Textile & Leather Review

ISSN 2623-6281| www.tlr-journal.com | 10.31881/TLR

Research on AI-Driven Load Prediction and Elastic Scaling Intelligent Decision-Making Technology for Multi-Cloud Heterogeneous Resources

Chenglin Li, Yonghui Ren, Jing Bai

How to cite: Li C, Ren Y, Bai J. Research on AI-Driven Load Prediction and Elastic Scaling Intelligent Decision-Making Technology for Multi-Cloud Heterogeneous Resources. Textile & Leather Review. 2026; 9:3497-3521. https://doi.org/10.31881/TLR.2026.3497

How to link: https://doi.org/10.31881/TLR.2026.3497

Published: 25 April 2026

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License

Research on AI-Driven Load Prediction and Elastic Scaling Intelligent Decision-Making Technology for Multi-Cloud Heterogeneous Resources

Chenglin Li*, Yonghui Ren, Jing Bai

Digital Intelligence Operations Center of Yunnan Power Grid Corporation Limited, Kunming 650000, Yunnan, China

*a644910687@163.com

Article

https://doi.org/10.31881/TLR.2026.3497

Published 25 April 2026

ABSTRACT

To address the challenges of complex load fluctuations and lagging resource adaptation within the digitized production processes of the textile industry, this paper proposes an AI-driven intelligent decision-making scheme for load prediction and elastic scaling in multi-cloud heterogeneous environments. This approach ensures that the high-frequency data generated by automated weaving and dyeing systems are processed with optimal efficiency, allowing computational resources to dynamically align with the volatile demands of modern textile manufacturing. An improved extended Long Short-Term Memory (xLSTM) load prediction model is proposed, which enhances the adaptability to heterogeneous data and the ability to capture burst loads through multi-source feature fusion, adaptive normalization, and lightweight optimization. A decision-making mechanism based on improved Proximal Policy Optimization (PPO) is designed, integrating a multi-objective optimization function, fuzzy comprehensive evaluation for adaptability assessment, and a dynamic balance strategy. A hybrid simulation platform combining OpenStack and Amazon Web Services (AWS) is built, and comparative experiments are conducted under three types of load scenarios. The results show that the proposed prediction model achieves a Mean Absolute Percentage Error (MAPE) below 2.5% in normal, sudden, and fluctuating load scenarios. Compared with traditional methods, the decision-making mechanism achieves a cost optimization rate of 18.3%, improves the Service Level Agreement (SLA) compliance rate to 99.2%, and reduces the decision delay to 280 ms. This research establishes a closed loop prediction-decision-scheduling framework, providing technical support for the efficient management of multi-cloud heterogeneous resources in digitized textile manufacturing.

KEYWORDS

AI-driven, multi-cloud heterogeneous, load prediction, elastic scaling, textile manufacturing

INTRODUCTION

Research Background and Significance

Rise and Development Trend of Multi-Cloud Heterogeneous Environments

With the in-depth development of the digital economy and cloud computing, textile enterprises' demand for elastic IT resources and requirements for security redundancy continue to increase to support the high-frequency data processing of intelligent weaving and dyeing systems. This digital transition ensures that the industry's computational infrastructure can dynamically scale to maintain operational continuity and data integrity amidst the increasingly complex demands of modern textile manufacturing. More and more enterprises are migrating their core production business systems to the cloud [1]. The multi-cloud architecture has become the mainstream choice for enterprises' digital transformation due to its advantages of integrating resources from different cloud vendors and disaster recovery capabilities.

According to Gartner's report, more than 85% of enterprises worldwide will adopt a multi-cloud strategy by 2025, and heterogeneity (differences in computing architectures, storage protocols, and network standards) has become a core feature of multi-cloud environments [2]. Although this heterogeneity improves the flexibility of resource allocation, it also leads to more complex resource load fluctuations, posing severe challenges to load prediction and elastic scaling [3].

Core Value of Load Prediction and Elastic Scaling for Multi-Cloud Heterogeneous Resources

Load prediction is a prerequisite for elastic scaling. Accurate load prediction can prevent resource over-provisioning and performance degradation caused by under-provisioning [4]. As a core means of multi-cloud resource management, elastic scaling's intelligent decision-making capability directly determines resource utilization and Quality of Service (QoS) [5].

In multi-cloud heterogeneous scenarios, AI-based load prediction and intelligent scaling enable textile enterprises to maximize the cost-performance advantages of different cloud providers while maintaining the stable operation of automated production lines under fluctuating workloads. This technological integration provides both theoretical and practical value for the textile industry by maintaining seamless synchronization between high-speed weaving data processing and elastic computational resources.

Challenges and Bottlenecks of Existing Technologies

Current technologies face three core bottlenecks:

(1) Traditional load prediction methods (such as ARIMA and SVM) are difficult to capture the nonlinear and sudden characteristics of loads in multi-cloud heterogeneous environments, resulting in large prediction errors [6].

(2) Scaling decisions mostly rely on static thresholds or rules, which cannot dynamically adapt to heterogeneous resource differences and load fluctuations [7].

(3) The prediction and decision-making processes are disconnected and therefore fail to form a closed loop prediction-decision-scheduling mechanism, which leads to delayed scaling and resource mismatch.

Review of Domestic and Foreign Research Status

Research Progress in Multi-Cloud Heterogeneous Resource Management

Foreign research focuses on optimizing multi-cloud resource scheduling algorithms. For example, Google proposed the cross-cloud resource orchestration framework Kubernetes Federation, which realizes the unified management of basic resources but does not conduct in-depth optimization for load prediction and scaling decisions [8].

Alibaba Cloud, a leading domestic cloud service provider, has launched its Kubernetes-based container platform, Alibaba Cloud Container Service for Kubernetes (ACK), which provides scalable and high-performance container application management and supports the full lifecycle of enterprise-level containerized applications.

Existing research mostly focuses on resource integration and lacks the collaborative design of load prediction and intelligent decision-making.

Research Status of Load Prediction Technologies

Among traditional load prediction methods, the ARIMA model is widely used due to its simplicity and ease of implementation, but its MAPE generally exceeds 10% when dealing with nonlinear loads [9]; the SVM model has strong generalization ability but low training efficiency, making it difficult to adapt to real-time scenarios [10].

Among AI methods, deep learning models such as LSTM and GRU have become mainstream. Compared with LSTM, GRU has a simpler structure and effectively improved computational efficiency [11]. Li et al. proposed a BiLSTM-GRU combined prediction model, which effectively improved prediction accuracy, reduced prediction time, and optimized the scaling performance of cloud computing containers [12]. However, existing models do not fully consider the differences in multi-cloud heterogeneous data, resulting in insufficient adaptability.

Research Status of Elastic Scaling Decision-Making Mechanisms

Existing decision-making mechanisms can be divided into three categories: rule-based, optimization-based, and learning-based.

Rule-based decision-making (such as based on CPU utilization thresholds) is simple to implement but lacks flexibility.

Optimization-based decision-making (such as based on integer programming) can achieve cost optimization but has high solution complexity.

Learning-based decision-making (such as reinforcement learning) has dynamic adaptability. Among them, algorithms such as DQN and PPO have been verified to be effective in single-cloud environments [13,14], but the state space modeling and reward function design in multi-cloud heterogeneous scenarios still need to be broken through.

Research Content and Technical Route

Core Research Content

The study focuses on the following aspects:

(1) Construct a load prediction model suitable for multi-cloud heterogeneous scenarios, focusing on solving the problems of data heterogeneity and burst loads.

(2) Design an intelligent decision-making mechanism for elastic scaling based on reinforcement learning to achieve multi-objective optimization.

(3) Build a multi-cloud heterogeneous simulation platform and verify the effectiveness and superiority of the scheme through experiments.

Technical Route Design

The technical route is divided into four steps:

(1) Data collection and preprocessing: Integrate multi-source heterogeneous load data and perform feature engineering.

(2) Model construction and training: Design an improved xLSTM prediction model and a reinforcement learning decision-making model.

(3) Mechanism optimization: Propose multi-cloud resource scheduling strategies and dynamic balance mechanisms;

(4) Experimental verification: Evaluate model performance through multi-scenario comparative experiments.

RELATED TECHNICAL FOUNDATIONS

Core Characteristics of Multi-Cloud Heterogeneous Resource Environments

The core characteristics of multi-cloud heterogeneous resource environments are reflected in three aspects:

(1) Resource heterogeneity: significant differences exist in computing nodes (x86/ARM architectures), storage devices (block storage/object storage), and network bandwidth among different cloud vendors, leading to great difficulties in resource adaptation.

(2) Management complexity: each cloud platform has independent API interfaces and scheduling mechanisms, lacking unified resource management standards.

(3) Load volatility: affected by factors such as business peaks, network delays, and regional failures, the load presents multi-cycle (daily/weekly/monthly) and sudden characteristics.

Load Prediction-Related Technologies

Principles and Limitations of Traditional Load Prediction Methods

Traditional load prediction methods are mainly linear models and machine learning models [15].

The ARIMA model stabilizes non-stationary sequences through differencing and then models based on autocorrelation coefficients, but it is only suitable for linear stationary load sequences and has large prediction errors for burst loads.

The SVM model maps load data to a high-dimensional space through kernel functions to achieve nonlinear fitting, but it has low training efficiency when processing massive time-series data and poor real-time performance.

Core Advantages and Application Scenarios of AI Prediction Technologies

AI prediction technologies, especially deep learning models, have strong nonlinear fitting and feature learning capabilities, making them more suitable for multi-cloud heterogeneous scenarios.

Their core advantages include:

(1) Ability to automatically extract temporal features of loads and heterogeneous resource features.

(2) Balancing prediction accuracy and real-time performance through model improvement.

(3) Supporting multi-source data fusion to improve adaptability to complex scenarios.

Among them, LSTM and GRU models are suitable for short-term and medium-term load prediction, Transformer models are suitable for long-cycle load prediction, and xLSTM models perform better in long-context processing and parallel computing.

Core AI Models for Temporal Load Prediction

The core AI models for temporal load prediction include LSTM, GRU, and xLSTM.

LSTM solves the problem of RNN gradient disappearance through the gating mechanism of forget gate, input gate, and output gate, and can effectively capture long-cycle load features, but it has many parameters and slow training speed [16].

GRU simplifies the LSTM structure, retaining only the update gate and reset gate, reducing parameters by 30% and improving training speed by 42%, but its accuracy is slightly inferior to LSTM in complex load scenarios [17].

As an extended variant of LSTM, xLSTM introduces exponential gating and matrix memory units, realizing parallel computing while enhancing long-context processing capabilities. Its time complexity is O(N), making it more suitable for massive load data processing [18].

Key Technologies for Elastic Scaling

Definition and Core Indicators of Elastic Scaling

Elastic scaling refers to the process of dynamically adjusting resource allocation according to load changes. Its core indicators include:

(1) Response delay: the time from load exceeding the threshold to the completion of resource scaling.

(2) Cost optimization rate: the reduction ratio of resource costs after scaling compared with static configuration.

(3) SLA compliance rate: the ratio of indicators such as service availability and response time meeting the agreed standards.

(4) Resource utilization rate: the actual usage ratio of resources after scaling.

Special Requirements for Elastic Scaling in Multi-Cloud Environments

Elastic scaling in multi-cloud environments needs to meet three special requirements:

(1) Heterogeneous resource adaptation: select appropriate resource types according to business types (e.g., x86 architecture for CPU-intensive businesses, ARM architecture for energy-sensitive businesses).

(2) Cross-cloud collaboration: solve the problems of resource scheduling and data synchronization among different cloud platforms.

(3) Disaster recovery redundancy: scaling decisions need to consider the risk of single-cloud failures to ensure the redundancy of resource distribution.

AI-DRIVEN LOAD PREDICTION MODEL FOR MULTI-CLOUD HETEROGENEOUS RESOURCES

Load Data Preprocessing and Feature Engineering

Load Data Collection Scheme in Multi-Cloud Heterogeneous Environments

A multi-source data collection scheme is designed:

(1) Collect basic load data (CPU utilization, memory utilization, network IO) through open APIs of various cloud platforms (such as AWS CloudWatch and OpenStack Nova API).

(2) Collect business feature data (user traffic, number of transactions) through business system logs.

(3) Collect environmental data (network delay, regional fault alarms) through third-party tools.

The collection frequency is set to 1 minute/time to ensure data timeliness.

Data Cleaning and Outlier Handling

A hierarchical cleaning strategy is adopted for heterogeneous data differences:

First, unify data formats (e.g., unify the unit of CPU utilization of different cloud platforms to percentage).

Second, use the interquartile range (IQR) method to detect outliers and eliminate data outside the interval $[Q_1 - 1.5\ IQR, Q_3 + 1.5\ IQR]$.

Finally, use linear interpolation to fill in missing values to avoid the impact of data sparsity on model performance.

The data normalization in preprocessing is used only for format unification and range standardization (e.g., scaling values to [0,1]) to eliminate dimensional differences between heterogeneous data sources. This step is independent of the adaptive normalization mechanism in the model's exponential gating, and the two components are not in conflict. The latter dynamically adjusts the normalization parameters (mean and variance) according to the architectural characteristics of the current input data (e.g., x86/ARM) during model inference, aiming to adapt to the inherent performance baseline differences of heterogeneous computing resources.

Feature Extraction and Fusion

Three types of core features are extracted:

(1) Temporal features: hourly, daily, and weekly cycle features extracted through Fourier transform.

(2) Resource features: CPU architecture, storage type, and bandwidth size converted through one-hot encoding.

(3) Business features: user access peaks and transaction types extracted through statistical methods.

Multi-source feature fusion is achieved through feature concatenation to construct a 64-dimensional feature vector.

Feature Selection and Dimensionality Reduction Optimization

Kernel Principal Component Analysis (KPCA) is used for feature selection and dimensionality reduction. The

fused features are mapped to a high-dimensional space through a Gaussian kernel function, and the principal components with a cumulative variance contribution rate of more than 98% are selected to reduce the feature dimension from 64 to 24, which not only retains core information but also reduces model computational complexity.

The construction of the 64-dimensional high-dimensional feature vector is intentional to comprehensively cover the potential correlations between multi-source heterogeneous data (temporal, resource, and business features). Many features that may appear redundant are in fact informative, as they capture overlapping associations across different dimensions. KPCA does not discard 62% of the information; rather, it projects the original 64-dimensional feature space onto 24 principal components while preserving 98% of the cumulative explained variance. This transformation represents a process of information compression and structure extraction rather than information loss. This approach avoids the loss of key association information caused by direct manual feature screening and ensures the model's ability to capture complex load patterns.

Construction of Load Prediction Model Based on Improved AI Algorithm

Model Design Ideas

The model is designed based on the following principles:

(1) Adapt to heterogeneous data by integrating multi-source heterogeneous features through a feature fusion module.

(2) Enhance the ability to capture burst loads by introducing an attention mechanism to strengthen key time-step features.

(3) Balance accuracy and real-time performance by optimizing training efficiency through the parallel architecture of xLSTM.

The model consists of four layers: input layer, feature fusion layer, improved xLSTM layer, and output layer.

Selection and Improvement of Basic Prediction Model

xLSTM is selected as the basic model, with two improvements:

(1) Introduce an adaptive normalization mechanism in the exponential gating to improve the model's adaptability to heterogeneous data;

The adaptive normalization mechanism is mathematically defined as: For input data x from heterogeneous architectures, the normalization function is $x = \frac{x-\mu_\theta\ （A）}{\sigma_\theta\ （A）+\epsilon}$, where A represents the architecture type (x86/ARM), $\mu_\theta\ （A）$ and $\sigma_\theta\ （A）$ are the dynamic mean and variance learned by the model for architecture A, and ϵ = 10^{-6} is a smoothing term. This mechanism dynamically adjusts the normalization parameters according to the architecture type of the input data, effectively bridging the performance baseline gap between x86 and ARM architectures.

(2) Add a load fluctuation factor in the matrix memory unit to enhance the ability to capture burst loads.

The load fluctuation factor is defined as follows: Let L=[l_1, l_2, ..., l_T] be the load sequence within the sliding window of size T, then the load fluctuation factor $\gamma = \frac{\mathrm{std}(L)}{\mathrm{mean}(L)+\epsilon}$, where $\mathrm{std}(L)$ is the standard deviation of the load sequence, $\mathrm{mean}(L)$ is the average value, and ϵ = 10^{-6} avoids division by zero. The factor γ is incorporated into the matrix memory unit as an additive weighting term, i.e., $z_t = \sigma(W_Z[h_{t-1}, x_t] + b_z + \alpha\gamma)$, where α= 0.1 is the adjustment coefficient, allowing the model to dynamically regulate the memory update intensity according to the degree of load fluctuation.

The parameters of the improved model are set as follows: the hidden-layer dimension is 256, the learning rate is 0.001, the batch size is 64, and the number of training epochs is 100.

Multi-Model Fusion Strategy

A weighted fusion strategy is adopted to fuse the improved xLSTM and GRU models:

Determine weights through the validation set (improved xLSTM weight of 0.7, GRU weight of 0.3).

Dynamically adjust weights using a sliding window mechanism. When the load fluctuation coefficient is greater than 0.3, increase the GRU weight to 0.4 to enhance model stability.

The output of the fusion model is the weighted sum of the prediction results of the two models.

Model Lightweight Optimization

Model pruning and quantization optimization are adopted:

Prune redundant connections (retain connections with absolute weight values greater than 0.01), reducing model parameters by 40%.

Quantize model weights from 32-bit floating-point numbers to 16-bit fixed-point numbers, improving

inference speed by 55% and ensuring that the prediction delay is controlled within 500ms to meet real-time prediction requirements.

Training and Optimization of Prediction Model

Dataset Construction and Partitioning

The dataset consists of two parts:

(1) Public dataset: Google Cluster Trace, containing 120 million load records.

(2) Self-built dataset: Collect 3 months of load data under the hybrid architecture of AWS and OpenStack, totaling 518,000 records.

Partition the dataset into training set, validation set, and test set in a ratio of 7:2:1, and adopt data augmentation techniques (time reversal, noise addition) to improve model generalization ability.

Selection of Loss Function and Optimizer

MAPE is selected as the loss function, which is more suitable for the error evaluation of load prediction. The formula is:

$$MAPE=\sum|(y_{true}-y_{pred})/y_{true}|/n\times 100\% \quad (1)$$

AdamW is selected as the optimizer to suppress overfitting through weight decay. The weight decay coefficient is set to 0.0001, and the momentum parameters are $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

Hyperparameter Tuning Method

Bayesian optimization is used for hyperparameter tuning. The optimized variables include learning rate (0.0001−0.01), hidden layer dimension (128−512), and batch size (32−128).

Taking the validation set MAPE as the optimization target, the optimal hyperparameter combination is obtained after 20 iterations: learning rate of 0.001, hidden layer dimension of 256, and batch size of 64.

Performance Verification of Prediction Model

Design of Evaluation Indicators

Four core evaluation indicators are selected: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and Coefficient of Determination (R^2).

Among them, MAPE is the core indicator for evaluating prediction accuracy; R^2 is used to evaluate model fitting degree, and the closer it is to 1, the better the fitting effect.

Design of Comparative Experiments

Five groups of comparative experiments are set up: ARIMA, SVM, basic LSTM, GRU, and improved xLSTM fusion model.

Compare the prediction performance of each model under the same dataset and experimental environment to verify the superiority of the proposed model.

INTELLIGENT DECISION-MAKING MECHANISM FOR ELASTIC SCALING BASED ON LOAD PREDICTION

Objectives and Constraints of Elastic Scaling Decision-Making

Core Objectives

Establish multi-objective optimization objectives:

(1) Cost optimization: minimize multi-cloud resource rental costs.

(2) Performance guarantee: ensure CPU utilization ≤ 80%, memory usage rate ≤ 75%, and network delay ≤ 100ms.

(3) SLA compliance: service availability ≥ 99.9%.

Convert multi-objective to a single-objective function using the weighted method, with weights set as 0.4 for cost, 0.3 for performance, and 0.3 for SLA.

Key Constraints

Clarify three key constraints:

(1) Resource upper limit constraint: the scaled resources of a single cloud platform shall not exceed 30% of

its total resources.

(2) Scaling delay constraint: cross-cloud scaling delay ≤ 5 minutes, single-cloud scaling delay ≤ 1 minute.

(3) Compatibility constraint: scaled resources must be compatible with business types (e.g., GPU-intensive businesses need to select cloud nodes that support GPUs).

Construction of Multi-Objective Optimization Function

Construct a multi-objective optimization function:

$$\min F(x) = 0.4\ C(x)+0.3\ P(x)+0.3\ (1\text{-}S(x)) \tag{2}$$

Where $C(x)$ is the normalized value of resource cost, $P(x)$ is the normalized value of performance loss, and $S(x)$ is the SLA compliance rate.

The constraints are defined as follows: $x \in [x_{min}, x_{max}]$ (resource upper limit), $t \leq t_{max}$ (scaling delay), and $Cmp(x) = 1$ (compatibility constraint, where 1 indicates compatibility).

Design of Intelligent Decision-Making Model Based on Reinforcement Learning

Markov Decision Process Modeling of Decision-Making Problems

Model the scaling decision-making problem as a Markov Decision Process (MDP):

The state space S includes load prediction values, current resource configuration, resource costs, and SLA compliance status, with a dimension of 18.

The action space A includes three types of actions: scaling (selecting cloud vendors, resource types, and scaling scales), scaling down, and maintaining the status quo, totaling 32 specific actions.

The state transition probability $P(s'|s,a)$ represents the probability of transitioning from state s to state s' after taking action a.

Design of State Space, Action Space, and Reward Function

A hierarchical strategy is adopted for reward function design.

The basic reward is defined as $R_0 = 10 - 0.1C(x) + 0.2\ S(x)$, which reflects cost and SLA objectives.

The penalty terms are given by $R_1 = -50$ for compatibility constraints and $R_2 = -20$ when the scaling delay

exceeds the predefined threshold.

The incentive term is defined as $R_3 = 30$ when the performance indicators reach the optimal level.

The total reward is computed as $R = R0 + R1 + R2 + R3.$

Selection and Improvement of Reinforcement Learning Algorithm

The PPO algorithm is selected and improved:

(1) Introduce a prioritized experience replay mechanism to improve the sampling probability of key decision-making experiences.

(2) Adopt an adaptive learning rate strategy to adjust the learning rate according to reward value fluctuations (reduce the learning rate when fluctuations are large).

The policy network of the improved PPO algorithm is a 3-layer fully connected network, and the value network is a 2-layer fully connected network, with 200 iterations.

Model Training and Convergence Analysis

An incremental training method is adopted: First pre-train the model based on historical data, then update the model parameters in real-time through online learning.

Convergence analysis shows that the reward value of the model tends to be stable after 120 iterations (fluctuation range ≤ 5%), proving that the model has good convergence and can be used for practical decision-making.

The claim of convergence after 120 iterations is supported by sufficient experimental evidence: (1) The reward value curve shows that after 120 iterations, the reward value fluctuates within ± 2% (far below the 5% threshold), and there is no significant upward or downward trend, indicating that the model has entered a stable state; (2) The validation set's decision deviation rate stabilizes at ≤ 7% after 120 iterations, and further increasing the number of iterations (up to 200) does not significantly improve performance, excluding the possibility of local optimum; (3) The improved PPO algorithm (prioritized experience replay + adaptive learning rate) accelerates the convergence speed—compared with the basic PPO algorithm (which requires about 300 iterations to converge), the proposed algorithm reduces the convergence iterations by 60% through efficient experience utilization and learning rate adjustment, making 120 iterations a reasonable and reliable result for this specific model structure.

Scaling Scheduling Strategy in Multi-Cloud Heterogeneous Environments

Adaptability Evaluation Method for Heterogeneous Resources

A fuzzy comprehensive evaluation-based adaptability evaluation method is proposed:

Establish an evaluation index system (CPU architecture adaptability, memory adaptability, bandwidth adaptability, cost-performance ratio), and determine weights through the analytic hierarchy process;

Use triangular fuzzy numbers to represent evaluation results, and obtain adaptability scores through fuzzy matrix operations. A score ≥ 0.8 is considered adaptable.

Multi-Cloud Resource Selection and Allocation Strategy

A two-layer cost-performance selection strategy is adopted.

In the first layer, cloud vendors and resource types with adaptability scores ≥ 0.8 are selected.

In the second layer, the cost-performance ratio (CPR), defined as the cost divided by the performance score, is calculated for each candidate scheme, and the scheme with the minimum CPR is chosen.

The allocation strategy adopts a greedy algorithm, prioritizing resource allocation to business nodes with the highest load prediction values.

Dynamic Balance Mechanism for Scaling and Scaling Down

A dynamic balance mechanism based on load thresholds is designed:

Trigger scaling when the predicted load ≥ threshold (CPU utilization 70%), and the scaling scale = ceil ((predicted load - threshold)/unit resource load capacity).

Trigger scaling down when the actual load ≤ lower threshold (CPU utilization 30%), and the scaling down scale = floor ((lower threshold - actual load)/unit resource load capacity).

Trigger scaling down only when the actual load ≤ lower threshold (CPU utilization 30%) and the duration of maintaining this state reaches 3 consecutive minutes. This delay window is introduced to prevent the thrashing effect caused by sudden load drops followed by rapid spikes. Additionally, the load buffer interval is adjusted from 5% to 8%, and the scaling down scale is limited to 50% of the current redundant resources per operation. These improvements effectively reduce redundant resource churn and ensure that cross-cloud scaling operations (with a 5-minute delay constraint) are not triggered frequently, avoiding SLA violations

caused by scheduling delays.

Set a 5% load buffer interval to avoid frequent scaling and scaling down.

Effectiveness Verification of Intelligent Decision-Making Mechanism

Evaluation of Scaling Response Speed and Accuracy

Evaluate efficiency and accuracy through decision delay and decision deviation:

Decision delay is defined as the time from receiving the prediction result to outputting the decision.

Decision deviation is defined as the deviation between the actual scaling effect and the optimal scheme.

The verification results indicate that the decision delay of the proposed mechanism is ≤ 300 ms, and the decision deviation is ≤ 8%, which is superior to traditional decision-making methods.

Analysis of Multi-Objective Optimization Effect

Evaluate the multi-objective optimization effect through cost optimization rate, performance compliance rate, and SLA compliance rate.

Comparative experiments demonstrate that the proposed mechanism improves the cost optimization rate by more than 15% compared with traditional methods, the performance compliance rate is ≥ 95%, and the SLA compliance rate is ≥ 99%, achieving multi-objective balance.

EXPERIMENTAL VERIFICATION AND PERFORMANCE ANALYSIS

Experimental Environment Construction

Construction of Multi-Cloud Heterogeneous Simulation Platform

Build a multi-cloud heterogeneous simulation platform:

Integrate OpenStack (private cloud) and AWS (public cloud) to realize cross-cloud resource orchestration through Terraform.

Deploy Docker containerized business systems (Web services, database services) to simulate real business loads.

Use Prometheus to monitor load data and Grafana for visual experimental results.

Preparation of Experimental Dataset

The dataset includes real and simulated data:

Real data comes from 3 months of load records (CPU, memory, network IO) of a multi-cloud environment of an e-commerce platform;

Simulated data is generated through LoadRunner, including three scenarios: normal load (fluctuation coefficient ≤ 0.2), sudden load (fluctuation coefficient ≥ 0.5), and fluctuating load (fluctuation coefficient 0.2-0.5), totaling 1 million records.

Software and Hardware Configuration and Parameter Settings

Hardware configuration: CPU is Intel Xeon E5-2690 v4 (28 cores), memory 128 GB, hard disk 1 TB SSD, network bandwidth 10 Gbps.

Software configuration: Operating system Ubuntu 20.04, deep learning framework TensorFlow 2.10, reinforcement learning framework Stable Baselines3, Python 3.9. The experimental parameters are shown in Table 1.

Table 1. Experimental Parameters

Parameter Type	Parameter Name	Parameter Value
Prediction model parameters	Learning rate, batch size, number of iterations	0.001, 64, 100
Reinforcement learning parameters	Learning rate, number of iterations, reward weight	0.0003, 200, cost 0.4/performance 0.3/SLA 0.3
Experimental scenario parameters	Load fluctuation coefficient, experimental duration	0.1-0.6, 72 hours

Experimental Design Scheme

Experimental Objectives and Core Verification Indicators

Experimental objectives: Verify the accuracy and real-time performance of the load prediction model, and the multi-objective optimization effect of the elastic scaling decision-making mechanism.

Core verification indicators: Prediction model indicators (MAE, RMSE, MAPE, prediction delay); decision-making mechanism indicators (cost optimization rate, performance compliance rate, SLA compliance rate, decision delay).

Selection of Comparative Algorithms and Schemes

Comparative algorithms for prediction models: ARIMA, SVM, basic LSTM, GRU; comparative schemes for decision-making mechanisms: rule-based decision-making based on thresholds, optimization-based decision-making based on integer programming, and basic PPO decision-making.

All comparative algorithms/schemes are run under the same experimental environment to ensure fairness.

Design of Different Experimental Scenarios

Design three types of experimental scenarios: Normal load scenario (CPU utilization 30%-60%, fluctuation coefficient 0.1-0.2); sudden load scenario (load peak occurs from 10:00 to 12:00 every day, fluctuation coefficient 0.5-0.6); fluctuating load scenario (load fluctuates randomly, fluctuation coefficient 0.2-0.5). Each scenario runs for 24 hours, and experimental data is recorded.

Analysis of Load Prediction Model Experimental Results

Comparison of Prediction Accuracy Under Different Scenarios

The comparison results of prediction accuracy of each prediction model under different scenarios are shown in Table 2.

Table 2. Comparison of Prediction Accuracy of Each Prediction Model Under Different Scenarios

Model	Normal Load	Sudden Load	Fluctuating Load	Average

	(MAPE)	(MAPE)	(MAPE)	RMSE
ARIMA	8.7%	10.9%	9.3%	18.6
SVM	6.2%	7.8%	6.9%	12.3
Basic LSTM	2.9%	4.2%	3.5%	5.7
GRU	3.1%	4.5%	3.7%	6.1
Improved xLSTM Fusion Model	1.8%	2.4%	2.1%	3.2

The experimental results indicate that the proposed xLSTM fusion model achieves the best performance across all three scenarios. Among them, the MAPE in the sudden load scenario is only 2.4%, which is 42.9% lower than that of the basic LSTM and 78.3% lower than that of ARIMA, proving its strong ability to capture burst loads.

Analysis of Model Training Efficiency and Real-Time Performance

The analysis results of model training efficiency and real-time performance are shown in Table 3.

After lightweight optimization, the training time of the proposed model is 40% shorter than that of the basic LSTM, and the prediction delay is controlled within 420ms, meeting real-time prediction requirements. This improvement can be attributed to the parallel computing architecture of xLSTM as well as the pruning and quantization strategies.

Table 3. Analysis of Model Training Efficiency and Real-Time Performance

Model	Training Time (hours)	Prediction Delay (ms)	Memory Occupation (GB)
Basic LSTM	8.2	680	4.2

GRU	6.5	550	3.5
Improved xLSTM Fusion Model	4.9	420	2.8

Verification Results of Heterogeneous Data Adaptability

Verify the model's heterogeneous data adaptability by comparing the prediction accuracy under different data types (x86/ARM architecture load data, block storage/object storage load data).

The results demonstrate that the MAPE variation of the proposed model across different data types is ≤ 0.3%, whereas that of the basic LSTM reaches 1.2%, indicating that feature fusion and adaptive normalization mechanisms effectively improve adaptability to heterogeneous data.

Analysis of Elastic Scaling Decision-Making Mechanism Experimental Results

Evaluation of Scaling Response Speed and Accuracy

The evaluation results of scaling response speed and accuracy are shown in Table 4.

Table 4. Evaluation of Scaling Response Speed and Accuracy

Decision-Making Scheme	Average Decision Delay (ms)	Scaling Deviation (%)	Misdecision Rate (%)
Rule-based Decision-Making	1000	20	12.5
Integer Programming Optimization-based Decision-Making	850	12	8.3
Basic PPO Decision-Making	370	10	5.1
Improved PPO Decision-Making (Proposed in This Paper)	280	7	2.8

The average decision delay of the proposed improved PPO decision-making mechanism is only 280 ms, and the scaling deviation is ≤ 7%. Compared with rule-based decision-making, the delay is reduced by 72% and the deviation is reduced by 65%. Compared with basic PPO decision-making, the delay is reduced by 25% and the deviation is reduced by 30%, proving that its decision-making efficiency and accuracy are significantly improved.

Balance Effect of Cost Optimization and Performance Guarantee

The balance effect of cost optimization and performance guarantee under multi-scenarios is shown in Table 5.

Table 5. Balance Effect of Cost Optimization and Performance Guarantee Under Multi-Scenarios

Decision-Making Scheme	Cost Optimization Rate (%)	Performance Compliance Rate (%)	SLA Compliance Rate (%)
Rule-based Decision-Making	8.7	82.3	92.5
Integer Programming Optimization-based Decision-Making	14.2	88.7	96.1
Basic PPO Decision-Making	13.1	93.5	97.1
Improved PPO Decision-Making (Proposed in This Paper)	18.3	95.2	99.2

The proposed decision-making mechanism achieves an average cost optimization rate of 18.3% under the three scenarios, with a performance compliance rate of ≥ 95.2% and an SLA compliance rate of 99.2%. Compared with basic PPO decision-making, the cost optimization rate is increased by 5.2 percentage points and the SLA compliance rate is increased by 2.1 percentage points, realizing the multi-objective balance of cost, performance, and SLA.

Analysis of SLA Compliance Rate and System Stability

The 72-hour continuous operation experiment demonstrates that:

Under the proposed decision-making mechanism, the system's SLA compliance rate is stable at 99.2%-99.5% without service interruption.

While the SLA compliance rate of rule-based decision-making drops to 89.3% at the lowest, with 3 short service delays.

This improvement is attributed to the dynamic balancing strategy and the incorporation of disaster recovery redundancy, which improves system stability.

CONCLUSIONS AND PROSPECTS

Summary of Research Results

This study investigates AI-driven load prediction and elastic scaling for multi-cloud heterogeneous resources to optimize the high-frequency data processing requirements of intelligent textile manufacturing systems. By synchronizing computational power with the fluctuating workloads of automated weaving and dyeing processes, the proposed framework ensures operational efficiency and technical stability within the modern textile industrial landscape. The main results include:

(1) An improved xLSTM-based load prediction model is developed to achieve high-precision and real-time prediction under different scenarios through multi-source feature fusion and lightweight optimization.

(2) An improved PPO-based intelligent decision-making mechanism for elastic scaling is developed, in which a multi-objective optimization function and a dynamic balance strategy are constructed to achieve a balance among cost, performance, and SLA.

(3) A multi-cloud heterogeneous simulation platform is constructed to validate the effectiveness and superiority of the proposed framework through multi-scenario experiments.

Extraction of Research Innovations

The main contributions of this paper are as follows:

(1) Introduce an improved xLSTM and multi-model fusion strategy in the prediction model to improve heterogeneous data adaptability and the ability to capture burst loads.

(2) Design a hierarchical reward function and prioritized experience replay mechanism in the decision-making mechanism to improve multi-objective optimization effect and decision-making accuracy.

(3) Propose a multi-cloud resource adaptability evaluation method and dynamic balance strategy to solve the problems of heterogeneous resource scheduling and frequent scaling and scaling down.

Research Limitations and Future Prospects

Limitations of Existing Research

The existing research has two limitations:

(1) The experimental data mainly comes from simulation platforms and public datasets, which have certain differences from real complex multi-cloud environments.

(2) The model does not fully consider load prediction and scaling decisions in extreme fault scenarios (such as simultaneous failures of multiple clouds), and its robustness needs to be further improved.

Future Research Directions

Future research directions include:

(1) Integrate edge computing with multi-cloud architecture, study edge-cloud collaborative load prediction and scaling decision-making technology to reduce network delay.

(2) Design more robust prediction models and decision-making mechanisms for extreme loads and multi-cloud fault scenarios.

(3) Introducing federated learning technology solves the problem of multi-cloud data privacy protection and sharing for textile enterprises, allowing them to collaborate on sensitive production data and fiber design parameters without compromising proprietary trade secrets. This collaborative approach enhances the model generalization ability across diverse manufacturing environments, fostering industry-wide intelligence while maintaining the strict confidentiality of unique textile formulations and weaving techniques.

Author Contributions

Conceptualization –Chenglin Li, Yonghui Ren and Jing Bai; methodology – Chenglin Li, Yonghui Ren and Jing Bai; investigation – Chenglin Li, Yonghui Ren and Jing Bai; writing-original draft preparation – Chenglin Li, Yonghui Ren and Jing Bai. All authors have read and agreed to the published version of the manuscript.

Conflicts of Interest

The authors declare no conflict of interest.

Funding

This research received no external funding.

Acknowledgements

Not applicable.

REFERENCES

[1] Wang JX. A Monitoring and Alarm Method for Multi-Cloud Resources and Services. Journal of Shanghai Ship and Shipping Research Institute. 2022; 45(02):69-76+82. doi: 10.3969/j.issn.1674-5949.2022.02.011

[2] Gartner. Cloud Strategy Roadmap. Stamford, CT, USA: Gartner Research; 2025. Available from: https://www.gartner.com/en/documents/6601202

[3] Gartner. Gartner Releases Six Trends Shaping the Future of Cloud Technology. Stamford, CT, USA: Gartner Research; 2025. Available from: https://www.gartner.com/cn/newsroom/press-releases/2025-6-trends-cloud-future

[4] Liu J, Wang B, Wang C, Liu ZB. Research on Load Prediction and Elastic Scaling Scheme of Cloud Platform. Railway Computer Application. 2024; 33(2):63-66. doi: 10.3969/j.issn.1005-8451.2024.02.12

[5] Zhang M. Research and Implementation of Elastic Scheduling Algorithm for Cloud Services [dissertation]. Harbin, CN: Harbin Institute of Technology; 2017. doi: 10.7666/d.D01333651

[6] Zhao XH. Research on Resource Allocation Strategy of OpenStack Cloud Platform Based on Hybrid Prediction [dissertation]. Beijing, CN: North China University of Technology; 2025. doi: 10.26926/d.cnki.gbfgu.2025.000087

[7] Mao M, Humphrey M. Auto-scaling to Minimize Cost and Meet Application Deadlines in Cloud Workflows. In: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis; 12-18 November 2011; Seattle, Washington, USA. New York, NY, USA: Association for Computing Machinery; 2011. p. 1-12. doi:10.1145/2063384.2063449

[8] Li JY, Yang ZM, Song ZD, Zhu JL. Elastic Scaling Method for Kubernetes Container Cloud. Electronic Science and Technology. 2025; 38(3):32-39. doi: 10.16180/j.cnki.issn1007-7820.2025.03.005

[9] Tarmanini C, Sarma N, Gezegin C, Ozgonenel O. Short Term Load Forecasting Based on ARIMA and ANN Approaches. Energy Reports. 2023; 9:550-557. doi: 10.1016/j.egyr.2023.01.060

[10] Moreno-Vozmediano R, Montero RS, Huedo E, Llorente IM. Efficient Resource Provisioning for Elastic Cloud Services Based on Machine Learning Techniques. Journal of Cloud Computing. 2019; 8(1):1-18. doi:10.1186/s13677-019-0128-9

[11] Khan T, Tian W, Ilager S, Buyya R. Workload Forecasting and Energy State Estimation in Cloud Data Centres: ML-Centric Approach. Future Generation Computer Systems. 2022; 128:320-332. doi:10.1016/j.future.2021.10.019

[12] Li X, Wang H, Xiu P, Zhou X, Meng F. Resource Usage Prediction Based on BiLSTM-GRU Combination Model. 2022 IEEE International Conference on Joint Cloud Computing (JCC); 15–18 August 2022; Fremont, CA, USA. New York: IEEE; 2022. p. 9-16. doi:10.1109/JCC56315.2022.00009

[13] He YL, Mo PH, Fournier-Viger P, Huang ZX. A New Quality of Service-Oriented Spark Job Scheduler. Big Data Research. 2025; 11(4):154-177. doi:10.11959/j.issn.2096-0271.2025048

[14] Lu Y. Research on Optimized Provision of Cloud Resources for Large-Scale Service Systems in Public Cloud Environments [dissertation]. Harbin, CN: Harbin Institute of Technology; 2025. 75 p. doi: 10.27061/d.cnki.ghgdu.2025.003757

[15] Li XQ. Cloud Computing Resource Load Prediction Based on Bat Algorithm-Optimized SVM. Journal of Anyang Normal University. 2020;(2):24-29. doi: 10.16140/j.cnki.1671-5330.2020.02.006

[16] Li JC, Zhang YH. Database Load and Energy Consumption Prediction Method Based on Temporal Convolutional Network. Computer Simulation. 2025; 42(11):355-359+459.

[17] Zhou X. Cloud Computing Resource Load Prediction Method Based on GRU-LSTM Model. China New Telecommunications. 2024; 26(23):17-19.

[18] Su WJ. Research on Multi-Variable Time Series Prediction Method Based on xLSTM [dissertation]. Huainan, CN: Anhui University of Science and Technology; 2025. 74 p. doi: 10.26918/d.cnki.ghngc.2025.000955