Transactions on Computational and Scientific Methods | Vol. 4, No. 5, 2024
ISSN: 2998-8780
https://pspress.org/index.php/tcsm
Pinnacle Science Press

Collaborative Dual-Branch Contrastive Learning for Resource Usage Prediction in Microservice Systems

Guanzi Yao
Northwestern University, Evanston, USA
y19976122010@gmail.com

Abstract: This paper addresses the modeling challenges in resource usage prediction for microservice systems. It proposes a collaborative modeling method that combines a dual-branch structure with a contrastive learning mechanism. The method includes a local branch to model the temporal evolution of individual services and a global branch to capture collaborative dependencies among multiple services. A feature fusion module integrates these two types of representations, improving the model's ability to represent complex service behaviors. To enhance the discriminative power of feature representations, the model introduces a contrastive loss. This guides the feature encoding process to focus on semantic consistency and temporal distinctiveness. As a result, the model achieves better performance in collaborative modeling scenarios. The proposed method is systematically evaluated on a real-world microservice dataset. It is compared with several representative prediction models from recent years. The results show clear advantages in regression metrics such as MAE, RMSE, and R^2. In addition, this paper conducts ablation studies across several sensitivity dimensions, including hyperparameters, system topology complexity, data scale, and the proportion of heterogeneous services. These analyses further demonstrate the model's stability and robustness under various environmental conditions. Experimental results confirm that the proposed method effectively models both the temporal and collaborative patterns of microservice resource usage. It is suitable for dynamic resource awareness and management in high-complexity distributed systems.

Keywords: Collaborative modeling, contrastive learning, resource prediction, microservice system

1. Introduction

With the rapid development of cloud-native architecture, microservices have gradually replaced traditional monolithic systems and become the mainstream approach for building large-scale online services. In microservice architecture, system functions are decomposed into multiple autonomous modules. These services work together through lightweight protocols, enabling high availability, scalability, and flexible deployment[1]. However, the strong interdependencies and dynamic invocation chains between services also increase system complexity. This brings new challenges to performance tuning, resource management, and forecasting. Especially in multi-tenant and high-concurrency environments, resource usage fluctuates significantly. Traditional static resource allocation methods can no longer meet the dynamic needs of real-world applications[2].

Resource prediction in microservice systems is a core technology for achieving elastic scaling and performance assurance. Its accuracy directly affects service stability and resource efficiency. Unlike monolithic services, resource usage in microservices often exhibits "collaborative behavior," where the load of one service is influenced not only by its traffic but also by the states of upstream and downstream services.

Therefore, it is crucial to model the collaborative patterns among services and uncover their potential interdependencies. Traditional prediction methods based on single-point time series often ignore such collaboration. They fail to capture temporal collaborative behavior across service clusters, resulting in unstable performance and significant prediction errors in real systems[3].

Moreover, modern microservice systems generate high-dimensional, multi-source, and unstructured data. Simply stacking multi-dimensional inputs does not necessarily enhance the quality of representation. It may even introduce redundancy and noise. To achieve effective resource prediction, a representation mechanism capable of distinguishing between informative and non-informative features is needed. It must also support discriminative learning. In this context, contrastive learning has emerged as a promising approach to improve representation quality and generalization. By constructing positive and negative pairs, contrastive learning enables the model to learn a stable and discriminative representation space during training. This enhances the model's expressive power and robustness in resource prediction tasks.

Due to the heterogeneity among services in terms of resource usage patterns, traffic forms, and deployment strategies, a single-branch encoder architecture struggles to address both local feature extraction and global collaborative modeling. Therefore, designing a dual-branch architecture becomes essential. One branch focuses on modeling the historical resource usage of individual services. The other emphasizes extracting collaborative behavior across services. This design helps resolve the conflict between modeling granularity and collaborative understanding. By integrating contrastive learning into the training process, the model can simultaneously capture service-level evolution and system-level collaborative features more accurately[4].

In summary, resource prediction in microservice systems is not only a technical challenge but also a foundational task for intelligent scheduling and automated operations. Introducing contrastive learning and dual-branch architecture can significantly improve the discriminative power of feature representations. It also supports the development of high-performance prediction models tailored for complex microservice environments. Such work is essential for enhancing the intelligence of resource management and provides key support for building stable, efficient, and adaptive cloud-native infrastructures.

2. Related work

With the widespread adoption of microservice architecture in cloud computing environments, the increasing granularity of services has made dynamic resource management more complex[5]. To address this challenge, extensive research has focused on resource utilization prediction, especially in scenarios such as container orchestration, elastic scaling, and service quality assurance. Prediction models have been widely used to support decision-making in these contexts. Early approaches often relied on statistical methods based on historical time series, such as ARIMA or moving average techniques. Although these methods can capture certain trends, they often suffer from delayed response and high prediction error when dealing with the frequent fluctuations and high-dimensional, heterogeneous patterns of resource usage in microservice systems. As a result, they fall short in handling complex prediction tasks[6].

To further improve model performance, deep learning models have been increasingly introduced into resource prediction tasks. Representative architectures include convolutional neural networks for short-term pattern extraction, recurrent neural networks for time series modeling, and Transformer-based models for long-range dependency learning. These methods have improved prediction accuracy and generalization to some extent. However, most of them still treat microservices as independent units, ignoring the complex interactions and co-evolution among services. In reality, strong upstream and downstream dependencies often exist between services. Modeling a single service in isolation cannot fully capture its contextual role in the system, which negatively affects prediction accuracy[7].

In recent years, some studies have begun to consider the system-level perspective by incorporating collaborative behaviors among services. Graph neural networks and attention mechanisms have been used to model service topology and contextual influence. These methods focus on embedding collaborative

information during the feature learning stage. This helps alleviate the information loss caused by independent service modeling. However, two core challenges remain unresolved in collaborative modeling. First, inter-service collaboration is highly dynamic and diverse, making it difficult to capture through fixed structures. Second, in highly heterogeneous systems, distinguishing between informative collaborative features and redundant ones is still a key bottleneck for improving model performance.

At the same time, contrastive learning has emerged as a powerful unsupervised representation learning technique, showing strong discriminative ability in time series forecasting and system modeling. Its core idea is to construct positive and negative sample pairs to guide the model in learning a robust and semantically consistent representation space. However, existing contrastive learning approaches are mostly applied to image data or unimodal time series. They lack systematic design when applied to microservice systems, which are characterized by high-dimensional heterogeneity and multi-level interactions. Therefore, introducing contrastive learning into microservice resource prediction, and designing a dual-branch structure to decouple "individual service features" from "collaborative contextual features," is a promising direction for improving model accuracy and generalization.

3. Method

This study proposes a microservice collaborative resource prediction model that introduces a dual-branch contrastive learning mechanism, aiming to simultaneously capture the temporal evolution characteristics of the service itself and its collaborative behavior with other services in the system. The overall architecture consists of two main feature encoding branches, which process local single-service information and global multi-service context respectively. The detailed structure of the proposed model is illustrated in Figure 1.

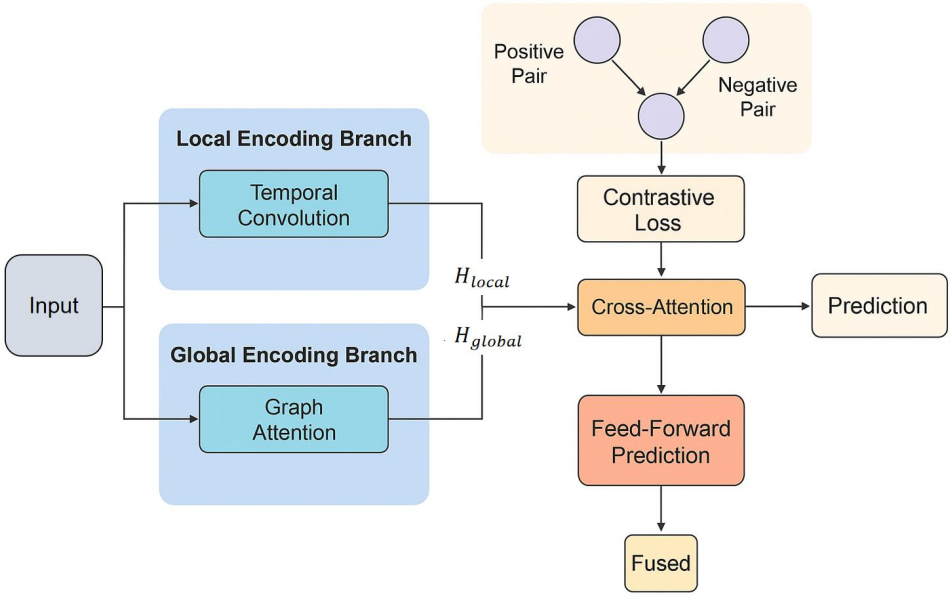


Figure 1. Overall model architecture diagram

The model input is a tensor containing historical resource usage information, set as $X \in R^{N \times T \times D}$, where N represents the number of services, T represents the time step length, and D represents the feature dimension of each time point (such as CPU, memory, etc.). The local branch extracts the short-term resource evolution characteristics of each service through a temporal convolution module, and the output is represented as $H_{local} \in R^{N \times T' \times d}$, while the collaborative branch uses a graph attention network or a structured modeling

method based on the service dependency matrix to output the collaborative context representation $H_{global} \in R^{N \times T' \times d}$.

In the feature integration stage, we use a cross-branch attention mechanism for fusion, so that the model can automatically learn the importance of local and global features. The integrated representation can be written as:

$$H_{fused} = \alpha \cdot H_{local} + (1 - \alpha) \cdot H_{global}$$

Where $\alpha \in [0,1]$ is a learnable gating coefficient that represents the weighted degree of local and collaborative information. Subsequently, the fused features are passed through a feedforward prediction network to output the resource usage prediction value $Y' \in R^{N \times T_{future}}$ for the future time step, which is defined as follows:

$$L_{contrast} = -\log \frac{\exp(sim(z_i, z_j)/\tau)}{\sum_{k=1}^{K} \exp(sim(z_i, z_k)/\tau)}$$

Where z_i, z_j represents the embedding vector of the positive sample pair, τ is the temperature coefficient, K is the number of negative samples, and $sim(\cdot,\cdot)$ represents the cosine similarity function.

The final training goal is to minimize the weighted combination of prediction error and contrast loss. The overall loss function is defined as follows:

$$L_{total} = L_{reg} + \lambda \cdot L_{contrast}$$

Where $L_{reg} = \| Y' - Y \|_2^2$ represents the mean square error between the predicted target and the actual resource usage value, and B is the contrast loss weight, which is used to regulate the balance between representation learning and prediction accuracy.

The original intention of designing this model architecture is to separate and model the local behavior and global coordination information of microservices in a structured way, using a dual-branch design to improve representation capabilities, while introducing contrastive learning to enhance the distinguishability between features. Through an end-to-end training process, the model can learn the highly coupled and time-varying resource usage rules within the microservice system, providing stable prediction support for subsequent resource scheduling and service governance.

4. Experimental Results

4.1 Dataset

The experimental dataset used in this study is the Alibaba Cluster Trace 2018. This dataset is collected from a real large-scale production environment and records the resource scheduling and runtime status of Alibaba's online services on a cloud platform. It covers multiple typical microservice tasks and includes multi-dimensional resource metrics such as CPU usage, memory consumption, and request rate. The dataset is representative and complex, reflecting the dynamic nature of resource usage under microservice architecture.

The dataset contains runtime traces of more than 4,000 physical machines and hundreds of thousands of container tasks. It provides information on resource requests, actual usage, and system allocation records across different periods. The data is recorded in time series format with intervals ranging from 5 seconds to 1 minute. It is well-suited for tasks such as resource prediction, load modeling, and service management.

The dataset spans a long period and includes diverse load types, covering typical usage patterns under multi-tenant and multi-service scenarios.

In this study, we use container-level resource usage data to construct service-level collaborative representation sequences. We perform regression modeling on the resource trends of each service within each time window. By leveraging this real-world large-scale microservice dataset, we can effectively validate the predictive performance and stability of the proposed model under high-dimensional and heterogeneous environments. This also enhances the practical relevance and applicability of the research findings in industrial settings.

4.2 Experimental setup

All experiments in this study were conducted on a high-performance computing server. The hardware configuration includes two NVIDIA A100 GPUs with 40GB memory each, dual Intel Xeon Gold 6338 CPUs with a total of 64 cores, 512GB DDR4 memory, and a 2TB NVMe SSD. The operating system is Ubuntu 22.04 LTS. The deep learning framework used is PyTorch 2.1, with Python version 3.10. Additional dependencies include DGL 1.1, NumPy, Pandas, and Matplotlib. This setup ensures high computational efficiency and stability for large-scale graph neural networks and contrastive learning tasks.

During the training phase, all models were evaluated under the same data-splitting strategy. The input time window was set to 60 steps, and the prediction horizon was set to 10 steps. The batch size was fixed at 128. The optimizer used was Adam, and the initial learning rate was set to 0.001 for all models. Early stopping was applied to prevent overfitting, and performance metrics on the validation set were recorded at each epoch. All experiments were executed in single-task processes to avoid resource contention. This ensures fair comparisons across models and guarantees reproducibility.

4.3 Experimental Results

This paper first conducts a comparative experiment, and the experimental results are shown in Table 1.

Table 1: Comparative experimental results

Method	**MAE**	**RMSE**	**R^2**
Ours	0.038	0.071	0.926
Autoformer[8]	0.051	0.093	0.881
Informer[9]	0.057	0.101	0.867
GTS[10]	0.049	0.088	0.892
TimesNet[11]	0.043	0.079	0.911

The comparison results in the table show that the proposed model demonstrates significant advantages in microservice resource prediction tasks. It outperforms existing mainstream methods across all three major evaluation metrics: MAE, RMSE, and R^2. The MAE and RMSE values are 0.038 and 0.071, respectively, which are substantially lower than those of the baseline models. This indicates that the model performs better in controlling prediction errors and fitting overall trends. It captures the temporal dynamics of service resource usage more accurately and enhances the ability to perceive behavioral patterns in complex microservice systems.

Compared with models such as Informer and Autoformer, which have relatively simple structures, the proposed model shows a significant improvement in the R^2 metric, reaching 0.926. This suggests a stronger capacity to explain the variance in resource changes. The improvement is mainly attributed to the model's

architectural design for capturing collaborative behavior among microservices. By employing a dual-branch mechanism to separately process local temporal features and global dependencies, the model addresses the limitations of traditional methods that treat services in isolation and lack contextual understanding.

In addition, when compared with recently proposed structure-enhanced models such as GTS and TimesNet, the proposed method still leads in all evaluation metrics, reflecting strong generalization ability. This demonstrates the critical role of the contrastive learning mechanism in the feature encoding stage. By constructing positive and negative sample pairs, the model learns a discriminative and temporally consistent representation space. This improves its ability to fit dynamic resource patterns and enhances robustness.

Overall, the proposed model achieves a strong balance between accuracy and stability. It is particularly suitable for microservice systems with complex resource usage patterns and tight inter-service dependencies. Through joint optimization of architectural design and training strategy, the model not only improves predictive performance but also exhibits greater adaptability to system variations. This provides a solid data-driven foundation for subsequent resource scheduling and service management.

This paper also gives the impact of different contrast loss weights on model performance, and the experimental results are shown in Figure 2.

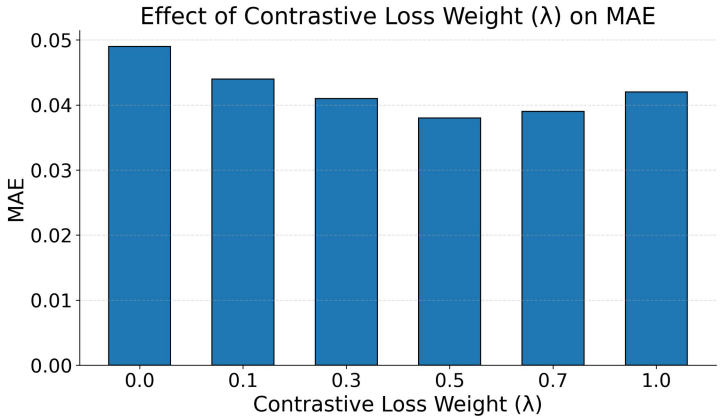


Figure 2. The impact of different contrast loss weights on model performance

The results in the figure show that the weight of the contrastive loss has a significant impact on model performance. When the contrastive loss is not introduced ($\lambda = 0.0$), the model reaches the highest MAE. This indicates that relying only on traditional supervised signals fails to constrain the structure of the representation space. The model struggles to learn discriminative feature representations, which negatively affects the accuracy of resource prediction.

As the contrastive loss weight increases to $\lambda = 0.5$, the model performance improves continuously. The MAE reaches its lowest value. This suggests that a moderate contrastive learning signal can effectively guide the model to distinguish fine-grained features between services during representation learning. It helps the model capture the collaboration between local and global information. This confirms the effectiveness of introducing semantic discrimination mechanisms in microservice scenarios, especially when service dependencies are complex and temporal patterns vary frequently.

However, when the weight increases further to $\lambda = 0.7$ and $\lambda = 1.0$, the MAE slightly increases. This indicates that an overly strong contrastive signal may overshadow the main regression objective. The feature encoding process may shift away from the prediction task. The model becomes more focused on the distance structure of representations instead of the target variable itself, leading to increased prediction error. This shows that the balance between contrastive loss and task loss is a key factor in model performance.

Overall, the experiment confirms the sensitivity of model performance to the contrastive loss weight. It shows that introducing a discriminative learning mechanism can enhance model performance in microservice resource prediction. Properly tuning the hyperparameter λ helps achieve a balance between representation learning and the prediction task. This improves generalization and stability, providing more reliable decision support for resource management in complex service environments.

This paper also gives an analysis of the performance changes of the model under the change of microservice topology complexity, and the experimental results are shown in Figure 3.

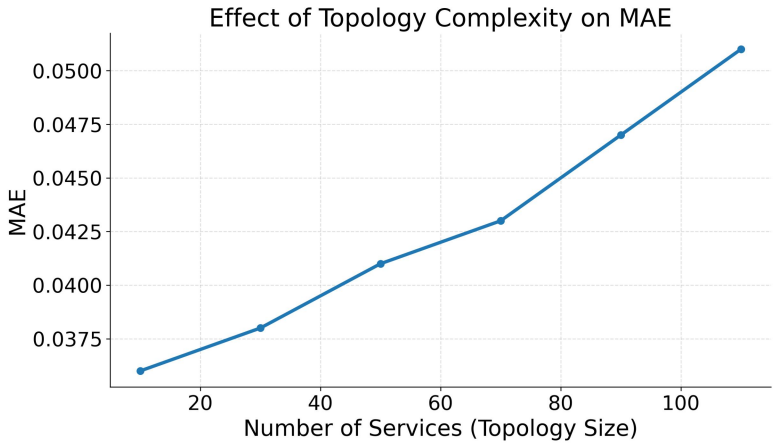


Figure 3. Analysis of model performance changes under changes in microservice topology complexity

The figure shows that the complexity of the microservice topology has a clear impact on model performance. As the number of services increases, the overall service dependencies in the system become more dense. This adds a modeling burden when capturing collaborative features between services, causing the prediction error (MAE) to rise gradually. This result indicates that as the system topology scales up, the model faces greater challenges in representing dynamic interactions and information transmission paths among services.

When the number of service nodes is small, the model can effectively extract temporal features of individual services and simple collaboration patterns. In this case, the MAE remains at a low level. This suggests that the dual-branch structure has a strong feature adaptation ability for small-scale topologies. However, as the number of services increases, the possible interaction paths grow rapidly. This expands the collaborative representation space and increases the model's sensitivity to irrelevant features, which in turn affects the overall regression accuracy.

By comparing trends across different topology scales, it can be observed that although the model uses a graph attention mechanism to capture global context, there is still a representation bottleneck in high-complexity scenarios. In particular, when the number of services exceeds 90, the prediction error rises sharply. This suggests that collaborative modeling capacity is limited by information overload or feature interference. It reflects the need for stronger structure-aware mechanisms or feature selection strategies to enhance model robustness in dense microservice environments.

Overall, this experiment confirms the model's sensitivity to topological complexity and reveals the performance boundary of collaborative modeling as scale increases. The results suggest that for large-scale microservice systems, future work could consider hierarchical modeling, modular design, or sparse graph optimization to further improve the model's adaptability to complex topologies.

This paper also presents a study on the impact of changes in the amount of training data on model performance, and the experimental results are shown in Figure 4.

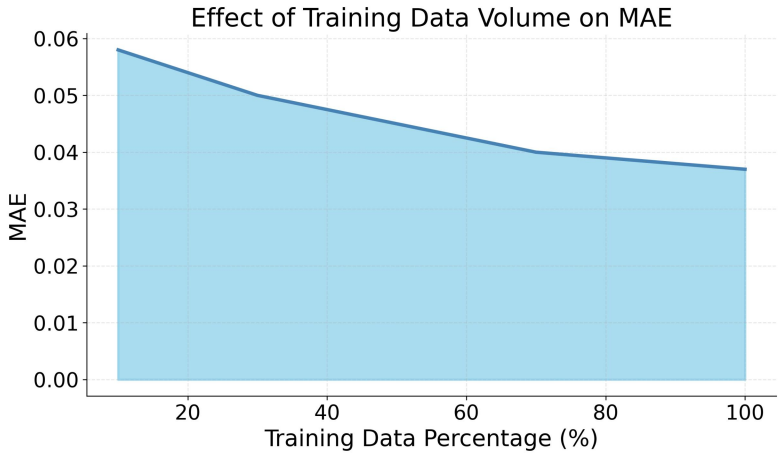


Figure 4. Study on the impact of changes in training data volume on model performance

The figure shows that the amount of training data has a clear impact on model performance. As the proportion of training samples increases, the prediction error (MAE) in the resource forecasting task consistently decreases. This result indicates that the proposed dual-branch contrastive learning structure performs better when sufficient data is available. It effectively captures both the temporal evolution patterns of services and their collaborative structural information, thereby improving overall prediction accuracy.

When only 10% of the total data is used for training, the prediction error remains relatively high. This suggests that a small dataset is insufficient to support comprehensive learning of complex microservice collaboration patterns. The resulting feature representations are limited, which reduces the model's regression capacity. However, as the training data size increases, the model gradually builds stable representations of service dependencies. When the training data exceeds 70%, performance improvements begin to plateau, reflecting the saturation of the model's learning capacity.

This experiment also reveals the sensitivity of contrastive learning to data volume. With limited training data, the space for constructing effective positive and negative pairs is small. This weakens the representational power of the contrastive loss and limits the model's ability to learn a highly discriminative embedding space. In contrast, with sufficient data, the model can optimize both the main regression objective and the contrastive objective. It can extract meaningful contextual differences from diverse samples, enhancing feature robustness and generalization.

In summary, the proposed model shows stronger performance and training stability when applied to medium or large-scale datasets. This suggests that the model design is well suited for real-world microservice environments with large volumes of heterogeneous logs and monitoring data. For deployment scenarios with limited data, pretraining or transfer learning strategies may be considered to compensate for the limitations of small-sample learning.

This paper also gives the impact of the proportion of heterogeneous services on the collaborative modeling effect, and the experimental results are shown in Figure 5.

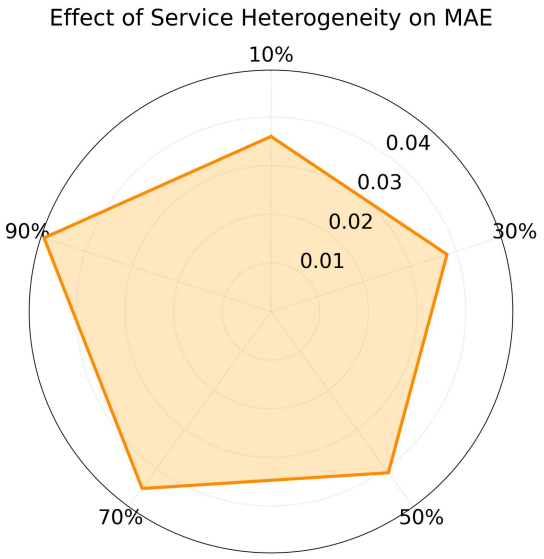


Figure 5. The impact of heterogeneous service ratio on collaborative modeling effectiveness

The figure shows that the proportion of heterogeneous services has a significant impact on the effectiveness of collaborative modeling. As the percentage of heterogeneous services in the system increases, the MAE in the prediction task also shows an overall rising trend. This indicates that greater service diversity disrupts the model's ability to stably learn collaborative features. When the proportion exceeds 70%, the error increases rapidly. This suggests that in scenarios with large differences in service functions, call paths, and load patterns, the robustness of the model's global representation becomes challenged.

When the heterogeneity ratio is low, such as between 10% and 30%, the service cluster is relatively homogeneous. The collaborative modeling mechanism can extract stable contextual representations based on structural commonalities among services. As a result, the MAE remains low. This structural consistency enhances alignment during the dual-branch fusion stage. The coordination between local and global features becomes tighter, which leads to more accurate prediction outputs.

As heterogeneity increases to 50% – 70%, the model begins to face difficulties caused by feature distribution differences. In this stage, service behavior becomes more diverse. Some services may introduce non-standard dependency paths or abnormal load patterns. This adds noise to the collaborative modeling module during feature aggregation. The ability to understand upstream and downstream relationships is affected.

Overall, the experimental results reveal the performance boundaries of collaborative modeling in heterogeneous environments. They also highlight the importance of structure-aware mechanisms and feature selection strategies. In highly heterogeneous scenarios, relying solely on a unified modeling path may lead to information conflicts. Future work may explore strategies such as cluster-based modeling or adaptive feature gating to improve the model's adaptability to diverse microservice systems.

5. Conclusion

This paper presents a collaborative modeling framework for resource prediction in microservice systems. The proposed model integrates a dual-branch structure with a contrastive learning mechanism. The local branch captures the temporal evolution of individual services, while the global branch models dependencies between services. In the fusion stage, contrastive learning is introduced to enhance feature discriminability. This design addresses limitations in traditional methods, such as insufficient collaborative modeling and weak contextual awareness. Extensive experimental results show that the method achieves strong robustness and

high prediction accuracy under various sensitivity conditions. It demonstrates good adaptability in heterogeneous and high-complexity microservice scenarios.

In key applications such as microservice resource scheduling, elastic scaling, and service management, accurate resource prediction models can significantly improve resource utilization. They also reduce service latency and system energy consumption. The proposed method features strong structural scalability and high training stability. It can serve as a foundational modeling component in large-scale service systems and provide solid support for automated operation strategies. In addition, the model's structural design offers new insights for tasks involving multi-source feature fusion and context-aware prediction. It has strong potential for transfer and practical deployment.

This work also includes a systematic performance analysis across multiple dimensions, including hyperparameters, system topology, data scale, and service heterogeneity. These experiments further validate the model's ability to operate in complex environments. The results support the rationality of the model design and offer a reference framework for future studies. In particular, in cloud-native environments with frequent resource fluctuations and high concurrency, the model provides a methodological foundation for intelligent scheduling and adaptive resource allocation.

Future research may explore two main directions. One is to further enhance the model architecture by incorporating structure-aware modules, such as hierarchical attention mechanisms or dynamic graph representations, to address the challenges of highly dynamic service topologies. The other is to investigate unsupervised pretraining and transfer learning strategies to improve performance in low-data or cold-start scenarios. Integrating the proposed method into real-world cloud scheduling systems and aligning it with policy-level optimization will also be an important step toward practical application.

References

[1] Luo S, Xu H, Ye K, et al. The power of prediction: microservice auto scaling via workload learning[C]//Proceedings of the 13th Symposium on Cloud Computing. 2022: 355-369.

[2] He H, Su L, Ye K. GraphGRU: A graph neural network model for resource prediction in microservice cluster[C]//2022 IEEE 28th International Conference on Parallel and Distributed Systems (ICPADS). IEEE, 2023: 499-506.

[3] Zhang Y, Hua W, Zhou Z, et al. Sinan: ML-based and QoS-aware resource management for cloud microservices[C]//Proceedings of the 26th ACM international conference on architectural support for programming languages and operating systems. 2021: 167-181.

[4] Al Qassem L M, Stouraitis T, Damiani E, et al. Proactive random-forest autoscaler for microservice resource allocation[J]. IEEE Access, 2023, 11: 2570-2585.

[5] Park J, Choi B, Lee C, et al. GRAF: A graph neural network based proactive resource allocation framework for SLO-oriented microservices[C]//Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies. 2021: 154-167.

[6] Chow K H, Deshpande U, Seshadri S, et al. Deeprest: deep resource estimation for interactive microservices[C]//Proceedings of the Seventeenth European Conference on Computer Systems. 2022: 181-198.

[7] Deng J, Li B, Wang J, et al. Microservice pre-deployment based on mobility prediction and service composition in edge[C]//2021 IEEE International Conference on Web Services (ICWS). IEEE, 2021: 569-578.

[8] Wu H, Xu J, Wang J, et al. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting[J]. Advances in neural information processing systems, 2021, 34: 22419-22430.

[9] Zhou H, Zhang S, Peng J, et al. Informer: Beyond efficient transformer for long sequence time-series forecasting[C]//Proceedings of the AAAI conference on artificial intelligence. 2021, 35(12): 11106-11115.

[10] Chen H, Eldardiry H. Graph time-series modeling in deep learning: a survey[J]. ACM Transactions on Knowledge Discovery from Data, 2024, 18(5): 1-35.

[11] Wu H, Hu T, Liu Y, et al. Timesnet: Temporal 2d-variation modeling for general time series analysis[J]. arXiv preprint arXiv:2210.02186, 2022.