Deep Temporal Convolutional Neural Networks with Attention Mechanisms for Resource Contention Classification in Cloud Computing

Ning Lyu , Feng Chen , Chong Zhang , Chihui Shao , Junjie Jiang *

Posted Date: 17 December 2025

doi: 10.20944/preprints202512.1556.v1

Keywords: resource contention classification; multi-scale convolution; attention mechanism; temporal feature modeling

Article

Deep Temporal Convolutional Neural Networks with Attention Mechanisms for Resource Contention Classification in Cloud Computing

Ning Lyu [1], Feng Chen [2], Chong Zhang [1], Chihui Shao [3] and Junjie Jiang [4,*]

[1] Carnegie Mellon University; Pittsburgh, USA
[2] Northeastern University; Seattle, USA
[3] Duke University; Durham, USA
[4] Illinois Institute of Technology; Chicago, USA
* Correspondence: junjiejiang1112@gmail.com

Abstract

This paper addresses the challenge of efficiently identifying and classifying resource contention behaviors in cloud computing environments. It proposes a deep neural network method based on multi-scale temporal modeling and attention-based feature enhancement. The method takes time series resource monitoring data as input. It first applies a Multi-Scale Dilated Convolution (MSDC) module to extract features from resource usage patterns at different temporal resolutions. This allows the model to capture the multi-stage dynamic evolution of resource contention behaviors. An Attention-based Feature Weighting (AFW) module is then introduced. It learns attention weights along both the temporal and feature dimensions. This enables the model to emphasize key time segments and core resource metrics through saliency modeling and feature enhancement. The overall architecture supports end-to-end modeling. It can automatically learn temporal patterns of resource contention without relying on manual feature engineering. To evaluate the effectiveness of the proposed method, this study constructs a range of contention scenarios based on real-world cloud platform data. The model is assessed under different structural configurations and task conditions. The results show that the proposed model outperforms existing mainstream temporal classification models across multiple metrics, including accuracy, recall, F1-score, and AUC. It demonstrates strong feature representation and classification capabilities, especially in handling high-dimensional, multi-source, and dynamic data. The proposed approach offers practical support for resource contention detection, scheduling optimization, and operational management in cloud platforms.

CCS Concepts: Computing methodologies; Machine learning; Machine learning approaches

Keywords: resource contention classification; multi-scale convolution; attention mechanism; temporal feature modeling

1. Introduction

Cloud computing serves as the core platform for modern information technology. It handles massive computing and storage tasks. The efficiency of resource scheduling and management directly affects system stability and performance. As cloud infrastructure continues to scale, the dynamic allocation of resources and the coexistence of multiple tenants become increasingly complex [1]. Resource contention has emerged as a key factor affecting service quality. In real-world operations, multiple virtual machines or container instances often compete for shared hardware resources such as CPU, memory, I/O, and network bandwidth. This contention leads to degraded task performance, increased system latency, and even service outages. Therefore, timely and accurate

 doi:10.20944/preprints202512.1556.v1

identification and classification of resource contention are critical to ensuring predictable performance and system stability[2].

Traditional resource management mechanisms rely mainly on static thresholds or rule-based strategies[3]. These approaches often lack flexibility and generalization when dealing with complex and dynamic contention scenarios. Resource contention in cloud platforms manifests in various forms. Its behavioral patterns often show temporal, periodic and burst characteristics. Static-feature-based analysis methods struggle to capture the evolving trends of resource usage effectively. With the widespread adoption of cloud-native architectures, resource states have become more transient and fragmented. This increases the temporal and spatial nonlinearity and complexity of contention detection tasks. To address these challenges, there is a pressing need for intelligent analysis methods that can model time series and extract deep features. These methods can improve both the accuracy and robustness of contention identification[4].

In recent years, artificial intelligence, especially deep learning, has shown strong performance in handling complex time series data [5-8]. Temporal Convolutional Networks (TCNs), as a novel sequence modeling architecture, perform well in overcoming training difficulties and gradient issues found in traditional recurrent neural networks. With causal and dilated convolutions[9], TCNs capture long-range dependencies without increasing computational complexity. This makes them suitable for modeling the dynamic evolution of resource states in cloud platforms[10]. The parallelism of TCNs also aligns with the cloud environment's demand for efficient model inference. It enables fast and real-time classification of resource contention. Introducing TCNs into resource contention detection follows the trend of intelligent operations and maintenance. It also supports the development of efficient and scalable resource management strategies.

In cloud environments, resource contention reflects not only system-level performance anomalies but also the complex coupling among scheduling policies, workload characteristics, and hardware capabilities. Accurate classification of different contention types helps enable fine-grained scheduling, intelligent fault prediction, and adaptive performance optimization. For example, CPU saturation, disk I/O bottlenecks, and network congestion may appear similar but affect applications differently. Extracting key temporal features from monitoring data and applying deep modeling can enhance platform insights and resource efficiency. Moreover, resource contention detection can support multi-tenant security analysis. It helps identify malicious workloads and potential resource abuse[11].

Overall, resource contention has become a major operational challenge in cloud platforms. Its hidden, bursty, and diverse nature demands highly accurate and robust identification methods. Classification based on TCNs provides strong temporal modeling and end-to-end feature learning capabilities. This approach overcomes limitations in traditional feature engineering and pattern generalization. By adopting this advanced architecture, we can improve contention recognition accuracy and lay the foundation for intelligent and autonomous cloud resource management. This study responds to real operational needs. It integrates deep temporal models with cloud operations scenarios and offers new ideas and methods for resource contention analysis. It contributes to enhancing the reliability and efficiency of cloud platform operations.

2. Related Work

Research on resource contention has proposed a variety of mechanisms to incorporate contention signals into scheduling and resource management. Workflow scheduling approaches incorporate endpoint communication contention into the scheduling objective, making decisions based on modeled interference between tasks and shared communication resources [12]. Dynamic architectures that monitor interference and adapt task placement have been developed to adjust scheduling policies in response to observed contention, demonstrating the benefit of explicitly modeling interference as part of the scheduling loop [13]. At a broader systems level, dynamic offloading strategies with adaptive weights have been introduced to mitigate resource contention across distributed nodes, where the offloading decision is formulated as an optimization problem

 doi:10.20944/preprints202512.1556.v1

influenced by measured contention indicators [14]. Beyond scheduling, contention itself has been used as a discriminative signal: by extracting low-level resource usage features and modeling their joint distribution, it is possible to classify co-resident programs based on the contention patterns they induce, illustrating that contention metrics can form a useful feature space for classification tasks [15]. Structural techniques such as service separation can be regarded as another way of manipulating the feature space of contention, by partitioning workloads and resources so that contention patterns become more predictable and controllable [16]. These works highlight the importance of measuring and exploiting contention-related features, but they generally rely on manually designed indicators, heuristic policies, or shallow models, and do not fully leverage end-to-end deep temporal feature learning.

Deep learning methods for time series provide a methodological foundation for modeling the dynamic evolution of resource usage. Architectures that combine frequency-domain representations with attention mechanisms show that jointly modeling temporal and spectral information and then learning attention weights over these representations leads to more accurate prediction of complex time-varying signals [17]. Attention-based deep models for multivariate time series forecasting further demonstrate that learning to weight different time steps and feature channels can effectively capture long-range dependencies, structural breaks, and heterogeneous dynamics in high-dimensional sequences [18]. Complementary to direct sequence modeling, approaches that transform multidimensional time series into interpretable event sequences provide a way to re-encode continuous temporal dynamics into structured event representations, which can then be processed by downstream models for pattern mining and classification [19]. These methods collectively motivate the design in this paper: using multi-scale temporal convolutions to capture patterns at different horizons, and attention mechanisms along temporal and feature dimensions to assign higher weights to salient segments and key resource metrics.

Another line of work focuses on structured representations and relational dependencies, which is highly relevant for modeling complex, multi-source monitoring data. Methods that integrate knowledge graph reasoning with pretrained neural encoders show how to fuse relational structure with learned representations to improve anomaly detection in structured data, typically by propagating information along graph edges and aligning symbolic relations with latent features [20]. Self-supervised graph neural network frameworks on heterogeneous information networks design pretext tasks to learn rich node and edge embeddings without extensive labels, thus enhancing feature extraction in scenarios with multiple entity types and relations [21]. Graph neural network–based classification models further illustrate how message passing and neighborhood aggregation can capture high-order interactions among features and entities in classification tasks [22]. From a methodological perspective, these works demonstrate the value of learning structured, high-level representations from multi-source signals. The approach in this paper can be seen as complementary: instead of explicitly constructing a graph, it captures structured dependencies along temporal and feature axes through multi-scale convolutions and attention, yielding a rich representation for resource contention patterns.

At the architectural level, recent work emphasizes modularity, adaptivity, and efficient integration of external knowledge. Selective knowledge injection via adapter modules introduces compact trainable components into large models, allowing external information to be incorporated in a parameter-efficient and controllable manner without fully fine-tuning the backbone network [23]. In parallel, research on the synergy between deep learning and neural architecture search shows that automatically exploring architectures—such as varying depth, width, kernel sizes, and connectivity patterns—can yield structures better matched to specific tasks than manually designed networks [24]. These directions suggest that specialized modules and principled architecture design can significantly improve model expressiveness and efficiency. The Multi-Scale Dilated Convolution and Attention-based Feature Weighting modules in this paper follow the same methodological philosophy: they are tailored building blocks designed to capture multi-stage temporal evolution and salient feature dimensions in resource monitoring data. By integrating these modules into an end-to-

 doi:10.20944/preprints202512.1556.v1

end model, this work extends prior contention-aware and deep temporal modeling approaches and provides a unified framework for high-dimensional resource contention classification.

3. Method

In this study, we apply a Temporal Convolutional Network (TCN)-based method to classify resource contention behaviors in cloud computing, aiming for precise identification of multiple contention types and phases. Building on the principles of scalable and communication-efficient intelligence described by Liu, Kang, and Liu [25], the model is tailored for practical deployment in large cloud environments. Specifically, we apply a Multi-Scale Dilated Convolution (MSDC) module to the TCN backbone, enabling the network to extract resource usage patterns at different temporal resolutions—a technique shown by Chen et al. [26] to be effective in capturing complex, multi-stage anomalies. To further distinguish salient contention characteristics, we employ an Attention-based Feature Weighting (AFW) module, which jointly learns to highlight important time intervals and resource dimensions. This approach is informed by Hu et al.'s [27] work on structurally emphasizing key components within microservice routing networks. The architecture of the overall model is illustrated in Figure 1.

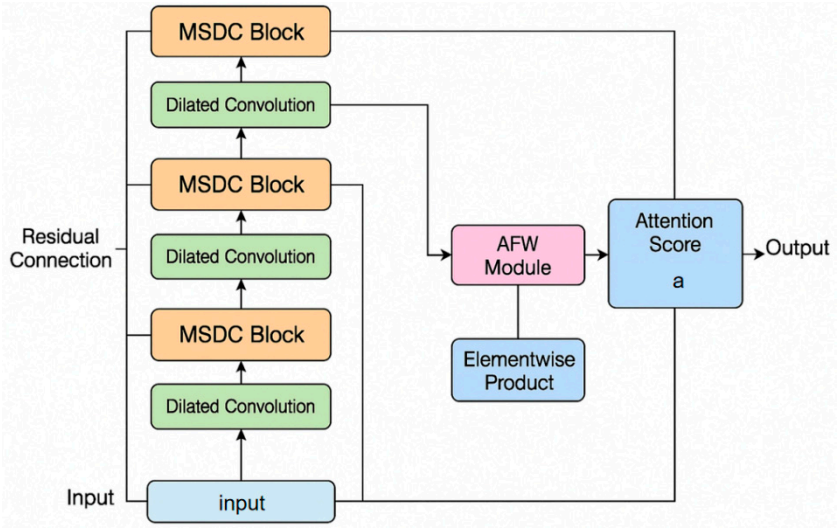

Figure 1. Overall model architecture diagram.

3.1. Multi-Scale Dilated Convolution

In order to effectively capture the dynamic characteristics of resource contention behavior at different time scales in cloud computing environments, this paper designs and introduces a multi-scale dilated convolution module (MSDC) as one of the core components of the model. Resource contention often involves both short-term bursts and long-term evolving trends, making it necessary for the model to handle features across multiple temporal resolutions. The MSDC module addresses this challenge by enabling the model to process input sequences with varying temporal contexts, thus supporting a more comprehensive representation of time-dependent behaviors.

The module operates by applying multiple parallel convolutional branches, each configured with a distinct dilation rate. These dilation rates determine the spacing between kernel elements, effectively adjusting the temporal receptive field of each branch. By aggregating the outputs from all branches, the MSDC module captures features from both fine-grained and coarse-grained time scales. This architectural design enhances the model's flexibility in identifying diverse patterns within multivariate time series data. The detailed structure and flow of operations within the MSDC module are illustrated in Figure 2.

 doi:10.20944/preprints202512.1556.v1

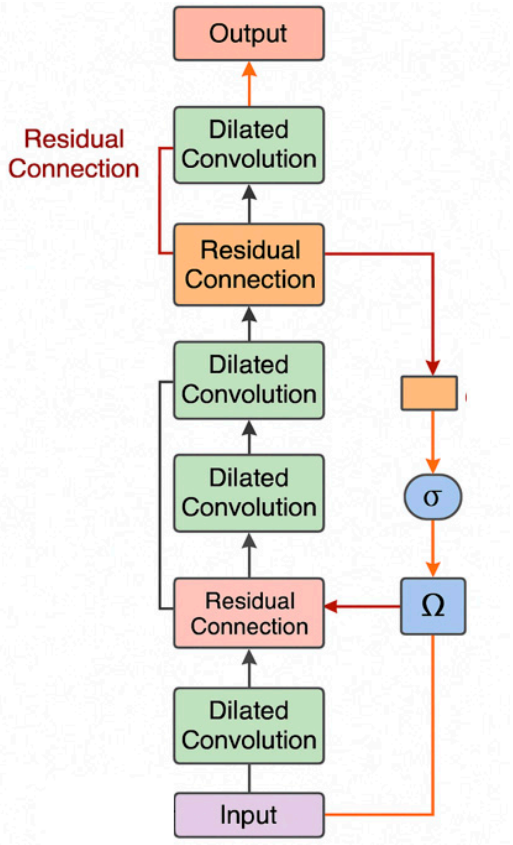

Figure 2. MSDC module architecture.

For any one-dimensional input sequence $X = [x_1, x_2, ..., x_T]$, its output features in the dilated convolution can be expressed as:

$$y_t^{(d)} = \sum_{i=0}^{k-1} w_i \cdot x_{t-d \cdot i}$$

Among them, $y_t^{(d)}$ represents the output at time t when the expansion rate is d, k is the convolution kernel size, w_i is the i-th convolution kernel weight, and $x_{t-d \cdot i}$ represents the input position after expansion sampling.

To achieve multi-scale modeling, the MSDC module constructs multiple parallel paths, each of which sets a different hole rate $d \in \{d_1, d_2, ..., d_m\}$, and generates a set of output sequences $\{Y(d_1), Y(d_2), ..., Y(d_m)\}$ with different temporal receptive fields. The multi-scale outputs are then integrated into a unified feature representation through a cascade or weighted fusion strategy:

$$Y_{MSDC} = Concat(Y(d_1), Y(d_2), ..., Y(d_m))$$

Or

$$Y_{MSDC} = \sum_{j=1}^{m} \alpha_j \cdot Y(d_j)$$

where α_j is a trainable weight parameter, which indicates the importance of features at different scales.

To further improve the model depth and nonlinear expression ability, each MSDC branch uses a residual connection mechanism, so that the convolution output is added to its input to form a residual path, thereby enhancing training stability and alleviating the gradient vanishing problem. This process can be formally expressed as:

 doi:10.20944/preprints202512.1556.v1

$$\widetilde{Y}^{(d_j)} = \sigma(Y^{(d_j)} + X)$$

where $\sigma(\cdot)$ represents a nonlinear activation function, such as ReLU or GELU, and X is the input sequence. The final output is synthesized by the residual path features of each scale for subsequent attention-weighting module processing.

The MSDC module not only significantly expands the model's time perception range, but also improves the ability to distinguish different contention behavior forms through structural hierarchy and feature fusion mechanisms. Its design fully considers the diversity and nonlinear characteristics of resource indicators changing over time, enabling the model to achieve more sophisticated behavior modeling and dynamic feature response when facing complex resource contention scenarios on cloud platforms.

3.2. Attention-based Feature Weighting

In order to further improve the model's ability to identify key resource indicators and important time segments, this paper designs and introduces the Attention-based Feature Weighting (AFW) module to achieve adaptive modeling of feature importance based on multi-scale convolutional features. The core idea of this module is to calculate the attention scores in the time dimension and feature dimension respectively, and improve the contribution of key segments in subsequent classification through a weighted mechanism. Specifically, after extracting rich multi-scale temporal features through the preceding convolutional layers, the AFW module applies two independent attention branches: one focusing on temporal attention to capture salient time points where resource contention patterns exhibit significant variation, and another targeting feature-wise attention to emphasize resource indicators that contribute more decisively to distinguishing contention behaviors. These attention scores are then used to generate a two-dimensional attention mask, which is applied to the feature map through element-wise multiplication, thereby modulating the original features in a soft and learnable way. This structure enables the model to not only learn which time steps and features are more informative but also suppress irrelevant or noisy information that might interfere with the classification. By integrating AFW into the end-to-end network, the model gains enhanced focus and interpretability, making it better suited to handle the complex, multi-dimensional nature of resource contention in cloud computing systems. The detailed module architecture of AFW is illustrated in Figure 3.

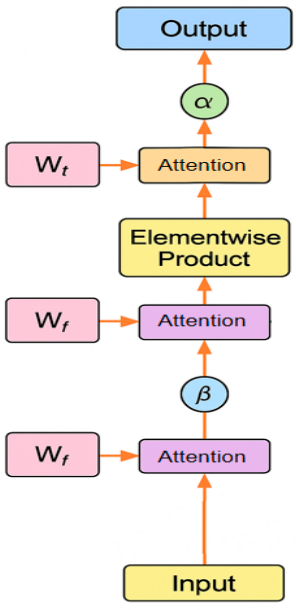

Figure 3. AFW module architecture.

 doi:10.20944/preprints202512.1556.v1

Assume that the input feature tensor is $Z \in R^{T \times F}$, where T represents the number of time steps and F represents the feature dimension, then the temporal attention weight vector can be defined as:

$$\alpha = Soft\max(W_t Z_T + b_t)$$

Among them, $W_t \in R^{1 \times T}$ and $b_t \in R$ are learnable parameters, and the output $\alpha \in R^T$ represents the importance weight of each time point.

Similarly, in the feature dimension, we calculate the importance distribution of feature channels. Let $\beta \in R^T$ represent the feature attention vector, which is calculated as follows:

$$\beta = Sigmoid(W_f Z + b_f)$$

$W_f \in R^{F \times 1}, b_f \in R$ is a trainable parameter, and the Sigmoid function limits the output to the $(0,1)$ interval to achieve soft selection.

Temporal attention and feature attention are then jointly applied to the weighted combination of input features, achieving saliency enhancement in two dimensions through element-by-element multiplication. The final weighted feature representation Z' can be expressed as:

$$Z' = H((\alpha \otimes 1_F) \otimes (1_T \otimes \beta), Z)$$

$\otimes$ represents the broadcast expansion operation, and $H(\cdot,\cdot)$ represents the Hadamard product, which is element-by-element multiplication. This mechanism ensures that the model can jointly focus on representative feature subspaces and time segments.

In order to enhance the nonlinear modeling capability of the model, the AFW module also introduces a gating mechanism to map weighted features to a unified representation space through a nonlinear transformation function. The gating function g is defined as follows:

$$g(Z') = RELU(W_g Z' + b_g)$$

where $W_g \in R^{F \times F}, b_g \in R^F$ is a learnable parameter. The output after the gated transformation is used as the final feature representation input to the classifier. The AFW module effectively improves the model's sensitivity to key influencing factors in heterogeneous resource indicators and has significant structural interpretability and generalization modeling capabilities.

4. Experimental Results

4.1. Dataset

This study uses the publicly released Alibaba Cluster Trace 2018, which records container-level scheduling logs, machine-level resource usage, and application traces for thousands of servers in a real large-scale production cluster. The trace provides second-/minute-level multivariate time series of key metrics (CPU, memory, disk I/O, network) plus machine and task status, enabling labeling of contention events such as CPU saturation, memory overflow, and bandwidth congestion. With nearly one million records and rich fluctuation patterns, it is widely used and well suited for large-scale deep learning on resource contention classification.

4.2. Experimental Setup

To validate the effectiveness of the proposed method in a realistic cloud computing scenario, this study conducts experiments based on the Alibaba Cluster Trace 2018 dataset. The dataset provides fine-grained, time-series resource monitoring data collected from thousands of servers in a large-scale production cluster, covering key indicators such as CPU utilization, memory usage, disk I/O, and network throughput. To prepare the data for model training and evaluation, all raw resource metrics are first normalized using min-max scaling to ensure consistency across different resource types and magnitudes. Continuous monitoring sequences are then segmented into fixed-length windows, each containing 60 time steps sampled at one-minute intervals. Each segment serves as an

 doi:10.20944/preprints202512.1556.v1

input sample representing the temporal behavior of resource usage within a short-term operating period.

The dataset is split into training, validation, and testing sets following an 8:1:1 ratio to maintain temporal consistency and statistical distribution across different phases. The model is trained using the Adam optimizer with an initial learning rate of 0.001 and a batch size of 128. The number of training epochs is set to 200, and the loss function used is cross-entropy, which is suitable for multi-class classification tasks. To ensure reproducibility and fairness, the same hyperparameter configuration is applied across all experimental runs. Table 1 summarizes the core experimental settings used throughout the evaluation process, covering both data handling and model training aspects. The detailed settings are shown in Table 1.

Table 1. Core hyperparameter configuration.

Parameter	Value
Time window length	60
Feature Dimension	5
Batch Size	128
Learning Rate	0.001
Optimizer	128
Activation Function	RELU
Epochs	200
Loss Function	Cross-Entropy

4.3. Experimental Results

(1) Comparative experimental results

This paper first gives the comparative experimental results, as shown in Table 2.

Table 2. Comparative Results.

Method	ACC	Precision	Recall	F1-Score	AUC
Ours	0.936	0.921	0.947	0.934	0.962
CNN with Attention[28]	0.902	0.885	0.915	0.900	0.937
Transformer-based Temporal Classifier[29]	0.911	0.895	0.928	0.911	0.945
LSTM with Attention[30]	0.884	0.868	0.892	0.880	0.924
Graph Temporal Embedding Network[31]	0.894	0.876	0.902	0.889	0.931

As shown in Table 2, the proposed model outperforms all public baselines on every metric, achieving 0.936 Accuracy, 0.962 AUC, and 0.947 Recall, demonstrating strong overall performance and robustness for resource contention detection. Compared with CNN-based attention models, it improves Precision and F1 by 3.6% and 3.4%, confirming that the MSDC + AFW design better captures bursty local patterns and emphasizes critical metrics while suppressing noise. Although Transformer-, LSTM-, and graph-based methods have certain advantages in global or structural modeling, their higher complexity and weaker performance on short-to-medium patterns make them less suitable for high-frequency, large-scale monitoring data. The proposed hierarchical convolution–attention architecture offers a better trade-off between accuracy, efficiency, and generalizability for real-world cloud O&M scenarios. shown in the comparative experimental results in Table 2, the proposed model outperforms all publicly available baseline methods across all evaluation metrics. This demonstrates its strong modeling capability in the resource contention classification task. In particular, the model achieves an Accuracy of 0.936 and an AUC of 0.962. These results indicate a clear advantage in overall classification ability and adaptability to sample distribution. The findings

 doi:10.20944/preprints202512.1556.v1

confirm the model's reliability and stability in identifying diverse resource contention patterns under complex cloud environments.

(2) Ablation Experiment Results

Table 3 summarizes the ablation results. The baseline model performs worst (Recall 0.883, AUC 0.919). Adding MSDC raises Recall to 0.918 and F1 to 0.906, while adding AFW mainly improves Precision from 0.872 to 0.889. With both MSDC and AFW, the full model achieves the best performance (F1 0.934, AUC 0.962), showing the two modules are complementary.

Table 3. Ablation Experiment Results.

Method	ACC	Precision	Recall	F1-Score	AUC
Baseline	0.891	0.872	0.883	0.877	0.919
+MSDC	0.914	0.894	0.918	0.906	0.939
+AFW	0.907	0.889	0.910	0.899	0.933
Ours	0.936	0.921	0.947	0.934	0.962

(3) Sensitivity experiment of dilation rate setting in multi-scale dilated convolutional structures

This paper also gives a sensitivity experiment on the dilation rate setting in the multi-scale dilated convolution structure, and the experimental results are shown in Figure 4.

As shown in Figure 4, model performance is highly sensitive to the dilation rates in the multi-scale dilated convolution. The best results (Accuracy 0.936, Recall 0.947, AUC 0.962) occur with dilation rates (1, 2, 4, 8, 16), which balance short- and long-term dependencies. Dense settings such as (1, 2, 4) overfit short-term changes, while sparse settings such as (4, 8, 16, 32) miss fine-grained patterns. Thus, carefully chosen, moderately spaced dilation rates are crucial for stable and accurate resource contention classification.

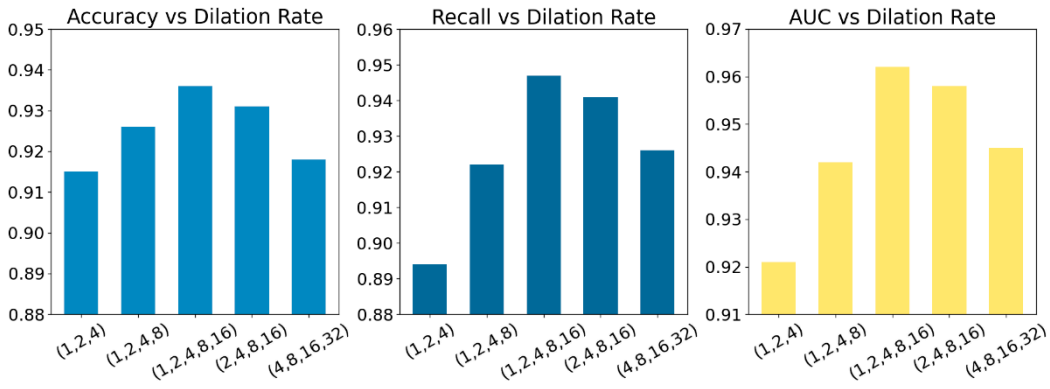

Figure 4. Sensitivity experiment of dilation rate setting in multi-scale dilated convolutional structures.

5. Conclusions

This paper focuses on the intelligent classification of resource contention behaviors in cloud computing environments. It proposes a deep learning framework that combines multi-scale temporal convolution and attention mechanisms. The goal is to enhance classification performance under highly dynamic and multi-dimensional conditions. The proposed approach integrates a Multi-Scale Dilated Convolution module (MSDC) to model resource usage patterns across different temporal granularities and designs an Attention-based Feature Weighting module (AFW) to highlight key resource metrics and time segments. Structurally, this framework improves the model's responsiveness to critical factors in contention behaviors. It also provides clear advantages in representation ability, temporal modeling, and interpretability. The algorithm design reflects strong adaptability and engineering feasibility for real-world cloud environments. In contrast, the proposed

 doi:10.20944/preprints202512.1556.v1

model uses an end-to-end architecture to learn directly from raw monitoring data, eliminating the need for extensive manual feature engineering. This makes it more suitable for modern cloud platforms that demand real-time processing of large-scale data. The ability to discriminate low-level resource behaviors supports finer-grained resource scheduling, container orchestration, and fault prediction, offering significant practical value and application potential.

In addition, the proposed architecture is not limited to resource contention classification. It also provides a general modeling framework for other time series classification problems in system operations. Areas such as intelligent operations, edge computing, and microservice governance face similar challenges in handling high-dimensional time series data. Tasks like resource state modeling, service anomaly detection, and runtime behavior classification can benefit from this approach. Therefore, this work not only advances the study of contention behavior recognition but also offers structural insights and technical references for intelligent decision-making in related fields. Future research may proceed in two directions. First, exploring lightweight model structures could help meet the resource constraints of edge devices and improve real-time deployment. Second, integrating more multimodal data from system and application layers, such as logs, configuration changes, and task graphs, could enhance behavioral modeling and support causal analysis. Embedding the model into real cloud scheduling and elasticity strategies to form a closed loop from monitoring to control is another important direction for further study.

References

1. V. Meyer, D. F. Kirchoff, M. L. Da Silva et al., "ML-driven classification scheme for dynamic interference-aware resource scheduling in cloud infrastructures," Journal of Systems Architecture, vol. 116, Article 102064, 2021.
2. W. Khallouli and J. Huang, "Cluster resource scheduling in cloud computing: literature review and research challenges," The Journal of Supercomputing, vol. 78, no. 5, pp. 6898-6943, 2022.
3. M. A. N. Saif, S. K. Niranjan and H. D. E. Al-Ariki, "Efficient autonomic and elastic resource management techniques in cloud environment: taxonomy and analysis," Wireless Networks, vol. 27, no. 4, pp. 2829-2866, 2021.
4. J. Zhang, H. Yu, G. Fan et al., "Handling hierarchy in cloud data centers: A Hyper-Heuristic approach for resource contention and energy-aware Virtual Machine management," Expert Systems with Applications, vol. 249, Article 123528, 2024.
5. R. Meng, H. Wang, Y. Sun, Q. Wu, L. Lian and R. Zhang, "Behavioral Anomaly Detection in Distributed Systems via Federated Contrastive Learning," arXiv preprint arXiv:2506.19246, 2025.
6. C. F. Chiang, D. Li, R. Ying, Y. Wang, Q. Gan and J. Li, "Deep Learning-Based Dynamic Graph Framework for Robust Corporate Financial Health Risk Prediction," 2025.
7. A. Xie and W. C. Chang, "Deep Learning Approach for Clinical Risk Identification Using Transformer Modeling of Heterogeneous EHR Data," arXiv preprint arXiv:2511.04158, 2025.
8. D. W. A. S. Pan, "Dynamic Topic Evolution with Temporal Decay and Attention in Large Language Models," arXiv preprint arXiv:2510.10613, 2025.
9. J. Lai, A. Xie, H. Feng, Y. Wang and R. Fang, "Self-Supervised Learning for Financial Statement Fraud Detection with Limited and Imbalanced Data," 2025.
10. R. Xu, R. Kumar, P. Wang et al., "ApproxNet: Content and contention-aware video object classification system for embedded clients," ACM Transactions on Sensor Networks (TOSN), vol. 18, no. 1, pp. 1-27, 2021.
11. A. Suresh, "Mitigating Resource Contention in Today's Data Centers," Ph.D. dissertation, State University of New York at Stony Brook, 2021.
12. Q. Wu, M. C. Zhou and J. Wen, "Endpoint communication contention-aware cloud workflow scheduling," IEEE Transactions on Automation Science and Engineering, vol. 19, no. 2, pp. 1137-1150, 2021.
13. V. Meyer, M. L. da Silva, D. F. Kirchoff et al., "Iada: A dynamic interference-aware cloud scheduling architecture for latency-sensitive workloads," Journal of Systems and Software, vol. 194, Article 111491, 2022.

 doi:10.20944/preprints202512.1556.v1

14. W. Mao, J. Tan, W. Tan et al., "E-CARGO-based dynamic weight offload strategy with resource contention mitigation for edge networks," Journal of Industrial Information Integration, vol. 42, Article 100695, 2024.
15. T. Langehaug, B. Borghetti and S. Graham, "Classifying co-resident computer programs using information revealed by resource contention," Digital Threats: Research and Practice, vol. 4, no. 2, pp. 1-29, 2023.
16. A. Kumar and G. Somani, "Service separation assisted DDoS attack mitigation in cloud targets," Journal of Information Security and Applications, vol. 73, Article 103435, 2023.
17. M. Wang, S. Wang, Y. Li, Z. Cheng and S. Han, "Deep neural architecture combining frequency and attention mechanisms for cloud CPU usage prediction," 2025.
18. Q. R. Xu, W. Xu, X. Su, K. Ma, W. Sun and Y. Qin, "Enhancing Systemic Risk Forecasting with Deep Attention Models in Financial Time Series," 2025.
19. X. Yan, Y. Jiang, W. Liu, D. Yi and J. Wei, "Transforming Multidimensional Time Series into Interpretable Event Sequences for Advanced Data Mining," 2024 5th International Conference on Intelligent Computing and Human-Computer Interaction (ICHCI), pp. 126-130, 2024.
20. X. Liu, Y. Qin, Q. Xu, Z. Liu, X. Guo and W. Xu, "Integrating Knowledge Graph Reasoning with Pretrained Language Models for Structured Anomaly Detection," 2025.
21. J. Wei, Y. Liu, X. Huang, X. Zhang, W. Liu and X. Yan, "Self-Supervised Graph Neural Networks for Enhanced Feature Extraction in Heterogeneous Information Networks", 2024 5th International Conference on Machine Learning and Computer Application (ICMLCA), pp. 272-276, 2024.
22. R. Liu, R. Zhang and S. Wang, "Graph Neural Networks for User Satisfaction Classification in Human-Computer Interaction," arXiv preprint arXiv:2511.04166, 2025.
23. H. Zheng, L. Zhu, W. Cui, R. Pan, X. Yan and Y. Xing, "Selective Knowledge Injection via Adapter Modules in Large-Scale Language Models," 2025.
24. X. Yan, J. Du, L. Wang, Y. Liang, J. Hu and B. Wang, "The Synergistic Role of Deep Learning and Neural Architecture Search in Advancing Artificial Intelligence", Proceedings of the 2024 International Conference on Electronics and Devices, Computational Science (ICEDCS), pp. 452-456, Sep. 2024.
25. H. Liu, Y. Kang and Y. Liu, "Privacy-Preserving and Communication-Efficient Federated Learning for Cloud-Scale Distributed Intelligence," 2025.
26. X. Chen, S. U. Gadgil, K. Gao, Y. Hu and C. Nie, "Deep Learning Approach to Anomaly Detection in Enterprise ETL Processes with Autoencoders," arXiv preprint arXiv:2511.00462, 2025.
27. C. Hu, Z. Cheng, D. Wu, Y. Wang, F. Liu and Z. Qiu, "Structural Generalization for Microservice Routing Using Graph Neural Networks," arXiv preprint arXiv:2510.15210, 2025.
28. J. Dogani, F. Khunjush, M. R. Mahmoudi et al., "Multivariate workload and resource prediction in cloud computing using CNN and GRU by attention mechanism," The Journal of Supercomputing, vol. 79, no. 3, pp. 3437-3470, 2023.
29. Q. Wen, T. Zhou, C. Zhang et al., "Transformers in time series: A survey," arXiv preprint arXiv:2202.07125, 2022.
30. D. Soni and N. Kumar, "Machine learning techniques in emerging cloud computing integrated paradigms: A survey and taxonomy," Journal of Network and Computer Applications, vol. 205, Article 103419, 2022.
31. J. Pei, Y. Hu, L. Tian et al., "Dynamic anomaly detection using In-band Network Telemetry and GCN for cloud–edge collaborative networks," Computers & Security, vol. 154, Article 104422, 2025.