0.1745	0.2282	0.2400	0.2064	0.2820	0.1612	0.1576	0.1704	0.1654	0.1426	0.1313	**0.1257**	**0.1215**

[1] Models with * mean that their prediction intervals are generated through ACI.
[2] **IPOC**-F, **IPOC**-C, and **IPOC**-E represent **IPOC** with FTPL, FTPL-CI, and EGD, respectively.

TABLE IV
COMPARISON OF INTERVAL FORECASTING PERFORMANCE

Datasets	Metrics	ARIMA	GPR	**IPOC**-F	**IPOC**-C	**IPOC**-E
HDFS	ρ-risk	0.0101	0.0115	0.0128	**0.0063**	**0.0056**
	ACR	0.8696	0.7911	**0.8941**	0.8922	**0.8935**
	MW	0.0275	0.0242	0.0304	**0.0237**	**0.0205**
Abase	ρ-risk	0.2369	0.2781	**0.1000**	0.1002	**0.0976**
	ACR	0.4379	0.3981	0.8847	**0.9071**	**0.9001**
	MW	0.2676	0.3741	0.2049	**0.1873**	**0.1698**
FaaS	ρ-risk	0.1558	0.1744	0.0866	**0.0801**	**0.0776**
	ACR	0.3841	0.4263	0.8932	**0.8978**	**0.9015**
	MW	**0.1817**	0.2148	0.2015	**0.1913**	0.1879
IaaS	ρ-risk	0.3342	0.3050	0.1371	**0.1298**	**0.1267**
	ACR	0.6743	0.4821	**0.9007**	0.8940	**0.9015**
	MW	0.3061	0.2075	0.1940	**0.1904**	**0.1879**
Alibaba	ρ-risk	0.1508	0.1742	0.1313	**0.1257**	**0.1215**
	ACR	0.3609	0.4510	0.8961	**0.9049**	**0.8992**
	MW	0.6091	0.7037	**0.4975**	0.5134	**0.4822**

TABLE V
THE AVERAGE TIME (MS) COST BY DIFFERENT ENSEMBLE STRATEGIES

Dataset	HDFS	Abase	Alibaba	FaaS	IaaS
IPOC(EGD)	6.57	7.31	7.22	7.37	7.36
IPOC(FTPL-CI)	6.68	7.55	7.46	7.71	7.92
IPOC(FTPL)	**4.81**	**4.81**	**5.17**	**5.16**	**5.23**
FFORMS	10.03	28.38	11.50	14.85	19.21
FFORMA	11.01	30.80	13.16	17.22	21.98

TABLE VI
COMPARISON OF DIFFERENT CONFORMAL INFERENCE METHODS. MW MEANS MEDIAN WIDTH, AND ACR MEANS AVERAGE COVERAGE RATIO.

Models	Abase		FaaS		IaaS		Alibaba	
	MW	ACR	MW	ACR	MW	ACR	MW	ACR
ACI	**0.163**	0.866	**0.162**	0.859	**0.367**	0.870	0.467	0.873
CopulaCPTS	0.191	0.913	0.224	0.919	0.471	**0.905**	0.525	**0.901**
AcMCP	0.193	0.902	0.212	0.909	0.440	0.894	0.531	0.902
SAOCP	0.147	0.869	0.182	0.875	0.396	0.882	**0.458**	0.867
ACCI(ours)	0.170	**0.900**	0.188	**0.902**	0.371	0.892	0.482	**0.899**

only the ensemble strategy part). The only variable that needs to be updated during the learning process is the loss set, which makes it very intuitive and efficient compared to other models. Table V shows the time required for the ensemble models to perform each ensemble step (in milliseconds). Compared with FFORMA, **IPOC**-E saves 72% time cost on average.

E. Sensitivity Analysis

This section evaluates **IPOC**'s performance under different hyperparameter settings to explore their impacts.

1) History Series Length L: We test different L with fixed L_c and α, and Fig. 6(a) presents MAE and RMSE across five datasets. The optimal L varies across datasets due to their distinct characteristics, and a larger L does not guarantee higher accuracy. For example, HDFS achieves lower MSE at $L = 35$ than $L = 50$—excessive L may incorporate outdated information before concept drifts, interfering with predictions. Thus, a proper L is more effective for capturing concept drifts in online time series forecasting than a blindly long one.

2) Loss Set Size L_c: We test $L_c = \{10, 20, 30, 40, 50\}$ with fixed $L = 30$ and $\alpha = 0.1$, and results are shown in Fig. 6(b). An optimal L_c exists for each dataset (especially when $L_c \leq L$), as L_c determines how ACCI's confidence interval captures workload patterns and concept drifts. Notably, $L_c > L$ does not

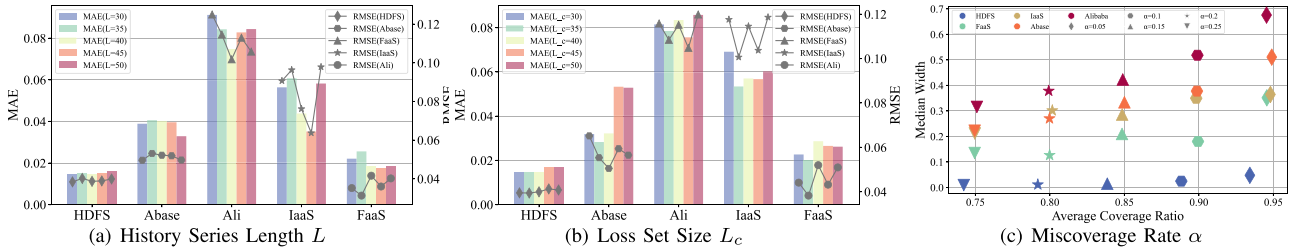

Fig. 6. Sensitivity Analysis of **IPOC**. Fig. 6(a), (b) describe the prediction accuracy of **IPOC** with different settings of L and L_c, on five datasets. The column and the line exhibit the MAE and RMSE results respectively. Fig. 6(c) illustrates the relationship between the miscoverage rate α and median interval width under different α. Each shape of points represents a unique α.

improve performance, consistent with online learning's limited observation constraints.

3) Miscoverage Rate α: We test different α with fixed L, L_c, and Fig. 6(c) shows the coverage rate-median width relationship. **IPOC**'s coverage ratios are close to $1-\alpha$, verifying reliable interval prediction. Additionally, median interval width expands with increasing coverage rate—intuitively, a wider interval is required to cover more data points.

F. The Effectiveness of ACCI

To evaluate the effectiveness of the proposed ACCI module, we compare it against four representative conformal inference methods. The performance is assessed across two dual dimensions: interval reliability, measured by the Average Coverage Ratio (ACR, where values closer to the target confidence level of 0.9 are optimal), and interval precision, measured by the Median Width (MW, where smaller values indicate sharper predictions). The quantitative results across four datasets are summarized in Table VI. As shown in the table, ACCI demonstrates superior coverage reliability. Specifically, its ACR stably ranges from 0.892 to 0.902 across all datasets, successfully avoiding the severe under-coverage issues observed in adaptive single-step methods like ACI and SAOCP. For instance, SAOCP yields an ACR of only 0.869 on the Abase dataset. The performance degradation of these baselines stems from their independent treatment of forecasting steps, which fails to capture the high-dimensional joint dependency structures inherent in multi-step trajectories. In contrast, IPOC utilizes Empirical Copulas to explicitly model these dependencies, ensuring valid joint coverage.

Furthermore, ACCI achieves a better balance between high coverage and narrow intervals compared to other multi-step or copula-based methods. While CopulaCPTS also utilizes copulas to maintain high coverage, its reliance on a fixed calibration scope makes it overly conservative in dynamic environments with concept drifts, resulting in unnecessarily wide intervals. On the Abase dataset, ACCI reduces the MW to 0.170 compared to 0.191 for CopulaCPTS and 0.193 for AcMCP (a compression of over 11%). Unlike AcMCP, which primarily focuses on residual sequence calibration, ACCI operates dynamically by updating the miscoverage rate α_t in real-time, allowing the interval width to adapt tightly to data volatility. Ultimately, integrating ACCI within the holistic IPOC ensemble framework (FTPL) guarantees not only adaptive uncertainty quantification but also regret-minimized model selection, yielding robust and balanced interval predictions.

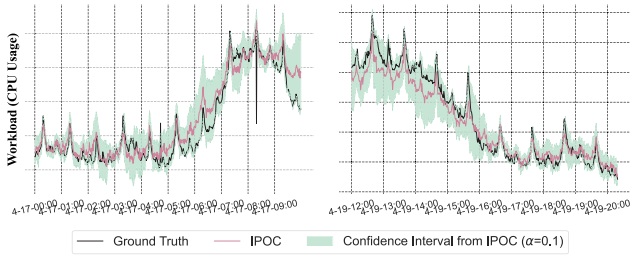

Fig. 7. Visualizing **IPOC**'s forecasts on IaaS dataset.

G. Visualization of *IPOC* Forecasts

Some examples of forecasting results for **IPOC** on IaaS are visualized in Fig. 7. The IaaS dataset exhibits drastic trend drifts and high-frequency fluctuations, which makes it challenging to forecast. We can observe that forecasts rapidly fit the sequence and the prediction intervals cover the workload accurately, especially during the process of trend drifts.

H. Case Study: Kubernetes Horizontal Pod Autoscaling

To assess the effectiveness of **IPOC** in a real-world production environment, we conduct a comparative analysis with other baseline forecasting models in the context of Kubernetes Horizontal Pod Autoscaling (HPA). In our experiment, we launch a test service on a Kubernetes cluster, applying a workload derived from historical Queries Per Second (QPS) data from a FaaS cluster.

Our analysis include comparisons between HPAs utilizing various forecasting models, and the Naïve HPA that comes with Kubernetes, with results detailed in Table VII and the result demonstrates that the **IPOC** system significantly enhances Quality of Service (QoS) by effectively forecasting future workloads. Compared with the Naïve HPA, **IPOC** reduces average latency by approximately 25%, as evidenced by the average number of pods. Specifically, the **IPOC**-E variant achieves the lowest average latency and maximum latency, with an average pod count of 15.328, outperforming the Naïve HPA's average latency of 0.299 s and maximum latency of 91.022 s. Relative to FFORMA, **IPOC** shows a 15.2% improvement in average latency, with

TABLE VII
RESULTS OF PREDICTIVE KUBERNETES HORIZONTAL POD AUTOSCALING. *AVE* STANDS FOR AVERAGE, *99.9-LAT*, *99-LAT* AND *90-LAT* REPRESENTS THE 99.9, 99 AND 90 PERCENTILE LATENCY, RESPECTIVELY.

Models	Ave-Lat(s)	Max-Lat(s)	99.9-Lat(s)	99-Lat(s)	90-Lat(s)	AvePod	MaxPod
Naïve	0.299	91.022	8.275	0.789	0.443	19.926	34
ARIMA	0.293	57.133	4.177	1.259	0.551	16.298	31
FFORMA	0.256	17.347	2.324	1.233	0.414	15.499	27
IPOC-F	0.261	13.285	2.454	1.197	0.415	**15.179**	25
IPOC-C	0.235	10.221	2.477	0.949	0.397	16.421	27
IPOC-E	**0.223**	**8.641**	**1.967**	**0.733**	**0.366**	15.328	**24**

fewer pods. For instance, the **IPOC**-E variant reduces average latency compared to FFORMA, while maintaining a lower average pod count. This indicates that **IPOC** optimizes resource utilization more efficiently. Notably, the **IPOC**-E variant reduces maximum latency by 90.5%, from 91.022 s in Naïve HPA to 8.641 s. Overall, **IPOC**-based HPAs outperform the Naïve HPA in both average and peak latency metrics while using fewer pods.

VIII. CONCLUSION

This study addresses online multi-step interval forecasting for large-scale system workloads: we formally define the task, model its ensemble learning as Dd-MDP, design the ACCI module for uncertainty quantification, and propose **IPOC**. Theoretically, **IPOC** guarantees sublinear regret and valid coverage; empirically, it outperforms 25 baselines on 5 datasets, boosting Kubernetes HPA's resource utilization by 18–22% and reducing latency by 30–35%. To the best of our knowledge, our framework is the first to integrate adaptive copula-based conformal inference with online ensemble learning for time series interval prediction. The potential limitation of our research is ensemble's parallel base-model resource overhead, and we will work for higher efficiency.

REFERENCES

[1] Z. Zhou et al., "AHPA: Adaptive horizontal POD autoscaling systems on alibaba cloud container service for kubernetes," in *Proc. Conf. Artif. Intell.*, 2023, pp. 15621–15629.

[2] S. Arbat, V. K. Jayakumar, J. Lee, W. Wang, and I. K. Kim, "Wasserstein adversarial transformer for cloud workload prediction," in *Conf. Artif. Intell.*, 2022, pp. 12433–12439.

[3] Y. Guo et al., "PASS: Predictive auto-scaling system for large-scale enterprise web applications," in *Proc. ACM Web Conf.*, 2024, pp. 2747–2758.

[4] L. Ma, D. Van Aken, A. Hefny, G. Mezerhane, A. Pavlo, and G. J. Gordon, "Query-based workload forecasting for self-driving database management systems," in *Proc. ACM Int. Conf. Manage. Data*, 2018, pp. 631–645.

[5] S. Das, F. Li, V. R. Narasayya, and A. C. König, "Automated demand-driven resource scaling in relational database-as-a-service," in *Proc. ACM Int. Conf. Manage. Data*, 2016, pp. 1923–1934.

[6] V. K. Jayakumar, J. Lee, I. K. Kim, and W. Wang, "A self-optimized generic workload prediction framework for cloud computing," in *Proc. Int. Parallel Distrib. Process. Symp.*, 2020, pp. 779–788.

[7] Y. Gao, X. Huang, X. Zhou, X. Gao, G. Li, and G. Chen, "DBAugur: An adversarial-based trend forecasting system for diversified workloads," in *Proc. Int. Conf. Data Eng.*, 2023, pp. 1–13.

[8] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "DeepAR: Probabilistic forecasting with autoregressive recurrent networks," *Int. J. Forecasting*, vol. 36, no. 3, pp. 1181–1191, 2020.

[9] Y. Chen, Y. Kang, Y. Chen, and Z. Wang, "Probabilistic forecasting with temporal convolutional neural network," *Neuro Comput.*, vol. 399, pp. 491–501, 2020.

[10] B. Lim, S. Ö. Arık, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time series forecasting," *Int. J. Forecasting*, vol. 37, no. 4, pp. 1748–1764, 2021.

[11] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," in *Proc. Int. Conf. Learn. Representations*, 2023.

[12] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 11121–11128.

[13] S. Wang et al., "TimeMixer: Decomposable multiscale mixing for time series forecasting," in *Proc. Int. Conf. Learn. Representations*, 2023.

[14] J. Chen et al., "IPOC: An adaptive interval prediction model based on online chasing and conformal inference for large-scale systems," in *Proc. ACM Conf. Knowl. Discov. Data Mining*, 2023, pp. 202–212.

[15] G. E. Box and G. M. Jenkins, "Some recent advances in forecasting and control," *J. Roy. Statist. Society. Ser. C Appl. Statist.*, vol. 17, no. 2, pp. 91–109, 1968.

[16] C. C. Holt, "Forecasting seasonals and trends by exponentially weighted moving averages," *Int. J. Forecasting*, vol. 20, no. 1, pp. 5–10, 2004.

[17] S. J. Taylor and B. Letham, "Forecasting at scale," *Amer. Statistician*, vol. 72, no. 1, pp. 37–45, 2018.

[18] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, pp. 1189–1232, 2001.

[19] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[20] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. System Sci.*, vol. 55, no. 1, pp. 119–139, 1997.

[21] K. Cho et al., "Learning phrase representations ssing RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*.

[22] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[23] B. Wang et al., "Deep uncertainty quantification: A machine learning approach for weather forecasting," in *Proc. ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2019, pp. 2087–2095.

[24] Z. Yu, X. Zheng, F. Huang, W. Guo, L. Sun, and Z. Yu, "A framework based on sparse representation model for time series prediction in smart city," *Front. Comput. Sci.*, vol. 15, pp. 1–13, 2021.

[25] L. Cao, B. Wang, G. Jiang, Y. Yu, and J. Dong, "Spatiotemporal-aware trend-seasonality decomposition network for traffic flow forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2025, pp. 11463–11471.

[26] M. S. Islam, W. Pourmajidi, L. Zhang, J. Steinbacher, T. Erwin, and A. Miranskyy, "Anomaly detection in a large-scale cloud platform," in *Proc. IEEE/ACM Int. Conf. Softw. Eng.: Softw. Eng. Pract.*, 2021, pp. 150–159.

[27] I. Gibbs and E. Candes, "Adaptive conformal inference under distribution shift," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 1660–1672.

[28] A. Bhatnagar, H. Wang, C. Xiong, and Y. Bai, "Improved online conformal prediction via strongly adaptive online learning," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 2337–2363.

[29] R. F. Barber, E. J. Candes, A. Ramdas, and R. J. Tibshirani, "Conformal prediction beyond exchangeability," *Ann. Statist.*, vol. 51, no. 2, pp. 816–845, 2023.

[30] S. Sun and R. Yu, "Copula conformal prediction for multi-step time series forecasting," in *Proc. Int. Conf. Learn. Representations*, 2024.

[31] X. Wang and R. J. Hyndman, "Online conformal inference for multi-step time series forecasting," 2024, *arXiv:2410.13115*.

[32] S. C. Hoi, D. Sahoo, J. Lu, and P. Zhao, "Online learning: A comprehensive survey," *NeuroComputing*, vol. 459, pp. 249–289, 2021.

[33] Y. Emek, R. Lavi, R. Niazadeh, and Y. Shi, "Stateful posted pricing with vanishing regret via dynamic deterministic markov decision processes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 2970–2982.

[34] A. Saadallah, M. Tavakol, and K. Morik, "An actor-critic ensemble aggregation model for time-series forecasting," in *Proc. Int. Conf. Data Eng.*, 2021, pp. 2255–2260.

[35] A. Kalai and S. Vempala, "Efficient algorithms for online decision problems," *J. Comput. System Sci.*, vol. 71, no. 3, pp. 291–307, 2005.

[36] D. Zhang, H. Zhang, A. Courville, Y. Bengio, P. Ravikumar, and A. S. Suggala, "Building robust ensembles via margin boosting," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 26669–26692.

[37] N. Alon, M. Bun, R. Livni, M. Malliaris, and S. Moran, "Private and online learnability are equivalent," *ACM J. ACM*, vol. 69, no. 4, pp. 1–34, 2022.

[38] A. Suggala and P. Netrapalli, "Follow the perturbed leader: Optimism and fast parallel algorithms for smooth minimax games," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 22316–22326.

[39] A. S. Suggala and P. Netrapalli, "Online non-convex learning: Following the perturbed leader is optimal," in *Proc. 31st Int. Conf. Algorithmic Learn. Theory*, 2020, pp. 845–861.
[40] V. Vovk, A. Gammerman, and G. Shafer, *Algorithmic Learning in a Random World*. Berlin, Germany: Springer, 2005.
[41] V. Vovk, J. Shen, V. Manokhin, and M.-G. Xie, "Nonparametric predictive distributions based on conformal prediction," *Proc. Mach. Learn. Res.*, vol. 60, pp. 82–102, 2017.
[42] M. Seeger, "Gaussian processes for machine learning," *Int. J. Neural Syst.*, vol. 14, no. 02, pp. 69–106, 2004.
[43] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," in *Proc. Adv. Neural Inf. Process. Syst.*, 1996, pp. 155–161.
[44] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Mach. Learn.*, vol. 63, no. 1, pp. 3–42, 2006.
[45] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Oxfordshire, U.K.: Routledge, 2017.
[46] J. L. Elman, "Finding structure in time," *Cogn. Sci.*, vol. 14, no. 2, pp. 179–211, 1990.
[47] T. S. Talagala, R. J. Hyndman, and G. Athanasopoulos, "Meta-learning how to forecast time series," *J. Forecasting*, vol. 42, no. 6, pp. 1476–1501, 2023.
[48] P. Montero-Manso, G. Athanasopoulos, R. J. Hyndman, and T. S. Talagala, "FFORMA: Feature-based forecast model averaging," *Int. J. Forecasting*, vol. 36, no. 1, pp. 86–92, 2020.
[49] G. Ke et al., "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3146–3154.
[50] J. Kivinen and M. K. Warmuth, "Exponentiated gradient versus gradient descent for linear predictors," *Inf. Computation*, vol. 132, no. 1, pp. 1–63, 1997.

Jiadong Chen received the bachelor's degree in computer science and technology from Shanghai Jiao Tong University, in 2020. He is currently working toward the PhD degree with the Department of Computer Science and Engineering, Shanghai Jiao Tong University under the supervision of Prof. Guihai Chen and Prof. Xiaofeng Gao. He works as a research Intern with Bytebrain group led by Tieying Zhang. His research interests include stream data mining and cloud resource allocation.

Yang Luo received the BS degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China, in 2022. His research interests include time series forecasting and cloud service auto scaling. He is currently working toward the PhD degree with Shanghai Jiao Tong University, China. He interned as an algorithm engineer with Ant Group and ByteDance, specializing in data mining and cloud resource scaling. He has published several papers in high-ranked international conferences such as SIGKDD, ICDM, CIKM, ICSOC, etc.

Xiuqi Huang received the BE degree in computer science and technology from the Harbin Institute of Technology, in 2019, and the PhD degree in computer science and technology from Shanghai Jiao Tong University, advised by Prof. Guihai Chen and Prof. Xiaofeng Gao. She is a research assistant professor with the State Key Lab of CAD&CG, Zhejiang University, working under the supervision of Prof. Wei Chen. Her research interests include data management and data services with machine learning technologies. She participated in this work during her tenure as a Ph.D. Student with Shanghai Jiao Tong University.

Fuxin Jiang received the BE degree in mathematics and applied mathematics from the Huazhong University of Science and Technology, and the PhD degree from the Academy of Mathematics and Systems Science, Chinese Academy of Sciences, advised by Prof. Shouyang Wang. He is a research scientist with Bytedance. His research interests include workload forecasting in cloud and resource scheduling in cloud.

Yangguang Shi received the PhD degree from the Institute of Computing Technology, Chinese Academy of Sciences, in 2016. He is a professor with the School of Computer Science, Shandong University, where he also serves as a Ph.D. advisor. He is a Qilu Young Scholar and a recipient of funding from the Shandong Excellent Young Scientists (Overseas) Program. He then worked as a postdoctoral researcher with the Technion – Israel Institute of Technology. His research interests include the intersection of algorithm design and analysis, game theory, and learning theory. His work has been published in leading conferences such as STOC, ESA, NeurIPS, ITCS, ICDE, KDD, and INFOCOM, as well as in prestigious journals including JACM, TCS, and MOR. He has served as a Program Committee member for international conferences such as TAMC, COCOON, and WINE, and as a reviewer for conferences and journals including EC, TEAC, and MOR.

Tieying Zhang is a research scientist with Bytedance US. Before joining Bytedance, he was a research scientist and manager with Alibaba DAMO Academy. Before that he was a postdoc researcher with the Computer Science Department (Database Group), Carnegie Mellon University, working with Prof. Andy Pavlo and Anthony Tomasic, on AI-powered database systems. Prior to coming to CMU, he was an assistant professor with the Chinese Academy of Sciences. His research interest includes AI for systems and systems for AI. He is particularly interested in providing practical implementations that are deployable in the real world with strong theoretical foundations. He is the regular PC member of USENIX ATC, VLDB, ICDE, FSE etc.

Xiaofeng Gao (Senior Member, IEEE) received the BS degree in information and computational science from Nankai University, China, in 2004, the MS degree in operations research and control theory from Tsinghua University, China, in 2006, and the PhD degree in computer science from the University of Texas at Dallas, USA, in 2010. She is currently a professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. Her research interests include wireless communications, data engineering, and combinatorial optimizations. She has published more than 200 peer-reviewed papers in the related area, including in well-archived international journals such as *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Mobile Computing*, *IEEE/ACM Transactions on Networking*, *IEEE Transactions on Computers*, *IEEE Journal on Selected Areas in Communications*, and also in well-known conference proceedings such as SIGKDD, ICDE, WWW, NeurIPS, INFOCOM. Her research interests include data engineering and network optimization. She has served on the editorial board of Discrete Mathematics, Algorithms, and Applications. She is an ACM senior member and CCF distinguished member.