Robust Auto-Scaling with Probabilistic Workload Forecasting for Cloud Databases

Haitian Hang*, Xiu Tang§, Jianling Sun*, Lingfeng Bao*, David Lo†, Haoye Wang‡
* College of Computer Science and Technology, Zhejiang University, China
§ School of Software Technology, Zhejiang University, China
† School of Computing and Information Systems, Singapore Management University, Singapore
‡ School of Computer and Computing Science, Hangzhou City University, China
{hanght, tangxiu, sunjl, lingfengbao}@zju.edu.cn, davidlo@smu.edu.sg, wanghaoye@hzcu.edu.cn

2024 IEEE 40th International Conference on Data Engineering (ICDE) | 979-8-3503-1715-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICDE60146.2024.00308

***Abstract*—Auto-scaling is crucial for achieving elasticity in cloud databases as well as other cloud systems. Predictive auto-scaling, which leverages forecasting techniques to adjust resources based on predicted workload, has been widely adopted. However, the inherent inaccuracy of forecasting presents a significant challenge, potentially causing resource under-provisioning.**

To address this challenge, we propose robust predictive auto-scaling that considers the uncertainty in forecasts. Unlike previous predictive approaches that rely on single-valued forecasts, we leverage probabilistic forecasting techniques to generate quantile forecasts, providing a more comprehensive understanding of the potential future workloads. By formulating the auto-scaling problem as a robust optimization problem, we enable the implementation of auto-scaling strategies with customizable levels of robustness, which can be determined by considering various quantile levels of forecasts. Moreover, we enhance the adaptability of our strategy by incorporating different quantile levels throughout the entire decision horizon, allowing for dynamic adjustments in the conservatism of our auto-scaling decisions. This enables us to strike a balance between resource efficiency and system robustness. Through extensive experiments, we demonstrate the effectiveness of our approach in achieving robust auto-scaling in cloud databases, while maintaining reasonable resource efficiency.

***Index Terms*—Resource Scaling, Workload Forecasting, Cloud Databases**

I. Introduction

Cloud databases have become increasingly popular due to their scalability, availability, and cost-effectiveness [1]–[4]. Auto-scaling is a crucial technique for achieving elasticity in cloud databases, enabling them to dynamically adjust their computing resources in response to changing workloads [5], [5]–[12]. Predictive auto-scaling, which leverages forecasting techniques to adjust resources based on predicted workloads, has been widely adopted in practice [8]–[15].

Motivations. Accurate forecasting is crucial in predictive auto-scaling to ensure adequate resource allocation for handling future workloads and cost optimization. Existing methods generate *point forecasts* for future workloads, which provide the central tendency or the most probable outcome of the forecast [11], [15]–[17], and then the allocation of database resources can be adjusted accordingly. However, this approach fails to consider the *uncertainty* in forecasts, which stems from their inherent lack of complete accuracy or precision, consequently leading to suboptimal decision-making.

Jianling Sun is the corresponding author.

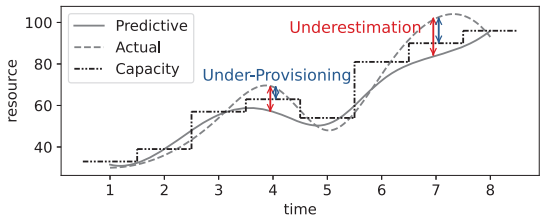


Fig. 1. **Robustness issue in predictive auto-scaling** - The red arrow signifies the extent of workload underestimation relative to the actual workload. The blue arrow represents the resource allocation gap between the allocated resources and the actual resource demand for workloads.

Figure 1 serves as an example that highlights the robustness issue that emerges when relying solely on point forecasts. More specifically, the future workload is forecasted using point estimates, and resource capacity planning is carried out accordingly. However, if the future workload is underestimated, relying solely on the point forecast can lead to inadequate resource provisioning. Consequently, it may lead to performance degradation and reduced service quality. While over-provisioning by allocating resources based on a higher value than the actual forecasted value can mitigate the risk of under-provisioning, it lacks a theoretical framework or uncertainty measure to guide the extent of over-provisioning. Consequently, suboptimal levels of redundancy may occur, resulting in inefficient utilization of resources.

Challenges. The example effectively illustrates the primary challenges in current predictive auto-scaling problems. ❶ *How to estimate the uncertainty in workload forecasts.* (**C1**). To comprehend the extent of underestimation is crucial for identifying potential risks of resource scarcity and enabling proactive mitigation strategies. Having a comprehensive range of potential future workloads helps in understanding the possibilities of underestimation and the potential risks involved in relying on those predictions. ❷ *How to incorporate forecast uncertainty into resource scaling to enhance the robustness of strategies.* (**C2**). For instance, allocating additional resources proportionally to the estimated underestimation mitigates under-provisioning, thus reducing the risk of service disruptions. Hence, a more robust predictive auto-scaling ap-

2375-026X/24/$31.00 ©2024 IEEE
DOI 10.1109/ICDE60146.2024.00308

proach is required to offer a comprehensive perspective on forecasts, without assuming the existence of a single accurate prediction value, and to appropriately scale resources.

Our Approach. To address these challenges mentioned above, there is a research gap in developing comprehensive approaches that not only quantify the inherent uncertainty in workload forecasting but also explicitly account for uncertainty in the resource scaling process. We propose a novel approach called *robust predictive auto-scaling*. To tackle the **C1** concern, we utilize probabilistic forecasting models to generate quantile forecasts. Unlike prior approaches that rely on point forecasts, quantile forecasts offer a spectrum of potential future outcomes along with their associated probabilities, thereby facilitating uncertainty quantification. Subsequently, to address the **C2** concern, we reformulate the auto-scaling problem as a robust optimization problem. This formulation explicitly incorporates the uncertainty of forecasts by utilizing quantile forecasts generated by the probabilistic models. Moreover, we extend our robust auto-scaling strategy to enable adjustment of policy conservatism based on varying conditions, without compromising robustness. This adaptability facilitates additional enhancements in resource utilization while maintaining the same level of robustness.

Contributions. In our study, we concentrate on the issue of horizontal scaling, known as scale-out, in disaggregated cloud databases. Within disaggregated architectures, horizontal scaling offers the benefit of uninterrupted service during scaling operations and involves minimal scaling overhead. These advantages enable us to prioritize workload forecasting and the associated scaling operations. To summarize, we make the following contributions:

- In the domain of workload forecasting, we introduce probabilistic forecasting techniques to predict quantiles, broadening the scope of existing studies on database workload forecasting.
- We seamlessly integrate probabilistic workload forecasting into the realm of cloud database auto-scaling. Our work addresses the challenge through a robust formulation, introducing predictive auto-scaling strategies that explicitly consider forecast uncertainty. This integration not only mitigates the risks of under-provisioning but also optimizes resource utilization efficiency.
- We empirically verify the effectiveness and superiority of our proposed methods using real-world workload data. The results from two distinct datasets showcase significant improvements in terms of both robustness and resource utilization efficiency.

II. Background

A. Auto-scaling in Cloud Databases

Auto-scaling has undergone extensive study within distributed and cloud databases, as well as in other cloud-based systems [13]–[15], [18]–[20].

Reactive & Proactive. When considering the timing of scaling operations, auto-scaling can be broadly classified into two primary types: *proactive scaling* and *reactive scaling*. The reactive one entails resource adjustment based on current demand, potentially resulting in delayed responses to changes in demand [5], [21]. In contrast, proactive scaling takes a forward-looking approach by adjusting resources in anticipation of potential workload changes [8]–[11], [13]–[15]. Typically, it relies on forecasting techniques to predict future workloads and allocate resources accordingly.

Disaggregated Database Systems. In the era of cloud computing, disaggregated architectures, characterized by resource disaggregation and pooling, have become the standard implementation for deploying cloud databases [1], [2]. These architectures provide independent scaling of various resources, such as compute and memory. This design philosophy fosters extreme elasticity, prompting a reevaluation of auto-scaling.

For instance, in the case of scaling out a distributed database with a shared-nothing architecture, the overhead of data migration can be significant [9]. However, in cloud databases with disaggregated storage, the cost of scaling out with new computing nodes is primarily associated with loading in-memory components. This process typically takes a very short amount of time, often just a matter of seconds.

As a result, our primary focus on scaling out in cloud databases allows us to prioritize demand forecasting and the corresponding scaling operations without the need to consider service interruptions or the scheduling of scaling activities.

B. Probabilistic Forecasting

Time series forecasting has become increasingly prevalent in recent years, particularly in large-scale industrial applications. For example, workload forecasting is a critical step towards an autonomous database system [16], [22]. Typically, in these applications of time series forecasting, the forecast is a single value that represents the most likely outcome.

In contrast to point forecasting, probabilistic forecasting is a powerful technique that facilitates the estimation of probability distributions for future outcomes, offering a comprehensive understanding of future prediction [23]. Specifically, probabilistic forecasting can represent the probability distribution using various mathematical functions, including the probability density function (PDF), cumulative density function (CDF), and quantile function [24].

Probabilistic forecasting has demonstrated superior performance to point forecasting in numerous domains [25], primarily due to its ability to improve decision-making by considering the uncertainties associated with different outcomes. This advantage is particularly evident in risk management-driven fields like finance [26], highlighting its potential to significantly enhance prediction-based optimization tasks in the context of database management systems.

III. Robust Predictive Auto-Scaling

A. Overview

Workflow. Our proposed robust predictive auto-scaling is a framework designed to address the issue of resource elasticity in cloud databases by offering automated scaling capabilities.

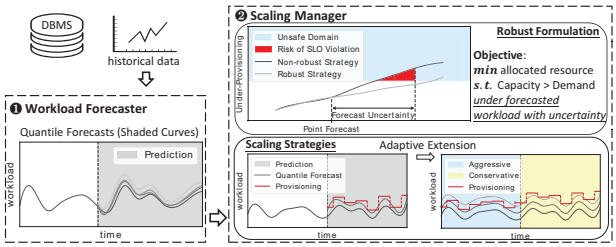


Fig. 2. **Overview of Our Approach** - The quantile forecasts generated by workload forecaster guides resource provisioning, and the selection of different quantile levels determines the conservatism of strategies.

As illustrated in Figure 2, the workflow is comprised of two phases, ❶ *Probabilistic Workload Forecaster* and ❷ *Robust Auto-Scaling Manager*. To begin, historical workload traces are used to train the workload forecaster. The forecaster employs probabilistic forecasting models to provide quantile forecasts, which offer a range of potential outcomes, capturing different levels of risk. The quantile forecasts generated by the forecaster are then fed into the Robust Auto-Scaling Manager, which formulates auto-scaling as a robust optimization problem, thus offering a robust resource scaling solution.

Probabilistic Workload Forecaster. One critical aspect of our framework is its ability to predict the workload using time series forecasting techniques. It relies on probabilistic forecasting models to generate quantile forecasts, which capture the uncertainty in the forecast. This research explores two distinct methodological categories of probabilistic forecasting models, which differ in their model output, i.e., parametric distribution and specified quantile levels. By utilizing a learned distribution or quantile levels, desired quantile forecasts can be generated.

Robust Auto-Scaling Manager. Our proposed Robust Auto-Scaling Manager is responsible for automating the scaling of the cloud database based on the quantile forecasts generated by the Probabilistic Workload Forecaster. Its main objective is to ensure robustness in resource scaling by incorporating quantile forecasts into the decision-making process. This is achieved by formulating auto-scaling as a robust optimization problem, where the aim is to minimize resource costs while avoiding resource under-provisioning within the range of potential workloads. Furthermore, we bolster the adaptability of our strategy by integrating various quantile levels across the entire decision horizon, enabling dynamic adjustments in the conservatism of our auto-scaling decisions.

B. Probabilistic Workload Forecaster

1) Predicting Quantiles instead of Single Values: Database workload forecasting is the process of predicting the future workload of a database system. A database workload can be characterized by various aspects, such as the arrival rate of queries (i.e., the number of submitted queries) and resource utilization (e.g., CPU usage), based on downstream tasks. From the perspective of resource scaling for cloud databases, resource utilization is a crucial factor and the most frequently employed metric for scaling, both in academic research and industrial applications [5], [11], [27]. Here, We provide a general definition of the workload forecasting task which does not specify the metric for characterizing the workload.

Definition 1 (Workload Forecasting)**.** *Given a historical workload time series* $\mathbf{w} = \{w_1, \ldots, w_T\}$ *where* w_t *is the workload at time* t *and* T *is the context length (i.e., the number of time steps used as input), workload forecasting is to predict the future workload time series* $\{\hat{w}_{T+1}, \ldots, \hat{w}_{T+H}\}$*, where* H *is the forecast horizon (i.e., the number of future time steps for which a forecast is generated).*

In previous studies, the workload forecasting definition provided earlier defines the predicted workload time series $\{\hat{w}_{T+1}, \ldots, \hat{w}_{T+H}\}$ as a sequence of individual workload values at each time step, thereby reflecting a point forecast. However, in practical scenarios, workload data frequently exhibits notable variations and outliers, which can result in inaccuracies within point forecasts.

Building upon the limitations of traditional point forecasting methods, we introduce the concept of quantile workload forecasting which is presented in Definition 2. Quantile workload forecasting provides a more comprehensive approach by incorporating the uncertainty associated with workload predictions. By estimating different quantiles of the workload distribution, such as the 10th, 50th (median), and 90th percentiles, quantile workload forecasting offers a range of potential future outcomes and their associated probabilities.

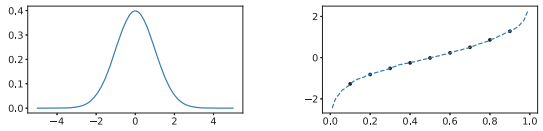
Definition 2 (Quantile Workload Forecasting)**.** *Given a historical workload time series* $\mathbf{w} = \{w_1, \ldots, w_T\}$*, the objective is to forecast the future workload at a prespecified quantile level (or quantile levels)* τ*, denoted by* $\left\{\hat{w}^{\tau}_{T+1}, \ldots, \hat{w}^{\tau}_{T+H}\right\}$.

Quantile workload forecasting addresses the shortcomings of point forecasts by capturing the complexity and variability inherent in workload dynamics. Unlike single-valued predictions, quantile forecasts provide decision-makers with a more nuanced understanding of the potential workload scenarios and the corresponding likelihoods. This allows for a better assessment of workload variations, outliers, and unexpected events, enabling informed decisions regarding resource allocation, capacity planning, and performance optimization.

2) Predicting Quantiles using Probabilistic Forecasting: For the implementation of quantile workload forecasting, advanced modeling techniques such as quantile regression can be utilized [28]. In our approach, we utilize learning-based probabilistic forecasting models rather than traditional statistical models. By harnessing the capabilities of deep neural networks to capture intricate patterns and dependencies in workload data, neural time series forecasters have demonstrated superiority over traditional statistical models in workload forecasting, as supported by prior studies [16], [17].

In this section, we explore two primary methodologies of probabilistic forecasting, categorized based on the model output and corresponding loss functions.

Learn parametric distributions. One of the most frequently used methods for representing probability distributions in forecasting is through the probability density function (Fig-



(a) Parametric distribution (b) Pre-specified grid of quantiles

Fig. 3. Two primary methodologies of probabilistic forecasting categorized based on the model output.

ure 3(a)). This approach assumes that the distribution of possible values conforms to a particular parametric distribution, such as the Gaussian distribution. The model subsequently produces distribution parameters, such as the mean and variance, which can be used to reconstruct the distribution.

As an illustration, consider a simple feed forward neural network. Its output layer can generate the mean (μ_t) and variance (σ_t) of a Gaussian distribution. A softplus function is commonly used to map the corresponding parameter and ensure that σ_t is non-negative. The negative log-likelihood (NLL) is a suitable choice for the loss function, as it enables direct computation of the likelihood of a given point under a probability density function. However, for different types of forecasting problems, various differentiable parametric distributions can be used in addition to the Gaussian likelihood. For instance, the Student-t distribution is one such distribution that is commonly used. In our implementation of a parameterized distribution-based forecaster, we chose the Student-t distribution because it has longer tails and a larger variance, allowing it to better handle outliers and noise.

With a parameterized distribution, we can now generate possible forecasts at a desired quantile level, using sampling methods. The accuracy of the estimation using sampling methods depends on the sample size, with larger sample sizes leading to more precise estimates. For certain distributions, specialized sampling techniques may be required to obtain accurate estimates.

Learn Pre-specified Grid of Quantiles. This technique directly outputs the values of specific quantiles, which can be of practical interest for directly providing desired lower or upper bounds of workload forecasting (Figure 3(b)).

To train the model, the quantile loss function is utilized to penalize the difference between the predicted quantile y and the actual values $\hat{y}$. The quantile loss (QL) function is expressed as:

$$\rho_\tau(y, \hat{y}) = (\tau - \mathbb{I}(y < \hat{y})) \cdot (\hat{y} - y), \tag{1}$$

where $\mathbb{I}(y < \hat{y})$ is the indicator function that equals 1 if $y < \hat{y}$ and 0 otherwise. This indicator function ensures that the loss is calculated differently for underestimation and overestimation cases, based on the given quantile level τ. When $\tau = 0.5$, it is equivalent to the Mean Absolute Error, and its minimizer is the median of the predictive distribution.

For multi-horizon, multivariate forecasting, the model is trained to minimize the total loss, which is a sum of losses over all forecast horizons and time series being predicted. The total loss of a prespecified quantile τ is defined as:

$$QL_\tau = \sum_{h=1}^{H} \sum_{i=1}^{n} \rho_\tau(y_{t+h,i} - \hat{y}_{t+h,i}), \tag{2}$$

where n is the number of time series being forecasted, H is the forecast horizon. The quantile loss function ρ_τ is defined as in the single-step, single-variable case, but is now applied over all-time series and forecast horizons. Depending on the problem, different weights can be assigned to components of the sum to or discount different quantiles and horizons.

Pros, Cons & Selection Criteria. Although both methods are capable of generating quantile forecasts, each has its own strengths and limitations. Learning parametric distributions directly yields a probability distribution without explicitly predicting quantiles. By sampling from the learned distribution, it allows for greater flexibility in selecting the desired quantile levels. Consequently, it enables more adaptable conservatism adjustment when applied to downstream tasks. However, it is essential to note that this method is sensitive to the correct specification of the parametric form, which can be challenging when data patterns are complex or non-standard.

In contrast, learning a pre-specified grid of quantiles involves selecting a set of quantiles of interest in advance so that it allows for a targeted estimation of the desired quantiles without making assumptions about the distribution. Therefore, it generally achieves higher accuracy in quantile forecasts within the same model. Nevertheless, since the quantile levels are pre-determined, retraining the model becomes necessary when different quantile levels are desired.

In summary, when choosing between these methods, careful consideration of the data characteristics and the desired level of flexibility in quantile forecasts is crucial.

Note that our paper does not extensively explore the suitability of models for various workload patterns, such as long-term versus short-term forecasting or handling specific cyclic and spike patterns. The appropriateness of models for these scenarios is typically determined by the architectural choices, including options like Multilayer Perceptron (MLP) or Recurrent Neural Network (RNN). However, given the primary focus and scope of our research, which revolves around the integration of quantile workload forecasting, we do not undertake an extensive analysis of model limitations and architectural aspects. Instead, our paper places a stronger emphasis on discussions related to the model's ability to provide quantile forecasts, including its sensitivity to the correct specification of the parametric form.

Concerning the models and their corresponding architectures we leverage, our paper aligns with the methodologies outlined above. We employ and evaluate two representative standard models for each approach: DeepAR which learns parametric distributions [29], and Temporal Fusion Transformer which learns a pre-specified grid of quantiles [30], as elaborated in Section IV-A. These neural models offer the flexibility to generate diverse model outputs and optimize various loss functions. For instance, an MLP can be trained

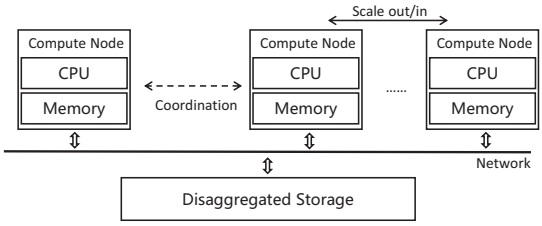


Fig. 4. **Horizontal Scaling in Storage Disaggregated Databases** - Scaling can be performed at the resource level without affecting the entire system.

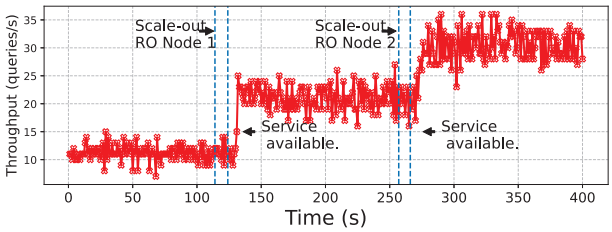


Fig. 5. **Scale-out Overhead** - It only takes a few seconds to scale out, i.e., to build in-memory components from the checkpoints.

to output distribution parameters or predict specific quantiles. To ensure an objective evaluation of the model's performance, the choice to either learn parametric distributions or a pre-specified grid of quantiles is in accordance with the initial design of each respective method.

C. Robust Auto-Scaling Manager

1) Primer on Robust Auto-Scaling: In this paper, our primary focus centers on the challenge of horizontal scaling within cloud databases that adopt disaggregated architectures, such as storage disaggregation [1] and memory disaggregation [2]. In contrast to traditional distributed shared-nothing architectures, disaggregated architectures facilitate resource pooling, thereby supporting extreme elasticity, as exemplified in Figure 4. Furthermore, as opposed to vertical scaling, which necessitates considerations regarding downtime duration and the scheduling of scale operations at suitable intervals, the scale-out approach allows us to place a stronger emphasis on demand forecasting and the associated resource provisioning.

Specifically, the horizontal scaling problem in cloud databases with disaggregated architectures can be formulated as an optimization problem that configures the number of compute nodes for each time as follows:

Definition 3 (Auto-Scaling Optimization)**.** *For a predicted workload time series* $\mathbf{w} = \{w_1, \ldots, w_H\}$ *where* w_t *is the workload at time* t*, an auto-scaling strategy involves allocating a certain number of compute nodes to guarantee that the average workload of these nodes remains below a predefined threshold at each time* t *all while minimizing the total number of compute nodes required, i.e.,*

$$\min \sum_{t=1}^{H} c_t, \qquad s.t.\ \frac{w_t}{c_t} \leq \theta_t, \tag{3}$$

where c_t *is the number of compute nodes allocated at time* t *and* θ_t *are predefined thresholds according to the metric for scaling (e.g., Percentage CPU).*

In our problem formulation, we intentionally omit consideration of scaling overhead for several reasons. In disaggregated architectures, scaling overhead is less prominent due to the independent scaling of resources. As illustrated in Figure 5, scale-out operations typically have short warm-up periods lasting only a few seconds[1]. In practice, scale-out operations are infrequent and typically have intervals of several tens of minutes to even several hours, rendering the warm-up time, which is on the order of seconds, negligible.

Secondly, our research primarily concentrates on optimizing resource utilization, specifically ensuring that average resource usage remains within predefined thresholds. The cost of scaling, including aspects like warm-up time, cannot be directly integrated into our optimization objective. Additionally, These costs are closely tied to system design and operational considerations, which can substantially differ across various implementations and environments. Therefore, attempting to model scaling costs would introduce unnecessary complexity and hinder the development of a generalized framework.

Furthermore, in scenarios with frequent scale-out operations, occurring perhaps every few minutes, we can tackle this by introducing constraints which restrict the frequency and timing of scale-out actions into our problem formulation, effectively managing the impact of scaling overhead. The content directly related to this issue and the corresponding solutions is elaborated upon in Section V-A.

In light of the probabilistic workload forecaster introduced in Section III-B, which provides a set of potential workloads, we now propose the definition of the robust auto-scaling optimization problem.

Definition 4 (Robust Auto-Scaling Optimization)**.** *Given the predicted workloads with uncertainty at time* t *represented by* $\hat{\mathcal{W}}_t$*, encompassing a range of potential values, a robust auto-scaling scheme aims to minimize the total number of compute nodes allocated while ensuring that, for each time* t*, the average workload of each node remains below a specific threshold, even in the presence of uncertainty:*

$$\min \sum_{t=1}^{H} c_t, \qquad s.t.\ \forall w_t \in \hat{\mathcal{W}}_t, \frac{w_t}{c_t} \leq \theta_t. \tag{4}$$

The comparison between Definition 3 and Definition 4 reveals that the primary differentiation between a robust auto-scaling strategy and a non-robust one hinges on the consideration of forecast uncertainty. Through the incorporation of the uncertainty $\mathcal{W}_t$ that outlines possible workload values, the robust approach ensures sufficient resources are allocated to handle future workloads, even with imprecise forecasts.

To solve Equation 4, we can reformulate it by its robust

[1]Data for Figure 5 is provided by Alibaba Cloud Database Service.

counterpart as follows:

$$\min \sum_{t=1}^{H} c_t, \quad s.t. \ \frac{\sup_{w_t \in \hat{\mathcal{W}}_t} w_t}{c_t} \leq \theta_t. \tag{5}$$

In the above equation, the supremum operator (sup) represents the maximum value taken overall possible workload scenarios within the uncertainty set $\hat{\mathcal{W}}_t$. It denotes the worst-case scenario where the workload takes on its maximum value within the uncertainty range. By considering this worst-case scenario, the strategy ensures that the allocated resources can handle the workload under any possible condition within the uncertainty set, thus preventing resource under-provisioning.

The probabilistic workload forecaster introduced in Section III-B enables us to obtain a set of specified quantiles of interest. Utilizing these quantile forecasts, we can determine an upper bound on the workload by selecting a specific quantile forecast at a chosen level. For instance, we can consider the 0.9 quantile forecast as a reference point for resource allocation. To achieve the desired level of conservatism, we can replace the supremum operator with a specific quantile, leading to the following formulation:

$$\min \sum_{t=1}^{H} c_t, \quad s.t. \ \frac{\hat{w}_t^{\tau}}{c_t} \leq \theta_t, \tag{6}$$

where $\hat{w}_t^{\tau}$ denotes the τ quantile forecast provided by the probabilistic forecaster for time period t, e.g., 0.9 quantile.

The resulting optimization problem is deterministic and can be solved using standard linear programming solvers. The quantile τ adopted in Equation 6 controls the level of conservatism an auto-scaling strategy realizes. In practice, it is common to select quantile forecasts with a quantile level greater than 0.5 to ensure robust resource scaling. By choosing a higher quantile level, we increase the conservativeness of the forecasts, making the optimization problem more robust against resource under-provisioning. Conversely, selecting a lower quantile level introduces a more aggressive approach, reducing under-utilization compared to the conservative ones.

2) Adaptive Extension: In the basic robust auto-scaling strategy mentioned above, a single quantile forecast is applied uniformly across the entire decision horizon. For example, the strategy scales resources based on the 0.9 quantiles for each time step. However, in practical scenarios, it is common practice to generate predictions for multiple future time steps and formulate corresponding strategies. Using a fixed quantile for the entire auto-scaling strategy across all prediction horizons is inadequate for several reasons.

Firstly, relying on a fixed quantile does not ensure the desired balance between resource under-provisioning and resource under-utilization at every time interval. Allocating resources based on a conservative quantile may effectively prevent under-provisioning during certain periods when workload demands are higher than expected, but it can lead to substantial overestimation during others.

Secondly, such an auto-scaling strategy fails to fully exploit the valuable information provided by probabilistic forecasting. The forecasted distribution or quantiles not only provide insights into the range of possible values and their likelihood but also implicitly convey the level of confidence or uncertainty associated with the forecast. For instance, a wider distribution or higher quantiles in a probabilistic forecast indicate greater uncertainty and lower confidence in the predictions.

To overcome this limitation, we propose a *robust auto-scaling strategy with adaptive robustness* that allows for the adjustment of the level of conservatism based on varying quantiles for different time periods. We formally define this strategy as follows:

Definition 5 (Auto-Scaling Optimization with Adaptive Robustness)**.** *Auto-scaling with adaptive robustness is an optimization scheme that dynamically adjusts the allocation of compute nodes based on varying quantiles of predicted workloads across different time intervals:*

$$\min \sum_{t=1}^{H} c_t, \quad s.t. \ \frac{\hat{w}_t^{\tau_t}}{c_t} \leq \theta_t, \tag{7}$$

where τ_t indicates the quantile level considered at each time t, specifically.

The adaptive robust auto-scaling strategy, defined in Definition 5, incorporates the ability to choose varying quantile levels for resource scaling at each time step, differentiating it from the basic approach. This additional flexibility empowers the strategy with enhanced adaptability, facilitating a more refined equilibrium between robustness and resource utilization.

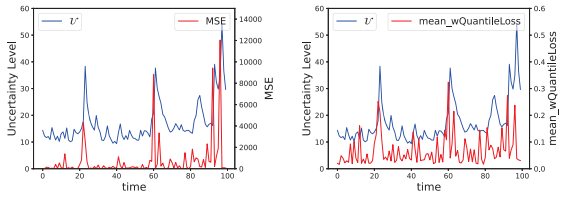
Nevertheless, a significant challenge persists in deciding *when to employ a conservative strategy versus an aggressive one*. To tackle this challenge, we introduce an *uncertainty-aware adaptive scaling* that tactically adjusts the level of conservatism based on the uncertainty of quantile forecasts.

As mentioned before, the forecasted distribution or quantiles provide valuable insights into the possible value range and probabilities, while implicitly indicating the level of confidence or uncertainty in the forecast. In this context, we introduce a metric to quantify uncertainty, which measures the extent of deviation of the forecasted values from the central tendency, such as the mean or median.

When dealing with probabilistic forecasting models that directly provide distribution parameters, the variance or standard deviation of the predicted distribution can be used as a direct measure of uncertainty. Alternatively, for models that estimate predetermined quantiles, we propose a metric to quantify it. More specifically, considering a single step of workload forecast at time i, comprising n quantile levels $\{\tau_1, \ldots, \tau_n\}$ along with the mean forecast (i.e., 0.5quantile), we define the level of uncertainty using the following expression:

$$\mathcal{U} = \sum_{i=1}^{n} (\tau_i - \mathbb{I}(w^{\tau_i} < w^{0.5})) \cdot (w^{0.5} - w^{\tau_i}), \tag{8}$$

with $\mathbb{I}(w^{\tau_i} < w^{0.5})$ being the indicator function that yields 1 when the workload forecast at the quantile level τ_i is lower than the mean forecast, and 0 otherwise.



(a) $\mathcal{U}$ & MSE (b) $\mathcal{U}$ & mean_wQuantileLoss

Fig. 6. The correlation between the level of uncertainty indicated by quantile forecasts and forecasting accuracy.

The proposed metric, denoted as $\mathcal{U}$, shares similarities with quantile loss in terms of calculation. However, the crucial difference is that it compares the forecast at each quantile level with the median forecast rather than the target value, thereby capturing the relative uncertainty of the forecasted results across all specified quantile levels. A higher value of the computed metric $\mathcal{U}$ signifies an elevated level of uncertainty linked to the forecasted results.

The correlation between the level of uncertainty in the forecasted results and their accuracy is further explored. To visually depict this relationship, Figure 6 showcases the metric $\mathcal{U}$ associated with quantile forecasts at each time period t within the sampled forecasting horizons. The Mean Square Error (MSE) of the mean forecast and the Mean Weighted Quantile Loss (mean_wQuantileLoss, an evaluation metric for quantile forecasts, detailed in Section IV) for all quantiles are also included in the visualization. The findings reveal a consistent trend between forecast uncertainty and accuracy within the sampled forecasting horizons. Higher levels of uncertainty at each time step are generally indicative of less accurate predictions, regardless of whether they are quantile forecasts or mean forecasts.

Uncertainty-aware Adaptive Scaling. Building upon the aforementioned observations, we now detail the uncertainty-aware adaptive approach that dynamically adjusts the level of conservatism in auto-scaling strategies based on the indicated uncertainty level in the forecasts. Algorithm 1 exemplifies our uncertainty-aware approach, featuring two optional quantile levels for adjustment. At each time step, the uncertainty level of the quantile forecasts is calculated using Equation 8. When the uncertainty level surpasses or equals the predefined threshold ρ_τ, compute nodes are allocated based on the more cautious quantile level's forecast. Conversely, if the uncertainty level falls below the threshold, compute nodes are allocated based on the more optimistic quantile level's forecast.

The uncertainty threshold ρ_τ plays a critical role in determining the conservatism of the scaling plan based on whether the uncertainty level exceeds or is below the threshold. In practical usage, it is advisable to select an appropriate uncertainty threshold based on the predictive model's performance. This selection process entails considering the relationship between the uncertainty level and the forecasting performance, which can be derived from historical data. Through this analysis, a suitable uncertainty threshold can be determined, striking a desired balance between accuracy and conservatism.

Algorithm 1 Uncertainty-aware Adaptive Scaling

Input: $\{\tau_1, \tau_2\}$: two optional quantile levels where $\tau_1 < \tau_2$; θ: workload threshold for resource scaling; ρ_τ: uncertainty threshold for adjusting quantile levels; $\{\hat{w}_1, \ldots, \hat{w}_n\}$: quantile workload forecasts for n steps

Output: $\{c_1, \ldots, c_n\}$: an allocation of compute nodes for n steps;

for $i = 1$ to n **do**
  Calculate $\mathcal{U}_i$ for quantile forecasts at step i
  if $\mathcal{U}_i < \rho_\tau$ **then**
    $c_i = \arg\min_{c_i} \left\{ \frac{\hat{w}_i^{\tau_1}}{c_i} : \frac{\hat{w}_i^{\tau_1}}{c_i} < \theta \right\}$
  else
    $c_i = \arg\min_{c_i} \left\{ \frac{\hat{w}_i^{\tau_2}}{c_i} : \frac{\hat{w}_i^{\tau_2}}{c_i} < \theta \right\}$
  end if
end for
Return $\{c_1, \ldots, c_n\}$

Furthermore, our adaptive approach extends beyond just two optional quantile levels. It offers the flexibility to include a staircase-like range of options, enabling more precise control over the auto-scaling strategy. This expanded range allows us to finely adjust resource allocation in accordance with specific requirements, resulting in a more customizable solution.

IV. Experimental Evaluation

A. Experimental Settings

1) Datasets and Workloads: We construct datasets for our experimental evaluation using two real-world datasets. For both datasets, we aggregate the data at 10-minute intervals.

- *Alibaba Cluster Trace*[2]. We construct a resource usage trace (including CPU, memory, and disk) by sampling a subset of machines and aggregating the resource usage.
- *Google Cluster Trace*[3] [31]. We construct a resource usage trace (including CPU and memory) by sampling a subset of tasks and aggregating the resource usage.

2) Compared Methods: In this study, we evaluate (1) workload forecasters and (2) auto-scaling strategies.

Workload Forecasters. For workload forecasting, we investigate several probabilistic forecasting models, ranging from statistic models to transformer-based methods. We implement all these compared methods using GluonTS, a Python package for probabilistic time series modeling [32] and determine their hyperparameters using Optuna, an automatic hyperparameter optimization software framework [33]. To ensure that the models are not overly sensitive to their hyperparameters, we fix the hyperparameters to be the same across the different prediction horizons for each type of model. Moreover, we set the learning rate to $1e^{-3}$ for all models.

Of particular interest are the DeepAR and TFT models, as they represent two main approaches for quantile workload

[2]https://github.com/alibaba/clusterdata
[3]https://github.com/google/cluster-data

forecasting mentioned before, learning parametric distributions and learning pre-specified grid of quantiles, respectively.

- *Autoregressive Integrated Moving Average (ARIMA)* [34]. A classic statistical time series model which combines autoregressive (AR) and moving average (MA). Quantile forecasts can be enabled by incorporating residuals to capture the uncertainty of the forecasts.
- *Multilayer Perceptron (MLP)* [35]. A simple feedforward neural network that generates probabilistic forecasts by outputting the parameters of a selected distribution.
- *DeepAR* [29]. A probabilistic forecasting model that learns a parametric distribution using autoregressive recurrent networks. It enables the computation of prediction quantiles by sampling in the predicted distribution.
- *Temporal Fusion Transformer (TFT)* [30]. An attention-based architecture that generates quantile outputs. It is trained by jointly minimizing the quantile loss, summed across all quantile outputs.

Resource Scalers. Regarding auto-scaling strategies, we compare our solution with reactive methods and predictive methods based on point forecasting. Reactive scalers, such as Google Autopilot [36] and Kubernetes default horizon pod auto-scaler (HPA) [21], are widely used in existing cloud systems. They employ a moving window approach to gather resource usage statistics over a recent period, which in turn informs the scaling of resources. In our experiments, we implement two reactive scalers that are conceptually similar to Google Autopilot.

- *Reactive-Max*. This scaler scales resources based on the maximum workload value within the window.
- *Reactive-Avg*. This scaler applies exponentially-decaying weights to the workload and performs scaling based on the weighted average. We set the half-life (decay factor) to 6, indicating that the weights decrease by half every 6 time intervals.

In the context of scalers based on point forecasting, we employ the following two models as workload forecasters.

- *QueryBot 5000 (QB5000)*. A hybrid forecasterthat combines linear regression, long short-term memory network, and kernel regression [16].
- *TFT-point*. We repurpose TFT, initially used as our quantile workload forecaster, as an independent model for point forecasting. More precisely, we train TFT to exclusively output the 0.5 quantile, effectively serving as a point forecasting model.

In the case of scalers based on point forecasting, we have also implemented an enhancement method described in [18] for comparison. This enhancement involves adding a small additional value to future predictions based on past underestimation errors. We refer to the combinations of this enhancement with the two point forecasting methods mentioned above as *QB5000-padding* and *TFT-point-padding*, respectively.

B. Evaluation of Forecasting Models

1) Forecasting Accuracy Evaluation: In this study, we compare the performance of various probability prediction models. The context length and prediction length are set to 12 hours, equivalent to 72 time steps. To evaluate the performance of probabilistic forecasting models in predicting quantiles, we utilize the following metrics:

- *Weighted Quantile Loss (wQL)*. It measures the accuracy of a model at a specified quantile. Given the quantile loss QL_τ at a specified quantile defined in Equation 2, we can derive the Weighted Quantile Loss (wQL) at a quantile level τ as follows:

$$wQL_{[\tau]} = \frac{2 \cdot QL_\tau}{\sum_{h=1}^{H} \sum_{i=1}^{n} y_{t+h,i}},$$

  where $y_{t+h,i}$ is the target value for the i-th time series at time $t+h$.
- *Coverage*. Another important metric we use for evaluating the predicted values at a quantile level. A coverage $Coverage_{[\tau]}$ measures the fraction of time series in the dataset for which the τ-percentile of the predictive distribution is larger than the true target. For a perfectly calibrated prediction, it holds that $Coverage_{[\tau]} = \tau$.
- *Mean Weighted Quantile Loss (mean_wQL)*. a global metric that measures the overall performance of the model in predicting the entire probability distribution which can be defined as:

$$mean\_wQL = \frac{1}{\mathcal{A}} \sum_{\tau \in \mathcal{A}} wQL_{[\tau]},$$

  where $\mathcal{A}$ is a set of prespecified quantile levels.

In this experiment, we test with $\mathcal{A} = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$. Given the necessity to employ quantile forecasts as upper bounds for guiding our robust auto-scaling strategy, special emphasis is placed on evaluating the performance of quantile predictions at specific levels, namely, the 0.7, 0.8, and 0.9 quantiles. Quantile loss and coverage are computed at their respective levels.

The results are presented in Table I. Based on these metrics, it is observed that both DeepAR and TFT outperformed the baseline methods, demonstrating their effectiveness in quantile forecasting. Particularly, TFT exhibited the highest performance among all the evaluated models.

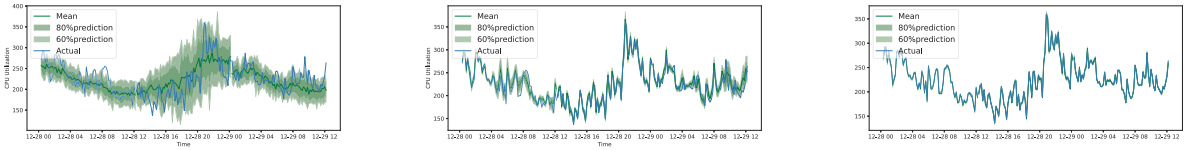
While our primary focus is on quantile forecasting and evaluating the associated metrics, we also assess the performance of the quantile forecasters in point forecasting using the Mean Squared Error (MSE) metric. To accomplish this, we derive the mean value from the forecast obtained at the predefined quantiles and utilize it as the point prediction. This supplementary analysis allows us to evaluate the accuracy of the models in predicting specific time points, complementing the overall assessment of their forecasting capabilities.

In addition, we visually present the prediction results of all models within a sampled forecasting horizon in Figure 7. DeepAR and TFT demonstrate superior performance compared to MLP in forecasting the possible range, as observed from the visualization of prediction intervals. DeepAR and TFT consistently maintain excellent coverage within narrow prediction intervals. The outstanding performance of DeepAR

TABLE I

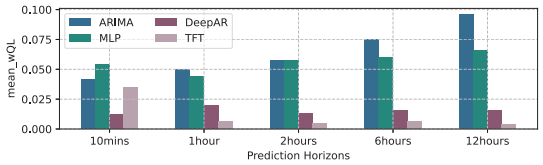
PERFORMANCE COMPARISON OF DIFFERENT MODELS WITH A CONTEXT LENGTH OF 72 STEPS AND PREDICTION LENGTH OF 72 STEPS. THE RESULTS ARE AVERAGED OVER 3 TRAINING RUNS.

Dataset	Model	Evaluation Metrics							
		mean_wQL	$wQL_{[0.7]}$	$wQL_{[0.8]}$	$wQL_{[0.9]}$	$Coverage_{[0.7]}$	$Coverage_{[0.8]}$	$Coverage_{[0.9]}$	MSE
Alibaba Trace	ARIMA	0.0320	0.0471	0.0314	0.0123	0.849	0.923	0.970	411.1
	MLP	0.0655	0.0777	0.0661	0.0457	0.676	0.765	0.857	869.1
	DeepAR	0.0153	0.0179	0.0153	0.0107	0.507	0.610	0.729	47.0
	TFT	0.0041	0.0045	0.0037	0.0023	0.647	0.779	0.908	3.1
Google Trace	ARIMA	0.4332	0.5358	0.4736	0.3487	0.737	0.810	0.884	5983.8
	MLP	0.3741	0.4708	0.4267	0.3257	0.620	0.705	0.806	5473.5
	DeepAR	0.0541	0.0599	0.0513	0.0380	0.6723	0.742	0.819	1068.7
	TFT	0.0163	0.0193	0.0171	0.0128	0.740	0.829	0.929	124.4

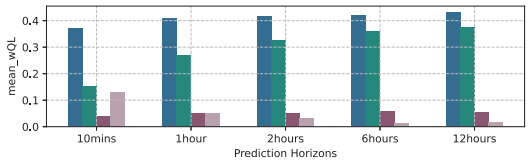


(a) MLP (b) DeepAR (c) TFT

Fig. 7. **Prediction Results** - A prediction interval is an estimate of the range within which a future observation is expected to fall. For instance, a 80% interval represents the range between the 0.1 quantile and the 0.9 quantile.



(a) Alibaba Trace



(b) Google Trace

Fig. 8. **Forecasting Horzions Evaluation** - Evaluation of forecasting models over different prediction horizons.

and TFT in forecasting narrow prediction intervals highlights their potential for more accurate and precise resource scaling. This capability is crucial in effectively addressing both under-provisioning and over-provisioning scenarios, as will be further substantiated through experimental analysis.

To summarize, the experimental results demonstrate that the proposed methods for quantile workload forecasting, DeepAR, and TFT, outperformed the traditional approaches (ARIMA and MLP) across multiple evaluation metrics. Notably, TFT exhibited exceptional performance in capturing the complete probabilistic distribution and predicting specific quantiles. Additionally, both DeepAR and TFT displayed competitive results in point forecasting, as evidenced by the MSE metric. These findings emphasize the potential of DeepAR and TFT in the realm of quantile workload forecasting.

2) Forecasting Horizons Evaluation: The performance of different models is experimentally evaluated across various prediction horizons. Specifically, with a fixed context length of 12 hours, the models' performance is assessed for prediction lengths of 10 minutes, 1 hour, 2 hours, 6 hours, and 12 hours, which correspond to 1, 6, 12, 36, and 72 time steps, respectively. As demonstrated in Figure 8, the DeepAR and TFT models consistently outperformed the baseline methods across all prediction horizons.

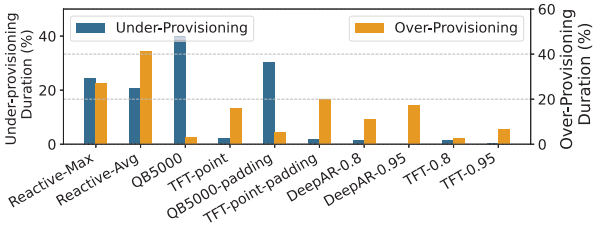
Additionally, we observe that the TFT model exhibits comparatively lower performance in short-term forecasting, such as one-step forecasting. This can be attributed to the fact that the model's hyperparameters were optimized for long-term prediction and applied to all prediction horizons, thereby not favoring short-term forecasting. Furthermore, DeepAR itself is designed as a one-step forecasting model that achieves multi-step forecasting iteratively. As a result, it demonstrates higher accuracy in short-term prediction due to the iterative approach, while its performance deteriorates as the prediction horizon increases due to the accumulation of iterative errors.

By analyzing the results, valuable insights can be gained regarding the patterns and trends in the models' performance across different prediction lengths. It aids in selecting the most suitable models for specific forecasting horizons.

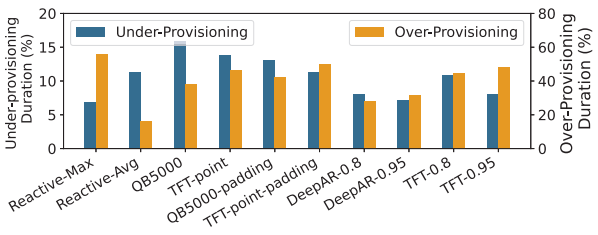
C. Evaluation of Auto-Scaling Strategies

In this subsection, we evaluate the performance of robust auto-scaling strategies based on probabilistic forecasting models through experimental analysis. We choose CPU utilization as the metric for scaling and make decisions for the future 72 time steps, i.e., 12 hours. Building upon the performance evaluation of probabilistic forecasting models discussed earlier, we select the DeepAR and TFT models as our quantile workload forecasters for further evaluation. Given our focus on the robustness of auto-scaling strategies in avoiding resource under-provisioning, we predominantly consider the forecasted values corresponding to quantile levels greater than 0.5 as the basis for our auto-scaling policies. Therefore, for models that learn pre-specific quantiles, we train them with forecasted quantile levels $\mathcal{A} = \{0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99\}$.

To evaluate the robustness in avoiding resource under-provisioning, we consider the following metric:

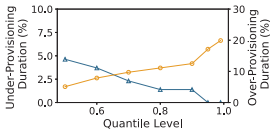


(a) Alibaba Trace

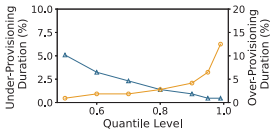


(b) Google Trace

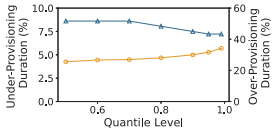
Fig. 9. **Under-Provisioning Rate Evaluation** - The number following the hyphen "-" in the model names indicates the quantile level used for auto-scaling guidance. TFT-point represents a TFT model trained for point forecasting, specifically targeting the 0.5 quantile.



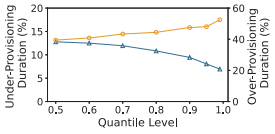
(a) DeepAR - Alibaba



(b) TFT - Alibaba



(c) DeepAR - Google



(d) TFT - Google

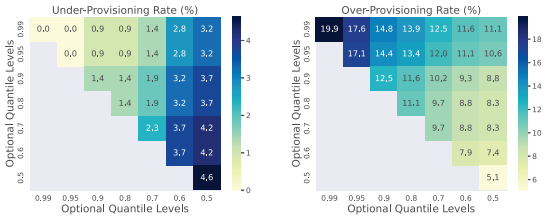
Fig. 10. **Analysis across Different Quantile Levels** - Evaluation of under-provisioning and over-provisioning rates when scaling resources based on forecasts at different quantile levels.

– *Under-Provisioning Rate*. It measures the percentage of periods in which the allocated resources exceed the actual workload demands.

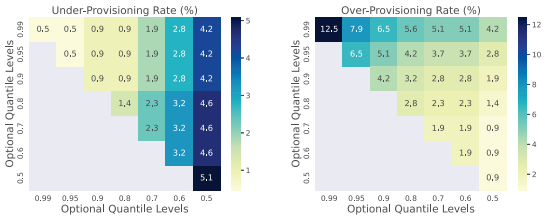
Due to the conservative estimation of future workload and the robust resource scaling strategies employed to avoid under-provisioning, there is a possibility of under-utilization caused by overestimating the workload. To evaluate these scenarios, we consider the following metric:

– *Over-Provisioning Rate*. It measures the percentage of time periods in which the allocated resources exceed the minimum amount required, reflecting the extent of resource under-utilization.

These metrics provide insights into the trade-off between avoiding resource under-provisioning and potential resource



(a) Alibaba Trace - DeepAR



(b) Alibaba Trace - TFT

Fig. 11. **Evaluation of Adaptive Approach** - These heatmaps illustrate the rates of under-provisioning and over-provisioning for various combinations of two optional quantile levels when coupled with the DeepAR and TFT model. The diagonal entries in the heatmap correspond to the basic method, which utilizes a fixed quantile level for the entire decision horizon.

under-utilization due to conservative workload estimation. By evaluating both metrics, we can assess the balance achieved by the auto-scaling strategies in effectively utilizing resources while avoiding under-provisioning.

1) Comparision with Existing Solutions: We compare our appproach with reactive ones and those utilizing point forecasts in addressing the challenge of under-provisioning. In our implementation, the reactive approach evaluated scales resources based on the average workload during the most recent n time steps, where n is set to 6.

The experimental results are presented in Figure 9. In the context of the TFT model, the label **TFT-point** refers to auto-scaling strategies based on point forecasting models trained using the TFT model. **DeepAR-**τ denotes the auto-scaling strategy that utilizes the forecasts based on the quantile level τ provided by the DeepAR-based quantile forecaster. For example, DeepAR-0.6 indicates the resource scaling strategy based on the 0.6 quantile forecasts generated by DeepAR. The same applies to **TFT-**τ.

The results depicted in Figure 9 highlights the consistent superiority of predictive auto-scaling strategies over reactive scalers. Even the reactive-max strategy, which scales resources based on the maximum observed value from the past, exhibits severe under-provisioning issues. This can be attributed to the inherent lag in reactive scaling.

We also investigate the benefits of incorporating quantile forecasts as opposed to relying exclusively on point forecasts. In our approach, when training a forecasting model like TFT, we develop both a point forecaster and a quantile forecaster. Our results demonstrate that auto-scaling strategies based on quantile forecasts, even when derived from models with lower long-term prediction accuracy like DeepAR, outperform those relying solely on point forecasts from more accurate models

like TFT. This underscores the significance of integrating quantile forecasts into the auto-scaling process.

Additionally, our approach outperforms predictive scalers based on padding enhancements. In our experiments, we observe that the likelihood and severity of underestimation in point forecasts for a specific time interval are not significantly correlated with past underestimation errors over a certain period. This is one of the reasons why scalers based on point forecasts with padding enhancements, while showing improvements compared to those exclusively based on point forecasts, still do not perform as well as our method.

Moreover, for the selected quantile levels, , our observations indicate that choosing more conservative quantile levels, such as higher quantiles, enhances the system's robustness against under-provisioning. By considering a wider range of potential workload variations, robust strategies based on more conservative quantile levels ensure that sufficient resource allocation.

There is typically a trade-off between conservative quantile levels and the occurrence of underover-provisioning. To gain a deeper understanding of the impact of different quantile levels, we conducted experiments to evaluate the performance of a robust scaler using various quantiles as presented in Figure 10.

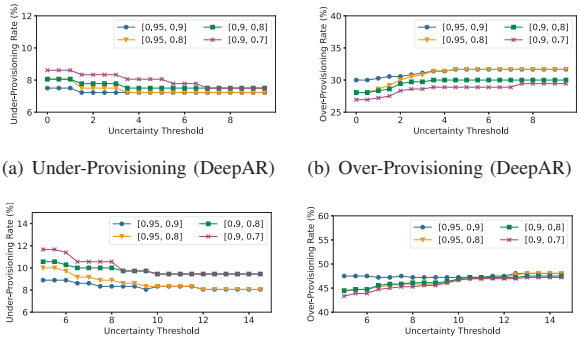
The analysis depicted in Figure 10 allows us to identify an optimal quantile level that strikes a balance between under-provisioning and over-provisioning. By evaluating a spectrum of quantile levels, we can determine the point where under-provisioning is effectively mitigated without incurring significant over-provisioning.

2) Evaluation of Adaptive Robust Approach: The aforementioned experiments indicate that it is crucial to select an appropriate quantile level where under-provisioning is effectively mitigated without incurring significant over-provisioning. In the case of multi-horizon decisions, we further perform a comparative evaluation between our adaptive method and the approach with a fixed quantile level.

Consistent with Algorithm 1, we set an uncertainty threshold and evaluated the performance for different combinations of quantile levels. Specifically, for the adaptive extension, we pre-select two quantile levels, and the conservatism of strategy within different time intervals depends on which of the two is chosen. By employing two optional quantile levels, we can demonstrate the adaptivity of our approach and enable clear and intuitive visualization of our experimental results. We evaluate different combinations of these two optional quantile levels. The results are depicted in Figure 11.

These heatmaps illustrate the rates of underover-provisioning for different combinations of two optional quantile levels. Our adaptive approach, which dynamically adjusts the quantile levels between the two pre-selected values, achieves a reduction in over-provisioning without increasing under-provisioning compared to the method with a fixed quantile level. This improvement in reducing over-provisioning leads to an enhancement in resource utilization efficiency while maintaining the robustness of auto-scaling.

Moreover, we perform a sensitivity analysis on the uncertainty threshold to investigate its impact on the performance of our adaptive approach. The results are presented in Figure 12. By varying the uncertainty threshold, we evaluate the resulting rates of under-provisioning and over-provisioning across selected quantile-level combinations.



(a) Under-Provisioning (DeepAR) (b) Over-Provisioning (DeepAR)

(c) Under-Provisioning (TFT) (d) Over-Provisioning (TFT)

Fig. 12. **Sensitivity Analysis of Uncertainty Threshold** - Examining the sensitivity of under-provisioning and over-provisioning rates to variations in the uncertainty threshold on Google Trace.

The sensitivity analysis demonstrates that the choice of uncertainty threshold plays a crucial role in balancing the trade-off between underover-provisioning. By appropriately adjusting the uncertainty threshold, we can achieve the desired level of resource utilization efficiency while maintaining the robustness of the auto-scaling strategy. In addition, as the uncertainty threshold varies, the underover-provisioning rates exhibit distinct step-like changes, indicating that certain ranges of uncertainty thresholds yield similar effects. This observation facilitates the identification and selection of suitable uncertainty thresholds, as different values within a specific segment can lead to comparable outcomes.

D. Computation Overhead

We present the experimental setup and results that showcase the computation overhead of different approaches. The execution time consists of two key components: workload forecasting, i.e., the inference speed of forecasting models, and auto-scaling optimization, i.e., the overhead of solving auto-scaling optimization problems using solvers.

We first compare the execution time of different methods and find as listed in Table II and observe that the computational overhead of nearly all the methods is within the millisecond range, indicating that the differences were practically negligible. Furthermore, we perform a detailed cost breakdown analysis specific to our proposed method. As presented in Table III, The DeepAR model exhibited higher inference overhead compared to the TFT model due to the need for sampling from parametric distributions. In contrast, the TFT model directly provided quantile forecasts, resulting in faster inference. The computation overhead of auto-scaling optimization demonstrates minimal differences. This observation is attributed to the fact that the computational time required for calculating the uncertainty level is negligible in comparison.

TABLE II
COMPUTATION OVERHEAD COMPARISON

Methods	Execution Time
Reactive-Max	6.22 ms
Reactive-Average	9.66 ms
Hybrid(QB5000)	160.76 ms
DeepAR	912.61 ms
TFT	40.38 ms

TABLE III
COMPUTATION OVERHEAD BREAKDOWN

Workload Forecasting		Auto-Scaling Optimization	
DeepAR	TFT	Basic	Adaptive
915.94 ms	30.13 ms	7.38 ms	7.89 ms

V. DISCUSSION

A. Handling Thrashing

In our auto-scaling problem formulation, the primary objective is to minimize the number of compute nodes required to satisfy the workload threshold, without considering scaling overhead. This is mainly attributed to the relatively low overhead associated with scaling compute instances in disaggregated databases. However, without such consideration, frequent fluctuating in the number of compute nodes may occur, which is sometimes referred to as *thrashing*, or *flapping*.

A common approach to address this concern is to impose restrictions on the number of compute nodes that can be added or removed in each step, thereby promoting a smoother auto-scaling process. Several methods have been discussed in other research papers that focus on controlling the scaling operations to achieve a more gradual adjustment of resources [21], [37]. However, due to the specific focus of this paper, a detailed exploration of these methods is beyond the scope of this work.

B. Quality of Service Optimization

In our experiments, we manually set a workload threshold for scaling and evaluate the fulfillment of resource usage requirements. However, for the following reasons, we choose not to delve into the configuration of this threshold or the analysis of other quality of service (QoS) metrics.

The determination of optimal thresholds is contingent upon specific requests and distinct Service Level Objectives (SLOs). For instance, the optimization of average query latency necessitates different thresholds from that of the 99th percentile query latency. Additionally, QoS is influenced by factors beyond the workload metrics considered for scaling and does not directly reflect the effectiveness of resource scaling strategies.

Worth mentioning, performance modeling is a promising approach to tackle the challenges of threshold configuration and QoS metric analysis [38]–[41]. It enables exploration of the relationship between workload metrics, thresholds, and system performance in a controlled and reproducible manner. However, our focus lies in evaluating the ability of auto-scaling strategies to meet resource usage demands, and performance modeling necessitates further investigation in future research.

VI. RELATED WORK

In Section II-A, we provide a brief overview of the auto-scaling taxonomy and the necessity for its reevaluation due to the emergence of disaggregated cloud databases. Thanks to resource disaggregation, unlike scaling out in shared-nothing databases, which necessitates selective state migration and the scheduling of scaling activities [9], our problem formulation allows us to focus squarely on resource demand forecasting and capacity allocation.

In the realm of predictive auto-scaling, prior research has explored the utilization of time series forecasting techniques [8]–[11], [13]–[15]. Some of these works have delved into how to quantify uncertainty in workload forecasting, including leveraging stochastic process models to capture stochasticity [20] or directly predicting the possible distribution using probabilistic forecasting models [42]. However, our work differs from theirs in the following aspects.

Firstly, we extend the scope of workload forecasting by introducing the concept of quantile forecasting and expanding the range of existing approaches through an exploration of two categories of learning-based probabilistic models. Secondly, we address the gap in investigating how predicted uncertainty can inform auto-scaling tasks. Some existing probabilistic workload forecasting methods have not explored the integration of probabilistic forecasts in downstream tasks [42]. We bridge this gap in our work. More importantly, our auto-scaling approach goes beyond considering the potential workload range and enables a more refined control over the conservatism level of auto-scaling decisions.

In terms of addressing the issues, [18] aligns with our work, aiming to mitigate underestimation errors in predictions. They use a padding approach, augmenting predicted resource demand with a small value based on under-estimation errors of historical forecasts. In contrast to their approach, which serves as a complement to point forecasting models, we directly utilize probabilistic forecasting models to enhance robustness through various quantile levels. This enables us to provide a more nuanced and adaptable approach to handling underestimation errors and uncertainty in our predictions.

VII. CONCLUSION AND FUTURE WORK

In this paper, we address the issue of robustness in under-provisioning within the context of predictive auto-scaling. Our significant contributions lie in proposing a robust predictive auto-scaling approach that considers forecast uncertainty, enhancing the reliability and efficiency of resource scaling. Through the utilization of probabilistic forecasting techniques and customizable quantile levels, we achieve a balance between resource efficiency and robustness. Extensive experiments confirm the effectiveness of our approach in achieving robust auto-scaling while maintaining reasonable resource efficiency.

ACKNOWLEDGMENT

This work is supported by the Fundamental Research Funds for Alibaba Group through Alibaba Innovative Research (AIR) Program. We thank the anonymous reviewers for their valuable feedback.

REFERENCES

[1] A. Verbitski, A. Gupta, D. Saha, M. Brahmadesam, K. K. Gupta, R. Mittal, S. Krishnamurthy, S. Maurice, T. Kharatishvili, and X. Bao, "Amazon aurora: Design considerations for high throughput cloud-native relational databases," *Proceedings of the 2017 ACM International Conference on Management of Data*, 2017.

[2] W. Cao, Y. Zhang, X. Yang, F. Li, S. Wang, Q. Hu, X. Cheng, Z. Chen, Z. Liu, J. Fang, B. Wang, Y. Wang, H. Sun, Z. Yang, Z. Cheng, S. Chen, J. Wu, W. Hu, J. Zhao, Y. Gao, S. Cai, Y. Zhang, and J. Tong, "Polardb serverless: A cloud native database for disaggregated data centers," *Proceedings of the 2021 International Conference on Management of Data*, 2021.

[3] J. C. Corbett, J. Dean, M. Epstein, A. Fikes, C. Frost, J. J. Furman, S. Ghemawat, A. Gubarev, C. Heiser, P. H. Hochschild, W. C. Hsieh, S. Kanthak, E. Kogan, H. Li, A. Lloyd, S. Melnik, D. Mwaura, D. Nagle, S. Quinlan, R. Rao, L. Rolig, Y. Saito, M. Szymaniak, C. Taylor, R. Wang, and D. Woodford, "Spanner: Google's globally-distributed database," in *USENIX Symposium on Operating Systems Design and Implementation*, 2012.

[4] P. Antonopoulos, A. Budovski, C. Diaconu, A. H. Saenz, J. Hu, H. Kodavalla, D. Kossmann, S. Lingam, U. F. Minhas, N. Prakash, V. Purohit, H. Qu, C. S. Ravella, K. Reisteter, S. Shrotri, D. Tang, and V. Wakade, "Socrates: The new sql server in the cloud," *Proceedings of the 2019 International Conference on Management of Data*, 2019.

[5] S. Das, F. Li, V. R. Narasayya, and A. C. König, "Automated demand-driven resource scaling in relational database-as-a-service," *Proceedings of the 2016 International Conference on Management of Data*, 2016.

[6] A. S. Higginson, N. W. Paton, S. M. Embury, and C. Bostock, "Dbaas cloud capacity planning - accounting for dynamic rdbms system that employ clustering and standby architectures," in *International Conference on Extending Database Technology*, 2017.

[7] S. Sakr and A. Liu, "Sla-based and consumer-centric dynamic provisioning for cloud databases," *2012 IEEE Fifth International Conference on Cloud Computing*, pp. 360–367, 2012.

[8] O. Poppe, Q. Guo, W. Lang, P. Arora, M. Oslake, S. Xu, and A. Kalhan, "Moneyball: Proactive auto-scaling in microsoft azure sql database serverless," *Proc. VLDB Endow.*, vol. 15, pp. 1279–1287, 2022.

[9] R. Taft, N. El-Sayed, M. Serafini, Y. Lu, A. Aboulnaga, M. Stonebraker, R. Mayerhofer, and F. J. Andrade, "P-store: An elastic database system with predictive provisioning," *Proceedings of the 2018 International Conference on Management of Data*, 2018.

[10] L. Viswanathan, B. Chandra, W. Lang, K. Ramachandra, J. M. Patel, A. Kalhan, D. J. DeWitt, and A. Halverson, "Predictive provisioning: Efficiently anticipating usage in azure sql database," *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pp. 1111–1116, 2017.

[11] A. S. Higginson, M. Dediu, O. Arsene, N. W. Paton, and S. M. Embury, "Database workload capacity planning using time series analysis and machine learning," *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, 2020.

[12] F. R. C. Sousa, L. de Oliveira Moreira, J. S. C. Filho, and J. C. Machado, "Predictive elastic replication for multi-tenant databases in the cloud," *Concurrency and Computation: Practice and Experience*, vol. 30, 2018.

[13] Z. Gong, X. Gu, and J. Wilkes, "Press: Predictive elastic resource scaling for cloud systems," *2010 International Conference on Network and Service Management*, pp. 9–16, 2010.

[14] S. Islam, J. W. Keung, K. Lee, and A. Liu, "Empirical prediction models for adaptive resource provisioning in the cloud," *Future Gener. Comput. Syst.*, vol. 28, pp. 155–162, 2012.

[15] N. Roy, A. Dubey, and A. S. Gokhale, "Efficient autoscaling in the cloud using predictive models for workload forecasting," *2011 IEEE 4th International Conference on Cloud Computing*, pp. 500–507, 2011.

[16] L. Ma, D. V. Aken, A. S. Hefny, G. Mezerhane, A. Pavlo, and G. J. Gordon, "Query-based workload forecasting for self-driving database management systems," *Proceedings of the 2018 International Conference on Management of Data*, 2018.

[17] Y. Gao, X. Huang, X. Zhou, X. Gao, G. Li, and G. Chen, "Dbaugur: An adversarial-based trend forecasting system for diversified workloads," 2022.

[18] Z. Shen, S. Subbiah, X. Gu, and J. Wilkes, "Cloudscale: elastic resource scaling for multi-tenant cloud systems," *Proceedings of the 2nd ACM Symposium on Cloud Computing*, 2011.

[19] W. Lang, K. Ramachandra, D. J. DeWitt, S. Xu, Q. Guo, A. Kalhan, and P. Carlin, "Not for the timid: On the impact of aggressive over-booking in the cloud," *Proc. VLDB Endow.*, vol. 9, pp. 1245–1256, 2016.

[20] H. Qian, Q. Wen, L. Sun, J. Gu, Q. Niu, and Z. Tang, "Robustscaler: Qos-aware autoscaling for complex workloads," *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pp. 2762–2775, 2022.

[21] T. T. Nguyen, Y.-J. Yeom, T. Kim, D.-H. Park, and S. Kim, "Horizontal pod autoscaling in kubernetes for elastic container orchestration," *Sensors (Basel, Switzerland)*, vol. 20, 2020.

[22] A. Pavlo, G. Angulo, J. Arulraj, H. Lin, J. Lin, L. Ma, P. Menon, T. C. Mowry, M. Perron, I. Quah, S. Santurkar, A. Tomasic, S. Toor, D. V. Aken, Z. Wang, Y. Wu, R. Xian, and T. Zhang, "Self-driving database management systems," in *Conference on Innovative Data Systems Research*, 2017.

[23] T. Gneiting and M. Katzfuss, "Probabilistic forecasting," *Annual Review of Statistics and Its Application*, vol. 1, pp. 125–151, 2014.

[24] K. Benidis, S. S. Rangapuram, V. Flunkert, Y. Wang, D. Maddix, C. Turkmen, J. Gasthaus, M. Bohlke-Schneider, D. Salinas, L. Stella *et al.*, "Deep learning for time series forecasting: Tutorial and literature survey," *ACM Computing Surveys*, vol. 55, no. 6, pp. 1–36, 2022.

[25] S. E. Haupt, M. G. Casado, M. Davidson, J. Dobschinski, P. Du, M. Lange, T. Miller, C. Mohrlen, A. Motley, R. Pestana *et al.*, "The use of probabilistic forecasts: Applying them in theory and practice," *IEEE Power and Energy Magazine*, vol. 17, no. 6, pp. 46–57, 2019.

[26] X. Yan, W. Zhang, L. Ma, W. Liu, and Q. Wu, "Parsimonious quantile regression of financial asset tail dynamics via sequential learning," *ArXiv*, vol. abs/2010.08263, 2018.

[27] F. Li, "Cloud-native database systems at alibaba: Opportunities and challenges," *Proceedings of the VLDB Endowment*, vol. 12, no. 12, pp. 2263–2272, 2019.

[28] L. Hao and D. Q. Naiman, *Quantile regression*. Sage, 2007, no. 149.

[29] V. Flunkert, D. Salinas, and J. Gasthaus, "Deepar: Probabilistic forecasting with autoregressive recurrent networks," *ArXiv*, vol. abs/1704.04110, 2017.

[30] B. Lim, S. Ö. Arik, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time series forecasting," *ArXiv*, vol. abs/1912.09363, 2019.

[31] J. Wilkes, "More Google cluster data," Google research blog, Mountain View, CA, USA, Nov. 2011, posted at http://googleresearch.blogspot.com/2011/11/more-google-cluster-data.html.

[32] A. Alexandrov, K. Benidis, M. Bohlke-Schneider, V. Flunkert, J. Gasthaus, T. Januschowski, D. C. Maddix, S. S. Rangapuram, D. Salinas, J. Schulz, L. Stella, A. C. Türkmen, and B. Wang, "Gluonts: Probabilistic and neural time series modeling in python," *J. Mach. Learn. Res.*, vol. 21, pp. 116:1–116:6, 2020.

[33] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019.

[34] B. K. Nelson, "Time series analysis using autoregressive integrated moving average (arima) models," *Academic emergency medicine*, vol. 5, no. 7, pp. 739–744, 1998.

[35] M. Popescu, V. E. Balas, L. Perescu-Popescu, and N. E. Mastorakis, "Multilayer perceptron and neural networks," *WSEAS Transactions on Circuits and Systems archive*, vol. 8, pp. 579–588, 2009.

[36] K. Rzadca, P. Findeisen, J. Swiderski, P. Zych, P. Broniek, J. Kusmierek, P. Nowak, B. Strack, P. Witusowski, S. Hand *et al.*, "Autopilot: workload autoscaling at google," in *Proceedings of the Fifteenth European Conference on Computer Systems*, 2020, pp. 1–16.

[37] S. Luo, H. Xu, K. Ye, G. Xu, L. Zhang, G. Yang, and C. Xu, "The power of prediction: microservice auto scaling via workload learning," *Proceedings of the 13th Symposium on Cloud Computing*, 2022.

[38] J. Duggan, O. Papaemmanouil, U. Çetintemel, and E. Upfal, "Contender: A resource modeling approach for concurrent query performance prediction," in *International Conference on Extending Database Technology*, 2014.

[39] B. Mozafari, C. Curino, A. Jindal, and S. Madden, "Performance and resource modeling in highly-concurrent oltp workloads," in *ACM SIGMOD Conference*, 2013.

[40] J. Schaffner, B. Eckart, D. Jacobs, C. Schwarz, H. Plattner, and A. Zeier, "Predicting in-memory database performance for automating cluster management tasks," *2011 IEEE 27th International Conference on Data Engineering*, pp. 1264–1275, 2011.

[41] P. Padala, K.-Y. Hou, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, and A. Merchant, "Automated control of multiple virtualized resources," in *Proceedings of the 4th ACM European conference on Computer systems*, 2009, pp. 13–26.

[42] X. Huang, S. Cao, Y. Gao, X. Gao, and G. Chen, "Lightpro: Lightweight probabilistic workload prediction framework for database-as-a-service," *2022 IEEE International Conference on Web Services (ICWS)*, pp. 160–169, 2022.