EURASIAN JOURNAL OF MATHEMATICAL AND COMPUTER APPLICATIONS
ISSN 2306–6172
Volume 14, Issue 2 (2026) 126 – 144

EXPLAINABLE HYBRID DEEP LEARNING FOR TASK TYPE PREDICTION IN CLOUD COMPUTING

Yousefi S. [1], **Haseebsalman Z.**, **Igenbayev A.**, **Akylaev Z.**

Abstract Cloud computing environments execute heterogeneous and dynamic workloads. Accurate task type prediction is required for efficient scheduling, resource allocation, and Quality of Service. Raw performance metrics remain difficult to classify because of noise, class imbalance, and overlapping resource patterns. This study proposes an explainable hybrid deep learning model for task type prediction. A three stage pipeline is applied for data preprocessing, model construction, and optimization. Feature selection, encoding, normalization, denoising, class balancing, and stratification improve data quality. A convolutional neural network extracts local feature patterns, while a bidirectional long short term memory network captures temporal task relationships. Model optimization is achieved through the Adam optimizer and focal loss. Experimental results demonstrate superior classification performance over baseline and recent hybrid models. Ablation analysis validates the preprocessing strategy and model architecture. We show that the proposed method provides reliable workload prediction for cloud computing environments.

Keywords: Cloud computing, Deep learning, Explainable AI, Information Systems, Modeling and Simulation, Applied Mathematics, Computational Mathematics, Computer Science Applications, Task type Prediction, Workload analysis.

AMS Mathematics Subject Classification: 68T05, 68T07, 68M11, 62H30.

DOI: 10.32523/2306-6172-2026-14-2-126–144

1 Introduction

Cloud computing provides scalable and on demand computational resources for diverse applications [1, 2]. Modern cloud environments support heterogeneous workloads with dynamic execution patterns and variable resource demands [3]. Multi tenant systems require efficient scheduling and resource allocation to satisfy Quality of Service requirements. Increasing workload complexity reduces resource efficiency and increases operational costs. Service Level Agreement violations may also occur [4, 5]. High energy consumption further increases management challenges in large scale data centers [6]. We therefore require intelligent workload management methods that accurately identify task behavior and adapt resource allocation under dynamic cloud conditions. Correct task classification is crucial for intelligent decision making within cloud computing [7]. Tasks have different computation and communication needs [8]. Identifying tasks prior to their execution enhances scheduling and resource allocation. Workload-aware optimization and proactive resource allocation are additionally enabled [9]. Correct resource allocation decreases task execution time and optimizes system performance. It also helps in increasing resource utilization. Wrong task classification leads to poor system performance and high energy usage and cost [1, 5]. Within multi tenant systems, such an effect increases the possibility of Service Level Agreement non-compliance [8]. AI with

[1]Corresponding Author.

the help of Machine learning (ML) and deep learning (DL)-based models can be adopted to address the complexity of cloud computing environments as discussed in [10]. AI tools can enable several essential tasks, e.g., resource allocation, workload prediction, energy efficiency, security, and system reliability [5]. Several studies in the literature e.g., [11] demonstrated that advanced learning models can achieve substantial performance gains in specific application domains, including energy-aware scheduling, intrusion detection, traffic flow prediction, workload forecasting, and fault management [12, 13, 14, 26, 27]. Despite these advances, the literature remain focused on problem-specific objectives and operate at the resource level, which offers limited support for generalized task-level prediction across cloud workloads. Furthermore, the majority of models function as black boxes, providing little insight into how performance metrics influence decisions, which restricts their practical applicability in real-world cloud management systems. These limitations emphasise the need for an explainable and general-purpose prediction task types using performance metrics, capturing both spatial and temporal workload characteristics, and decision-making in cloud computing. Concequently in this study, an Explainable CNN-BiLSTM method for cloud task, i.e., X-CNN-BiLSTM is proposed. The main objective is the identification of task types based on system-level performance metrics while preserving robustness and interpretability. The model consists of a three-stage pipeline: (1) data preprocessing, (2) hybrid model construction, and (3) model training and optimization. Cloud metrics in their raw form undergo preprocessing through feature selection, encoding, normalization, denoising, balancing of classes, and stratification of samples. Hence, high-quality and unbiased input data are obtained. A hybrid model of deep learning is then designed for predicting task types. Convolutional neural network is used for the extraction of spatial local features from cloud performance metrics. Bidirectional long short term memory network captures task dependencies based on time. Focal loss and Adam optimizer enhance learning stability and classification of the minority classes. In addition, we have included the explainability to enhance the clarity of the model. Our experiments show a better performance than the baseline and recent hybrid methods. Ablation study validates the approach adopted for preprocessing as well as for designing the network. Thus, our model makes an optimal trade-off between prediction accuracy, efficiency and interpretability. The objective of this study is that via experimental results would investigate the effectiveness of X-CNN-BiLSTM for task type prediction in cloud environments. Extensive evaluations conducted on a public cloud performance dataset should study that the model performance across heterogeneous workloads and outperforms several baseline and recent hybrid approaches. Further more the study contributes to study the integration of a multi-stage preprocessing pipeline with a CNN-BiLSTM architecture weather it enables effective learning of localized resource usage patterns and long-range temporal dependencies. In addition, comparative and ablation analyses should be conducted to study contributions of preprocessing, architectural ordering, and imbalance-aware optimization to overall performance improvement. The remainder of this paper is organized as follows. Section 2 reviews related studies on task type prediction and intelligent cloud management. Section 3 presents the proposed X-CNN-BiLSTM model. Section 4 reports the experimental results and performance evaluation. Section 5 discusses the results, including practical implications and limitations. Finally, Section 6 concludes the paper and presents directions for future research.

2 Related Work

Th erelevant litrature includes a wide range AI applications in cloud computing, i.e., energy-efficient scheduling, resource allocation, workload prediction, security, and fault management.

For instant Khan et al. [15] proposed a hybrid ML model that combines CNN and BiLSTM to improve energy-efficient task scheduling in fog-cloud. Their model optimizes task placement between fog and cloud nodes under QoS constraints while reducing energy consumption for better reliability. Their results indicate reduced energy consumption, higher job completion rates, lower response times, and fewer SLA violations. However, the study focuses on energy-aware scheduling, and the evaluation is restricted to specific simulated fog-cloud scenarios. As a result, uncertainty remains regarding its applicability to broader cloud systems with diverse task profiles. In addition, the model does not include an explicit explainability component that relates workload features to task scheduling decisions. On the other hand Wang and Yang [13] presented an energy consumption prediction study for cloud environments based on an improved Kernel Extreme Learning Machine (KELM) enhanced with a Vector Weighted Average (VWAA) algorithm. The method adjusts feature weights and optimizes kernel functions to capture nonlinear relationships in an effective manner. Their findings demonstrate relatively acceptable performance. Also their model achieved high coefficients of determination which indicates accurate and stable energy consumption estimation. However, their model reported to be limited to regression-based energy consumption prediction and does not address task-level intelligence. This limitation reduces its aplicability and generalization. Their method also does not provide an explainabile model. In other study by Ali et al. [16] an energy-efficient resource allocation model for urban traffic flow prediction in edge-cloud computing was proposed which promotes an intelligent transportation for resource constraints. Their study integrates traffic flow prediction with adaptive resource allocation to reduce energy consumption and maintain prediction accuracy and system responsiveness. Although their study improves prediction accuracy while reducing energy consumption and computational overhead, it does not address task-level classification within general cloud computing workloads. In addition, their proposed methods can not be considered an explainabile. Haider et al. [12] designed a hybrid DL architecture that integrates BiLSTM and Bidirectional Gated Recurrent Units (BiGRU) to enhance intrusion detection in cloud environments. The model exploits the strengths of BiLSTM for long-term temporal dependency modeling and BiGRU for efficient gating and reducing computational overhead. Their simulation achieves good accuracy while it reduces computational latency and memory usage. However, the model is tailored to security analytics and does not address task-level workload characterization required for broader cloud management systems. Their study again lacks explainability. Seshadri et al. [17] proposed a hierarchical workload characterization and prediction framework to address the high variability of cloud workloads in data centers. Their research introduces a Super Markov Prediction Model (SMPM) that switches between sequence models based on evolving workload patterns to improve elasticity-aware resource management. Although SMPM achieves lower prediction errors, it focuses on workload volume prediction rather than task-level intelligence. The model also operates as a black-box sequence predictor without an explicit explainability mechanism. Salanke et al. [18] presented a proactive fault management model for cloud environments that combines task failure prediction with dynamic task migration. It employs BiLSTM to capture temporal dependencies in task execution data and identify impending failures before they occur. Based on the predicted failures, a migration strategy reallocates tasks to alternative resources to reduce service disruption and improve system reliability. Experimental results on real-world data demonstrate that the BiLSTM-based predictor identifies task failures, reduces downtime, and improves resource utilization. Nevertheless, it emphasizes failure prediction, without addressing task type prediction that is essential for proactive scheduling and workload-aware optimization. Moreover, the model operates as a black-box temporal predictor and does not provide explainability. Sangani

et al. [19] introduced an efficient algorithm for error optimization and resource prediction to reduce operational cost and energy consumption in cloud environments. The authors focus on predicting resource requirements and minimizing allocation errors using historical workload and performance information. Although the paper reduces prediction errors and achieves a reasonable improvement in operational cost, it addresses resource-level prediction and does not incorporate task-level prediction. Moreover, the approach lacks an explainability method to show how individual performance metrics influence prediction outcomes. Gurusamy and Selvaraj [20] designed a resource allocation and task scheduling architecture for cloud computing based on a Hierarchical Auto-associative Polynomial Convolutional Neural Network (HAPCNN). Authors integrate DL-based resource allocation with an optimization-driven short-term scheduler to address challenges related to workload growth, fairness, and system efficiency. In addition, the paper incorporates encryption mechanisms to ensure data security while optimizing cloud performance under multiple design constraints. Experimental results demonstrate that it improves cloud performance, achieving lower energy consumption, reduced response time, and higher throughput. However, the architecture is designed for resource allocation and scheduling optimization and does not address task type prediction as an intelligence layer for proactive cloud decision-making. Furthermore, the paper lacks an explainability mechanism. Lajili et al. [14] presented an ML-based framework for workload prediction and task clustering to support efficient stream application offloading in heterogeneous edge-cloud environments. It profiles streaming application workloads and applies unsupervised techniques to group tasks with similar characteristics and enable suitable resource selection for offloading decisions. Multiple clustering algorithms are evaluated, with k-means identified as the most effective in aligning clustered tasks with an actual workload. Simulation results demonstrate that the paper improves offloading efficiency and resource utilization. Nevertheless, the framework is designed for stream application offloading in edge-cloud scenarios and relies on unsupervised clustering. Moreover, the approach does not provide an explainability mechanism to interpret how performance metrics influence task grouping and offloading decisions.

Table 1 provides a comparative overview of recent studies that apply ML-DL techniques to address issues in cloud computing. Overall, the literature demonstrates that advanced learning models, such as CNN-BiLSTM, BiLSTM-BiGRU, KELM-based predictors, clustering-based frameworks, and optimization-driven architectures, can improve performance metrics such as energy consumption, throughput, response time, prediction accuracy, and system reliability. However, a critical limitation that we observed across these works is their problem-specific focus, where most approaches target a single aspect of cloud management for instance energy optimization, intrusion detection, traffic prediction, or failure mitigation, rather to provid a unified task-level intelligence model which is applicable to general cloud workloads. Moreover, many studies emphasize workload volume prediction, clustering, or resource-level optimization, while explicit task type prediction, which is essential for proactive scheduling and workload-aware decision-making, remains unexplored. Another major limitation is the lack of explainability. Although many models achieve strong predictive performance, most operate as black-box systems and do not provide interpretable insights into how performance metrics affect predictions. This limitation reduces their practical adoption in real-world cloud operations. These shortcomings indicate the need for a generalized and explainable approach that predicts task types from performance metrics, captures both spatial and temporal workload characteristics, and provides transparent support for decision-making.

Table 1: Summary of related studies on intelligent cloud computing and workload analytics

Authors & Year	Objective	Approach	Dataset	Results	Limitations/Gaps
Khan et al., 2025 [15]	Energy-efficient task scheduling in fog-cloud environments	Hybrid CNN-BiLSTM	COSCO simulation datasets for fog-cloud environments	Efficient energy consumption, improved job completion rate, lower response time, and fewer SLA violations	Does not address task type prediction; lacks explainability
Wang & Yang, 2025 [13]	Predict cloud data center energy consumption	VWAA-KELM	Real-world cloud data (conference dataset)	Achieved high predictive accuracy with strong generalization	Focuses on regression-based energy prediction and does not support task-level classification; limited interpretability
Ali et al., 2025 [16]	Energy-efficient resource allocation in edge-cloud systems	LSTM	Urban traffic and IoV-based edge-cloud datasets	Improved traffic prediction accuracy and reduced energy consumption	Application-specific and traffic flow scenarios; lacks explainable modeling
Haider et al., 2025 [12]	Enhance cloud intrusion detection	Hybrid BiLSTM-BiGRU	CIC-IDS 2018	Achieved a high accuracy with reduced latency and memory usage; strong detection of known and zero-day attacks	Designed for security analytics rather than task intelligence; does not provide explainability
Seshadri et al., 2024 [17]	Predict dynamic cloud workloads for elastic resource management	SMPM	Alibaba Trace 2018 & 2020, Google Cluster Trace, TPC-W	Lower prediction errors	Focuses on workload volume prediction; operates as a black-box predictor
Salanke et al., 2024 [18]	Predict task failures and enable proactive migration	BiLSTM + dynamic task migration strategy	Real-world cloud execution data	Improved failure prediction accuracy and reduced service downtime	Concentrates on failure prediction and reactive migration; lacks interpretability
Sangani et al., 2024 [19]	Reduce operational cost and energy consumption through error optimization and resource prediction	Soft-computing-based optimization + task-ordering management	Cloud workload and performance datasets (experimental environment)	Achieved reduced prediction error, lower energy consumption, and decreased operational cost	Addresses resource-level prediction; lacks explainability
Gurusamy & Selvaraj, 2024 [20]	Improve resource allocation and task scheduling in cloud computing	HAPCNN + SMO	Cloud workload and performance datasets (simulation-based environment)	Achieved lower energy consumption, higher throughput, and reduced response time	Resource allocation and scheduling optimization; lacks an explainability mechanism
Lajili et al., 2024 [14]	Efficient stream application offloading in heterogeneous edge-cloud environments	K-means, MeanShift, Agglomerative	Simulated streaming application workloads	Improved offloading efficiency and resource utilization; K-means demonstrated superior clustering quality	Application-specific to stream application offloading and edge-cloud scenarios; relies on unsupervised clustering

3 Proposed Method

In this section, we propose an Explainable CNN-BiLSTM model for cloud task type prediction (X-CNN-BiLSTM), as illustrated in Figure 1. It is designed as a three-stage pipeline comprising (1) data preprocessing, (2) hybrid model construction, and (3) model training and optimization. In the preprocessing stage, raw cloud performance metrics are cleaned and transformed through missing data handling, categorical encoding, normalization, denoising, and class balancing to ensure high-quality and reliable inputs. The core of the framework is a hybrid DL architecture. A CNN is used to extract localized spatial patterns from multidimensional time-series metrics. A BiLSTM network then captures forward and backward temporal dependencies in task execution behavior. This order improves feature representation and reduces noise propagation into the temporal modeling layer.

In the final stage, model training is performed using the Adam optimizer and focal loss. This improves convergence, addresses class imbalance, and strengthens predictive robustness across diverse task types. To support scalability and near-real-time deployment in dynamic cloud environments, attention-based mechanisms are excluded. This design enables a balanced trade-off among predictive accuracy, interpretability, and computational efficiency.

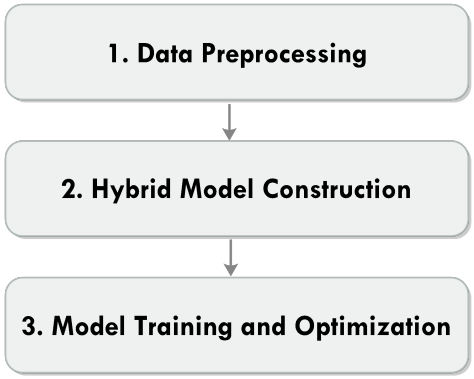

Figure 1: Architecture of the Explainable CNN-BiLSTM for cloud task type prediction

3.1 Data Preprocessing

The data preprocessing phase serves an extremely important task of preparing the cloud performance metrics into a form suitable for training the proposed model effectively. The process is comprised of seven steps, including feature selection and labelling, filtering out of missing or invalid data, feature encoding, feature normalization, data denoising, balancing of classes, and splitting of data into train and test sets, as shown in Figure 2. Even though the data preprocessing process comes at a small computational cost, it is negligible in comparison with the model training process and is fully justified due to the large increase in data quality and training stability.

3.1.1 Feature Selection

Feature selection is an essential part of the X CNN BiLSTM model. In cloud computing, there is generation of multiple performance metrics that contain redundant and irrelevant features. Redundant features can be removed using correlation analysis, mutual information, and model based ranking. Thus, noise and bias are eliminated. Only relevant features are selected to train the model effectively. The task type is the target variable in the machine

Figure 2: Data preprocessing workflow

learning approach. There will be historical usage of resources which can help to extract discriminative patterns.

3.1.2 Removing Missing or Invalid Data

In addition to feature selection, data cleansing technique is employed in order to ensure that high-quality data is used. Inconsistencies and missing values are eliminated from the dataset. As a result, reliable and clean datasets are derived. Inconsistencies occur as a result of data logging and measurement problems. These inconsistencies affect the reliability and stability of any models built from them. We eliminate incomplete entries in order to ensure that we only have useful information about the resource consumption behavior of applications. This is crucial in the hybrid CNN BiLSTM model since the spatial and temporal feature learning process is enhanced.

3.1.3 Feature Encoding and Normalization

Feature Encoding refers to the transformation of categorical data to numeric form, in order to be used in deep learning. The cloud dataset has categorical data that needs appropriate encoding. Label Encoding is done on the ordinal data while one hot encoding is performed on the non ordinal data. Proper encoding helps maintain the task relevant data and avoid any artificial numeric relationships. The correlation analysis shows low redundancy in the encoded data. The unique data is hence maintained in the selected features. Following the feature encoding, numerical features will be scaled by using Min Max scaling method [21]. All the feature values will be transformed into the range between 0 and 1. Performance indicators of clouds are measured by using various scales. This makes learning less efficient without normalization. Min Max scaling makes sure that the features are evenly weighted in the model training. Fast convergence and more stable predictions are thus guaranteed. We normalize the features to retain relevant patterns of tasks in the task.

3.1.4 Data Denoising, Class Balancing, Train-Test Data Splitting and Hybrid Model Construction

Data denoising technique is employed to enhance data quality and model resilience. The cloud performance measurement data has anomalies due to system failures and logging issues. Workload irregularities lead to additional noisy data. Consequently, an anomaly detection technique that utilizes both isolation forest and ensemble voting is adopted. An isolation forest and ensemble voting detect abnormal samples [22]. The isolation forest algorithm separates the outliers by data partitioning. The ensemble voting enhances the accuracy of the anomaly detection process. The statistical anomalies are filtered while retaining the valid behavior of the tasks. In the next step the class balancing is used to enhance the fairness of predictions and performance of the model. Datasets collected from clouds have imbalanced distributions

of task types. The dominant classes may influence the learning process of the model. As such, minority task types might be inaccurately predicted by the model. Distribution of classes is examined prior to training of the model. Minority classes are oversampled through creation of artificial samples. Majority classes are undersampled if required. A balanced distribution of tasks is achieved using this approach. The last preprocessing step is the splitting of the dataset into training and testing sets using a 90 to 10 split ratio. The process of random sampling maintains the same class distribution. As such, fair model evaluation is made possible for different task types. No validation set is created because the aim is to have maximum training data. Model generalizability can be assured with the help of the aforementioned preprocessing, class balancing, focal loss, and held-out testing. The setup above represents real world conditions that prevail in cloud computing. The figure below shows our proposed architecture which is a combination of the X CNN and BiLSTM architectures.

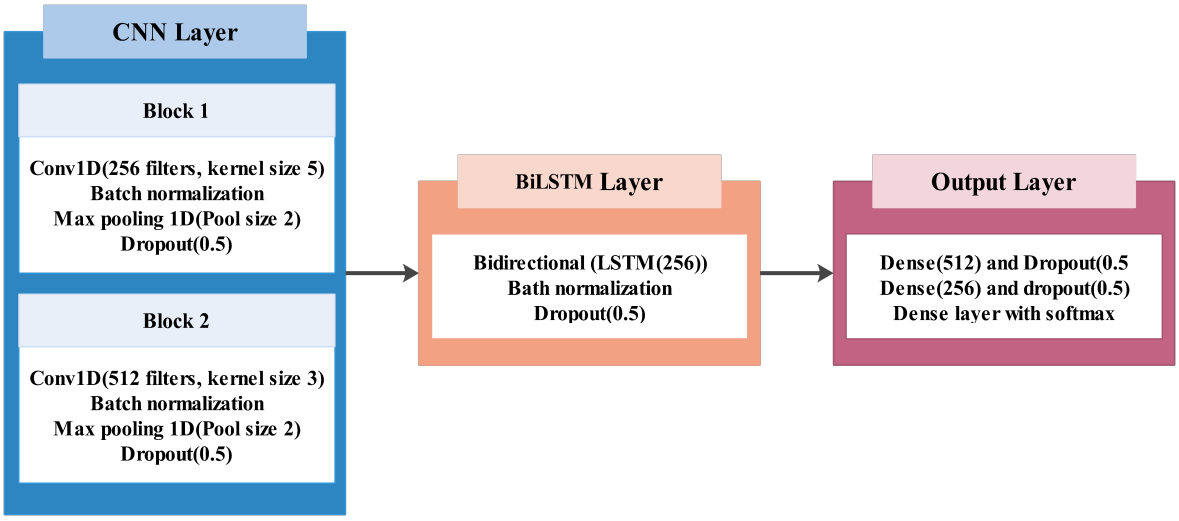

Figure 3: Architecture of the proposed X-CNN-BiLSTM model

3.1.5 CNN and BiLSTM Layers

Convolutional Neural Network is used for extracting spatial features from cloud performance metrics. 1D Convolution is performed on sequential workload data. Local features related to CPU utilization, memory utilization, I O operations, and execution time are automatically learned. Hence manual feature extraction becomes unnecessary. Figure 3 illustrates two consecutive convolutional layers. The first convolutional layer consists of 256 filters with kernel size 5. Feature extraction related to broad workload patterns is performed using this layer. The second convolutional layer consists of 512 filters with kernel size 3. Feature extraction related to fine-tuned task-specific features is done using this layer. Max pooling is used to reduce the dimensionality of the extracted features and enhance noise resistance. Batch Normalization stabilizes the training and helps in converging the model faster. Dropout operation with rate 0.5 is used to avoid overfitting. The extracted features are passed on to the BiLSTM layer for learning temporal dependencies. The BiLSTM layer is the temporal component of the hybrid architecture. It enables learning of sequential dependencies and dynamic task behavior from cloud performance metrics. As shown in Figure 3, the BiLSTM uses 256 hidden units. It processes feature representations from the CNN layer in both forward

and backward directions. This allows inclusion of contextual information from past and future states. This bidirectional modeling is suitable for cloud environments. Task patterns are influenced by evolving resource conditions and interdependent system events rather than isolated observations. Batch normalization and dropout with a rate of 0.5 are also applied. These steps stabilize training and reduce overfitting in high-variance workloads. Long-range temporal dependencies are preserved through this layer. It enables separation of task types that show similar short-term resource usage but different temporal structures. This improves classification accuracy and robustness in unseen cloud workload scenarios.

3.1.6 Output Layer

Output layer is used for task type classification. Spatial-temporal features learnt by CNN and BiLSTM layers are fused. A three-stage dense architecture is illustrated in Fig. 3. The first dense layer uses 512 neurons to learn the complex relationship among the learned features. A dropout layer with a rate of 0.5 is used to reduce overfitting problems. Second dense layer uses 256 neurons for learning compact representation of the features. Again, a dropout layer is used to increase the generalization capability of the model. The final layer uses Softmax function for multi-classification purposes. The probability scores of task types are calculated using these scores.

3.2 Model Training and Optimization

After the construction of our model architecture, model training is done using the pre-processed data. Our model uses focal loss [23], Adam optimizer [24], 50 training epochs, and batch size of 256 for training our model. Focal loss improves the accuracy of minority class and avoids any biases in predictions. Adam optimizer is helpful in improving the convergence during training our model. The number of training epochs and batch size are carefully chosen to ensure optimal learning with minimal overfitting. Thus, the discriminative spatial and temporal features are learned from diverse workloads. The algorithm for the proposed task prediction framework is shown in Algorithm 1. Pre-processing of data, hybrid model construction, and model training processes are done in a unified process. Convolutional layer captures spatial features, and BiLSTM layer captures temporal dependencies. Attention mechanism is avoided to reduce the complexity.

4 Results

This section introduces properties of datasets, experimental outcomes, and models compared with other techniques recently developed. All experiments are carried out in Python with an Apple M2 computer having 8 GB RAM and running macOS Sonoma 14.4.1. This configuration provides consistent evaluation conditions for all experiments. Table 2 shows the values of hyperparameters. Two CNN modules include 256 and 512 filters with kernel sizes of 5 and 3. Both broad and fine space workload features are obtained throughout network layers. Batch normalization increases the speed of convergence and the process of learning. MaxPooling decreases dimensionality and improves robustness in noisy environments. Dropout with probability 0.5 prevents overfitting in convolutional and dense layers. The BiLSTM layer includes 256 bidirectional units to analyze the time-dependent behavior in tasks. Batch normalization and dropout increase the stability of the learning process. Attention with softmax weighting highlights time-dependent samples and provides higher interpretability of results. The dense

Algorithm 1 Pseudocode of the proposed X-CNN–BiLSTM framework for cloud task type prediction

Require: Raw dataset containing cloud performance metrics and metadata, target label (task_type), train–test split ratio

Hyperparameters
CNN params:
Block1: Conv1D(filters=256, kernel=5) + Batch Normalization + MaxPool + Dropout(0.5)
Block2: Conv1D(filters=512, kernel=3) + Batch Normalization + MaxPool + Dropout(0.5)
BiLSTM params:
BiLSTM(units=256) + BN + Dropout(0.5)
Dense head:
Dense(512) + Dropout(0.5) → Dense(256) + Dropout(0.5) → Dense(C) + Softmax
Training params:
Loss = FocalLoss(γ, α) ▷ γ: focusing parameter, α: class weights
Optimizer = Adam(learning_rate)
Epochs = 50
BatchSize = 256

Ensure: Trained X-CNN–BiLSTM model, test-set evaluation metrics, explainability results

1. Data Preprocessing
1-1 Feature Selection + label definition
Remove redundant and non-informative features using correlation analysis, mutual information ranking, and model-based filtering; define a multi-class supervised target.
1-2 Removing Missing or Invalid Data
1-3 Feature Encoding (categorical → numerical)
1-4 Feature Normalization using Min–Max scaling (numerical → [0,1])
1-5 Data Denoising using a hybrid anomaly detection strategy
(a) Isolation Forest detection
(b) Ensemble voting
(c) Consensus-based anomaly validation
(d) Removal of detected anomalies
1-6 Class Balancing using resampling techniques
1-7 Stratified Train-Test Data Splitting

2. Hybrid Model Construction
2-1 CNN-based spatial feature extraction
2-2 BiLSTM-based bidirectional temporal modeling
2-3 Dense output layers with Softmax classification

3. Model Training and Optimization
using FocalLoss(γ, α) and Adam(learning_rate)

4. Evaluation
Evaluation of the trained model on the held-out test set using standard multi-class classification metrics

5. Explainability
Apply model-level explainability analysis

Table 2: Architecture and training configuration of the Proposed X-CNN-BiLSTM

Category	Layer / Component	Key Settings
CNN Block 1	Conv1D	256 filters, kernel size = 5, ReLU
	Batch Normalization	256 channels
	MaxPooling1D	Pool size = 2
	Dropout	Rate = 0.5
CNN Block 2	Conv1D	512 filters, kernel size = 3, ReLU
	Batch Normalization	512 channels
	MaxPooling1D	Pool size = 2
	Dropout	Rate = 0.5
Temporal Modeling	BiLSTM	256 units, bidirectional
	Batch Normalization	512 units
	Dropout	Rate = 0.5
	Attention	Softmax weighting
Feature Flattening	Flatten	—
Dense Head	Dense	512 neurons, ReLU
	Batch Normalization	512 units
	Dropout	Rate = 0.5
	Dense	256 neurons, ReLU
	Batch Normalization	256 units
	Dropout	Rate = 0.5
Output Layer	Dense (Output)	Softmax activation (multi-class)
Label Encoding	Target Encoding	One-Hot Encoding (LabelBinarizer)
Optimization	Optimizer	Adam
	Learning Rate	0.001
Loss Function	Loss	Focal Loss ($\gamma = 2.0, \alpha = 0.25$)
Evaluation Metrics	Metrics	Accuracy, Precision, Recall, F1-score
Training Setup	Epochs	100
	Batch Size	256
	Early Stopping	Monitor= validation loss, patience = 7

layers decrease the dimensionality from 512 to 256 neurons. The Softmax layer gives multi-class outputs. Label one-hot encoding helps to avoid the ordinal bias. The Adam optimizer is used with a learning rate of 0.001. It provides adaptive convergence behavior. Focal loss with $\gamma = 2.0$ and $\alpha = 0.25$ is used to address class imbalance. It increases focus on hard-to-classify samples. Training uses a batch size of 256 and runs for up to 100 epochs. Early stopping based on validation loss is applied. This prevents overfitting and improves training efficiency. This configuration supports stable learning and reliable deployment in real cloud environments.

4.1 Dataset

The experimental evaluation of the proposed approach is conducted using a cloud performance dataset. It contains diverse system-level metrics related to task execution behavior [26, 27]. As shown in Table 3, the dataset includes both numerical and categorical attributes. These attributes describe resource utilization, execution characteristics, and task metadata. Features such as cpu usage, memory usage,network traffic, power consumption, num executed instructions, execution time, and energy efficiency are used as core numerical inputs. These

features reflect computational load, memory demand, I/O activity, and energy behavior of cloud tasks. They provide useful information for task type prediction. In contrast, identifiers and non-behavioral attributes such as `vm_id` and `timestamp` are removed. These attributes do not provide semantic information about task behavior. The attribute `task_status` is also excluded to avoid information leakage. The target variable is `task_type`. It represents the class label for supervised learning.Approximately 10% missing values are observed across most retained features. These incomplete records are removed during preprocessing to preserve data integrity. The categorical feature `task_priority` is encoded to represent contextual task importance. This encoding avoids the introduction of artificial ordinal relationships. Overall, the selected feature set provides a balanced representation of cloud workloads. It enables the hybrid model to learn discriminative spatiotemporal patterns. It also supports robust task type classification under heterogeneous and noisy real-world conditions.

Table 3: Description of the cloud performance dataset

#	Feature	Target/Feature/Removed	Data Type	Missing Data	Missing Data (%)
1	vm_id	Removed	object	-	-
2	timestamp	Removed	object	-	-
3	cpu_usage	Feature	float64	199038	9.95
4	memory_usage	Feature	float64	200510	10.03
5	network_traffic	Feature	float64	199481	9.97
6	power_consumption	Feature	float64	200271	10.01
7	num_executed_instructions	Feature	float64	199686	9.98
8	execution_time	Feature	float64	199827	9.99
9	energy_efficiency	Feature	float64	200042	10
10	task_priority	Feature	object	199433	9.97
11	task_status	Removed	object	-	-
12	**task_type**	**Target/Label**	**object**	**199962**	**10**

Figure 4 displays the correlation matrix of the selected features against the target variable. It defines the relationship among features prior to training. For the majority of the pairs of features, their correlation value is relatively low. This means there is not much linear dependency and redundancy among features. The different metrics are associated with various aspects of task execution, such as computing, memory, I O operations, and energy behaviors. The correlation between features and `task_type` label is also weak. This shows that linear separability cannot be applied to classify these samples. Non linear and temporal models need to be designed.

4.2 Experimental Results

Table 4 summarizes final dataset properties after performing preprocessing. Preprocessing results in a clean and well-balanced dataset to be used in the experiment. There are 90,236 samples of task performance in the dataset. Eight selected features describe performance of each sample. Stratified split of 90/10 is performed on the dataset where the training set consists of 81,212 samples, while the test set contains 9,024 samples. Task types are classified as either network based, I O based or compute based. Balance of classes is maintained in both sets. The training set contains 24,076, 28,991 and 28,145 network based, I O based and compute based tasks respectively. The test set contains 2,675, 3,222 and 3,127 samples respectively. Missing and noisy data are eliminated during the process of preprocessing. No missing values and no outliers can be found in any of the sets.

Figure 5 shows the relative importance of the input features used in the proposed approach. It provides information on how different cloud performance metrics contribute to task type

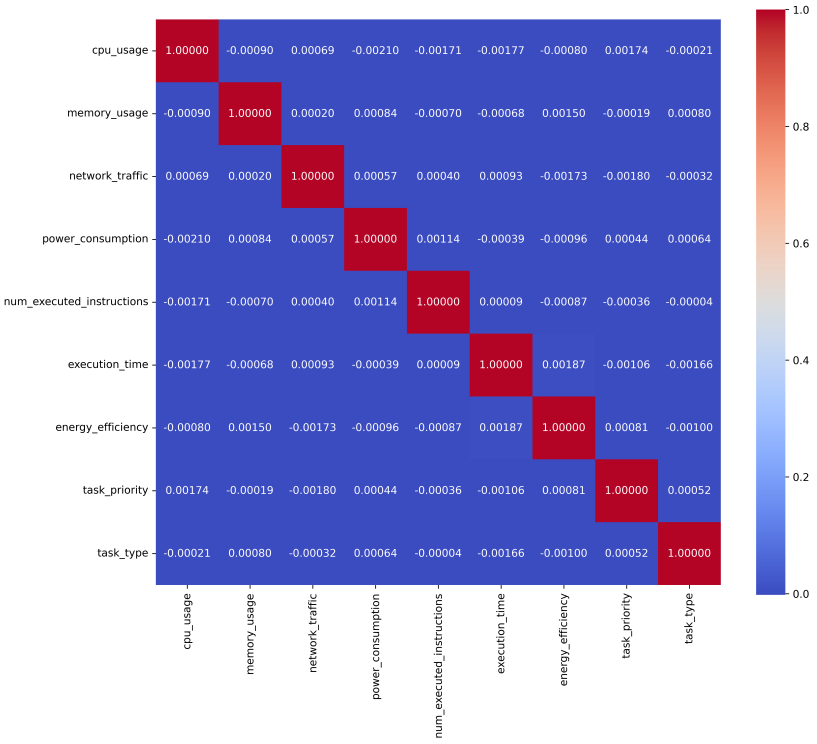

Figure 4: Correlation matrix of the selected metrics and the target variable.

Table 4: Dataset summary with label distribution for cloud task types

	Data (%)	Number of Features	Number of Data	Label Distribution (task_type)			Missing/Noisy Data
				Network-based	I/O-based	Compute-based	
Training	90%	8	81212	24076	28991	28145	0% / 0%
Test	10%	8	9024	2675	3222	3127	0% / 0%
Total	100%	8	90236	26751	32213	31272	0% / 0%

prediction. It also supports model explainability. Among all features, memory usage is the most influential factor. Network traffic and execution time follow next. This result indicates that memory-intensive behavior and data transfer patterns play a major role in task type separation in cloud environments. Features such as energy efficiency, CPU usage, and number of executed instructions show moderate importance. These features contribute additional information related to computational load and energy behavior. In contrast, power consumption and task priority show lower influence. These are characteristics which give context, but which lack discriminatory capability when used individually to classify. The above ranking of feature importance demonstrates that the model makes use of intelligible measures of performance rather than hidden internal workings of the model. This is beneficial in terms of transparency. From a practical viewpoint, explainability is valuable for cloud computing because it enables an understanding of which system-level measures have an effect on the decision made by the model. It can therefore assist with effective monitoring resource allocation.

Figure 6 shows the confusion matrix of the X-CNN–BiLSTM model. It provides a detailed evaluation of classification performance across the three task types.The values along the main diagonal are high. This pattern shows that the model correctly classifies most samples in each class. Specifically, 88.4% (2,764 out of 3,127) of network-based tasks are correctly identified.

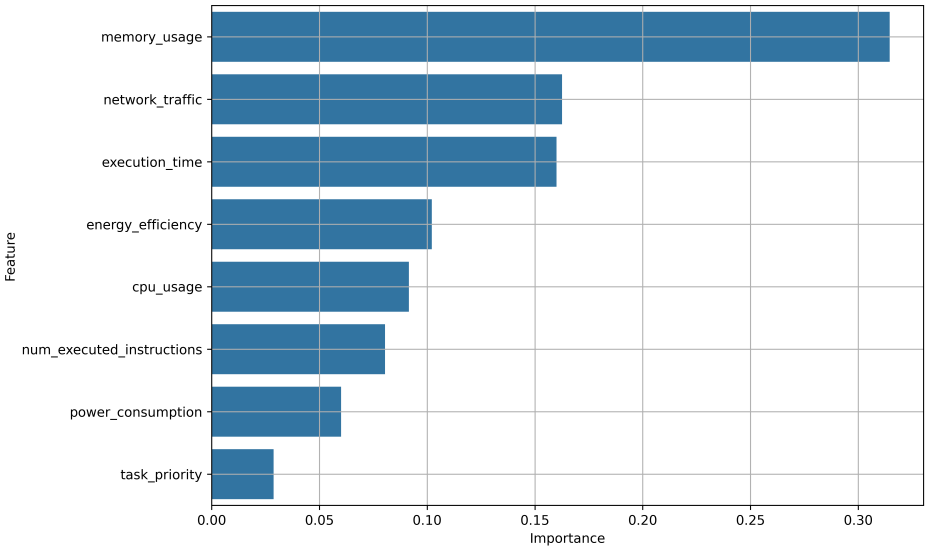

Figure 5: Feature importance analysis

Also, 86.5% (2,786 out of 3,222) of I/O-based tasks are correctly classified. In addition, 84.7% (2,267 out of 2,675) of compute-based tasks are correctly recognized. These results demonstrate balanced prediction performance. The cases of misclassification are minimal. They take place between the network-based and compute-based classes of workloads. In specific, 10.2% of the compute-based workloads (320 samples) are mistakenly predicted to be network-based workloads. The explanation for this fact is the partial overlapping of short-term resource usage like CPU and memory utilization. There are no classes demonstrating significant off-diagonal errors. These findings prove that there is no bias towards one type of tasks. Our results prove the success of the class balancing approach and the focal loss in overcoming imbalance issues. They also prove the stability of predictions made by the proposed approach.

Figure 7 illustrates the training and validation accuracy of our proposed framework across 50 epochs, providing insights into the learning dynamics and generalization behavior of the model. It should be noted that for model monitoring, 10% of the training set is allocated as a validation subset to assess generalization performance across epochs. Both curves exhibit a rapid increase in accuracy during the initial epochs, indicating efficient convergence and effective feature learning driven by the X-CNN–BiLSTM architecture. The validation accuracy follows the training one throughout the training process, with only a minimal and stable gap. It suggests that the model generalizes well to unseen data and does not suffer from overfitting. After 10–15 epochs, accuracies stabilize and converge toward a plateau around the mid-to-high 80% range. It reflects consistent performance gains without oscillations. This stable convergence behavior is attributed to the combination of robust preprocessing, focal loss optimization, dropout regularization, and the absence of excessive model complexity. Overall, the accuracy trends confirm that the proposed framework achieves reliable learning, making it suitable for deployment in dynamic cloud environments where consistent performance across diverse workloads is essential.

Class-wise precision, recall, and F1 score are shown in Figure 8 for three different tasks. This figure compares the performance of X CNN BiLSTM on heterogeneous tasks. It is clear that X CNN BiLSTM demonstrates excellent performance for all classes. F1 scores vary between 0.85 and 0.89, which means that there is balanced precision and recall. I O tasks

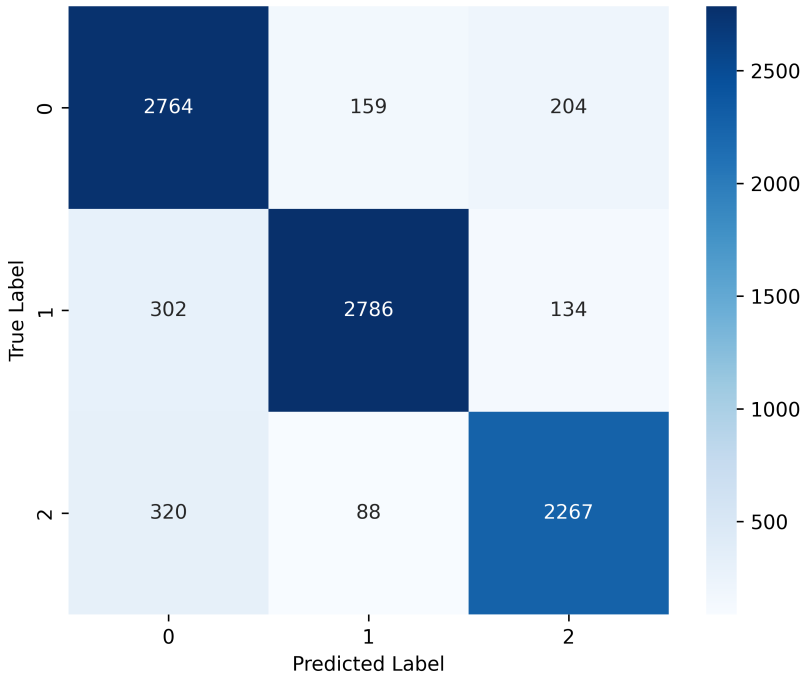

Figure 6: Confusion matrix of the X-CNN–BiLSTM framework

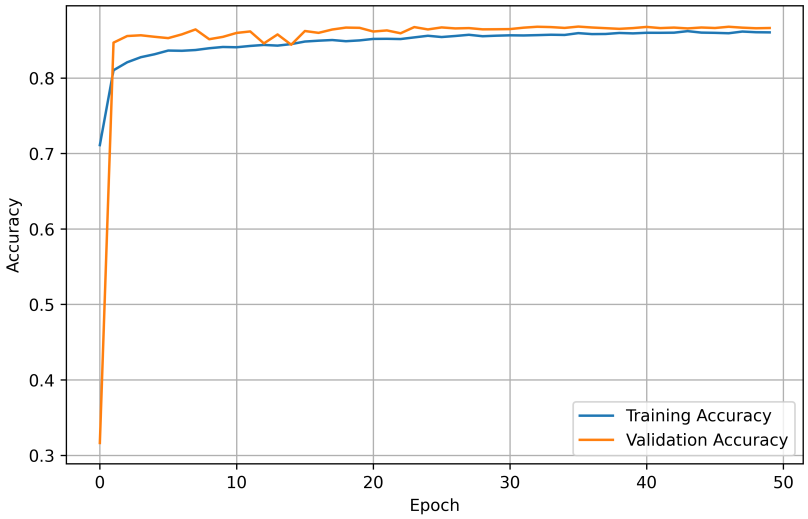

Figure 7: Training and validation accuracy of the X-CNN–BiLSTM framework across training epochs

obtain the best precision with 0.92. They also have the highest F1 score of about 0.89. This proves efficient pattern detection for IO-intensive tasks. Compute tasks get the highest recall rate of about 0.88. This gives them an F1 score of about 0.85. This proves efficient detection of compute tasks in cases where resources overlap. Network tasks get a precision of 0.87 and recall of 0.84. This gives them an F1 score of about 0.86. This proves stable detection of network-based workloads.

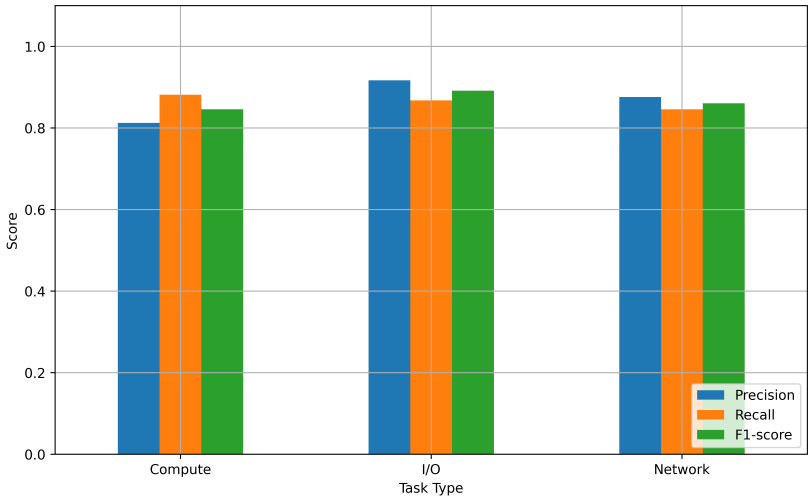

Figure 8: Class-wise precision, recall, and F1-score of X-CNN–BiLSTM for compute, I/O, and network-based task types

4.3 Comparative Analysis

Table 5 presents a comparative evaluation of the proposed approach against three recent methods, including a hybrid CNN–BiLSTM model [15], a standalone LSTM model [16], and the HAPCNN + SMO method [20]. The results show that the proposed model achieves superior performance across all evaluation metrics.

For accuracy, X-CNN–BiLSTM reaches 86.6%. This value is higher than the hybrid CNN–BiLSTM by 6.8 points (about 8.5% improvement), higher than the LSTM model by 12.0 points (about 16.1% improvement), and higher than HAPCNN + SMO by 8.4 points (about 10.7% improvement). Similar improvements are observed in precision, recall, and F1-score, where gains of approximately 6 –12% are recorded.

These results show that the combination of a structured preprocessing pipeline and a CNN–BiLSTM architecture improves the learning of discriminative task patterns from noisy and heterogeneous cloud metrics. The comparison also shows the effect of individual design choices. The hybrid CNN–BiLSTM model improves over the standalone LSTM model, with an accuracy gain of about 5.2%. However, it still performs worse than X-CNN–BiLSTM. This result shows the importance of preprocessing steps such as denoising, class balancing, and normalization in improving learning stability and generalization. The lower performance of HAPCNN + SMO indicates that models designed mainly for resource allocation and scheduling do not transfer well to task type prediction without explicit temporal modeling and imbalance-aware learning. By combining spatial feature extraction through CNN, bidirectional temporal modeling through BiLSTM, and imbalance-aware optimization using focal loss, the proposed X-CNN–BiLSTM achieves consistent improvement across all metrics. This makes it suitable for real-world cloud environments, where accurate and stable task classification is required for scheduling and resource management.

The proposed model shows excellent performance in predicting cloud task types. The accuracy is as high as 86.6%. Precision, recall, and F1 scores are balanced with heterogeneous workloads. The use of spatial and temporal learning together with structured preprocessing contributes to better modeling of the complex behavior of tasks in multi-tenant cloud environments. The proposed method tackles the issues of heterogeneity of the workloads,

Table 5: Comparative evaluation of the proposed framework against recent approaches

Approach	Accuracy (%)	Precision (%)	Recall (%)	F-Score (%)
X-CNN–BiLSTM	**86.6**	**86.8**	**86.6**	**86.7**
Hybrid CNN–BiLSTM	79.8	80.4	79.8	79.9
LSTM	74.6	75.2	74.6	74.7
HAPCNN + SMO	78.2	78.7	78.2	78.3

class imbalance, noise in the metric, and overlap of resource consumption patterns. Tasks of compute, I/O, and network types can be distinguished reliably. It enables workload-aware scheduling, resource allocation, and QoS-driven decisions. The moderate complexity model is chosen. The attention module is removed from the architecture to decrease computational costs and increase the feasibility for deployment close to the real time. The ablation study in Table 6 demonstrates the influence of the individual components and architecture order on the results. The accuracy of the BiLSTM-only architecture is 74.6%. The accuracy of the CNN-only architecture is 77.1%. The accuracy of the reversed BiLSTM and CNN model is 78.5%. The accuracy of the standard CNN-BiLSTM model is 79.8%. The accuracy of the proposed X-CNN-BiLSTM model is 86.6%.

Table 6: Impact of architectural components and layer ordering on task type prediction

Configuration	Accuracy (%)	Precision (%)	Recall (%)	F-Score (%)
BiLSTM only (full preprocessing)	74.6	75.2	74.6	74.7
CNN only (full preprocessing)	77.1	77.6	77.1	77.2
BiLSTM $\rightarrow$ CNN (inverse order)	78.5	79.0	78.5	78.6
Hybrid CNN–BiLSTM (no focal loss)	79.8	80.4	79.8	79.9
X-CNN–BiLSTM	**86.6**	**86.8**	**86.6**	**86.7**

Despite strong performance, the proposed approach is evaluated on a single public cloud performance dataset. Its generalization to edgeвЂ“cloud environments requires further study. The current implementation also uses offline training. Extension toward online or incremental learning may improve adaptability to rapidly changing cloud workloads. In addition, although a balance is achieved between accuracy and efficiency, further optimization is required to evaluate scalability and computational cost in large-scale, high-throughput cloud deployments.

Conclusion

This paper presents X-CNN–BiLSTM, an explainable hybrid DL approach for task type prediction in cloud computing environments using performance metrics. The approach integrates a structured multi-stage preprocessing pipeline with a CNN–BiLSTM architecture. It captures localized spatial patterns and bidirectional temporal dependencies in heterogeneous workloads. Experimental evaluation shows that the model achieves 86.6% accuracy with balanced precision, recall, and F1-score. It outperforms baseline methods, including standalone BiLSTM, CNN, inverse-order hybrid models, and recent scheduling-oriented approaches. Comparative and ablation studies confirm that the preprocessing strategy and architectural ordering contribute significantly to performance gains. From a practical perspective, the approach addresses major cloud computing challenges such as workload heterogeneity, noisy monitoring

data, and class imbalance. It supports workload-aware scheduling, adaptive resource provisioning, and QoS-driven decision-making in dynamic multi-tenant cloud systems.

Acknowledgments

The authors declare a minor use of AI for language editing. Authors declare possible prior coauthorships with editorial board members of the journal.

References

[1] M. Hosseini Shirvani, "A survey study on task scheduling schemes for workflow executions in cloud computing environment: classification and challenges," *J. Supercomput.*, vol. 80, pp. 9384–9437, 2024, doi: 10.1007/s11227-023-05806-y.

[2] N. Devi, S. Dalal, K. Solanki, S. Dalal, U. K. Lilhore, S. Simaiya, and N. Nuristani, "A systematic literature review for load balancing and task scheduling techniques in cloud computing," *Artif. Intell. Rev.*, vol. 57, p. 276, 2024, doi: 10.1007/s10462-024-10925-w.

[3] D. Alsadie, "Advancements in heuristic task scheduling for IoT applications in fog-cloud computing: challenges and prospects," *PeerJ Comput. Sci.*, vol. 10, p. e2128, 2024, doi: 10.7717/peerj-cs.2128.

[4] F. S. Prity, K. M. A. Uddin, and N. Nath, "Exploring swarm intelligence optimization techniques for task scheduling in cloud computing: algorithms, performance analysis, and future prospects," *Iran J. Comput. Sci.*, vol. 7, pp. 337–358, 2024, doi: 10.1007/s42044-023-00163-8.

[5] E. L. Pimentel, C. A. de Souza, and C. B. Westphall, "Detecting attacks in Fog and cloud computing environments using Deep Learning: a systematic literature review," *Comput. Netw.*, vol. 264, p. 111269, 2025, doi: 10.1016/j.comnet.2025.111269.

[6] M. Kirti, A. K. Maurya, and R. S. Yadav, "Fault-tolerance approaches for distributed and cloud computing environments: a systematic review, taxonomy and future directions," *Concurr. Comput. Pract. Exp.*, vol. 36, no. 13, Art. no. e8081, 2024, doi: 10.1002/cpe.8081.

[7] M. R. Rezaee, N. A. W. Abdul Hamid, M. Hussin, and Z. A. Zukarnain, "Fog Offloading and Task Management in IoT-Fog-Cloud Environment: review of algorithms, networks, and SDN application," *IEEE Access*, vol. 12, pp. 39058–39080, 2024, doi: 10.1109/ACCESS.2024.3375368.

[8] A. Rossi, A. Visentin, D. Carraro, S. Prestwich, and K. N. Brown, "Forecasting workload in cloud computing: towards uncertainty-aware predictions and transfer learning," *Cluster Comput.*, vol. 28, p. 258, 2025, doi: 10.1007/s10586-024-04933-2.

[9] S. Alharthi, A. Alshamsi, A. Alseiari, and A. Alwarafy, "Auto-Scaling Techniques in Cloud Computing: issues and research directions," *Sensors*, vol. 24, no. 17, p. 5551, 2024, doi: 10.3390/s24175551.

[10] S. Yousefi, S. Najjar-Ghabel, and Z. S. Owaid, "A Supervised Learning-based Framework for Failure Detection in Toy Cars Using Acoustic Signal Analysis," in *Proc. 2025 IEEE 7th Symp. Comput. & Informatics (ISCI)*, 2025, pp. 76–81, doi: 10.1109/ISCI64230.2025.11167791.

[11] S. Yousefi, F. Derakhshan, and H. Karimipour, "Applications of Big Data Analytics and Machine Learning in the Internet of Things," in *Handbook of Big Data Privacy*, K.-K. R. Choo and A. Dehghantanha, Eds., Cham: Springer International Publishing, 2020, pp. 77–108, doi: 10.1007/978-3-030-38557-6_5.

[12] Z. A. Haider, A. Zeb, T. Rahman, F. M. Khan, I. U. Khan, Q. Sohail, H. Bilal, M. A. Khan, and I. Ullah, "Optimizing Cloud Security with a Hybrid BiLSTM-BiGRU Model for Efficient Intrusion Detection," *ICCK Trans. Sens. Commun. Control*, vol. 2, no. 2, p. 106, 2025, doi: 10.62762/TSCC.2024.433246.

[13] Y. Wang and X. Yang, "Cloud Computing Energy Consumption Prediction Based on Kernel Extreme Learning Machine Algorithm Improved by Vector Weighted Average Algorithm," in *Proc. 5th Int. Conf. Artif. Intell. Ind. Technol. Appl. (AIITA)*, 2025. [Online]. Available: https://arxiv.org/abs/2503.04088

[14] S. Lajili, Z. Brahmi, and M. N. Omri, "ML WPStreamCloud: ML-based Workload Prediction and Task Clustering for Efficient Stream Application Offloading in Heterogeneous Edge and Cloud Environments," *Procedia Comput. Sci.*, vol. 246, pp. 1527–1537, 2024, doi: 10.1016/j.procs.2024.09.610.

[15] A. Khan, F. Ullah, D. Shah, M. H. Khan, S. Ali, and M. Tahir, "EcoTaskSched: a hybrid machine learning approach for energy-efficient task scheduling in IoT-based fog-cloud environments," *Sci. Rep.*, vol. 15, p. 12296, 2025, doi: 10.1038/s41598-025-96974-9.

[16] A. Ali, I. Ullah, S. K. Singh, A. Sharafian, W. Jiang, H. I. Sherazi, and X. Bai, "Energy-Efficient Resource Allocation for Urban Traffic Flow Prediction in Edge-Cloud Computing," *Int. J. Intell. Syst.*, vol. 2025, Art. no. 1863025, 2025, doi: 10.1155/int/1863025.

[17] K. Seshadri, K. Sindhu, S. N. Bhattu, and C. Kollengode, "Design and Evaluation of a Hierarchical Characterization and Adaptive Prediction Model for Cloud Workloads," *IEEE Trans. Cloud Comput.*, vol. 12, no. 2, pp. 712–724, 2024, doi: 10.1109/TCC.2024.3393114.

[18] V. S. Salanke, V. N. Kowshik, H. P. G, M. V, and P. H, "Task Failure Prediction and Migration in Cloud Environment," in *Proc. 2024 1st Int. Conf. Commun. Comput. Sci. (InCCCS)*, Bengaluru, India, 2024, pp. 1–6.

[19] S. Sangani, R. Patil, and R. H. Goudar, "Efficient algorithm for error optimization and resource prediction to mitigate cost and energy consumption in a cloud environment," *Int. J. Inf. Technol.*, vol. 16, no. 4, pp. 2187–2197, 2024, doi: 10.1007/s41870-024-01732-1.

[20] S. Gurusamy and R. Selvaraj, "Resource allocation with efficient task scheduling in cloud computing using hierarchical auto-associative polynomial convolutional neural network," *Expert Syst. Appl.*, vol. 249, p. 123554, 2024, doi: 10.1016/j.eswa.2024.123554.

[21] W.-N. Mohd-Isa, J. Joseph, N. Hashim, and N. Salih, "Enhancement of digitized X-ray films using Contrast-Limited Adaptive Histogram Equalization (CLAHE)," *F1000Res.*, vol. 10, p. 1051, 2021, doi: 10.12688/f1000research.73236.1.

[22] D. Gogri, "Remote Browser Isolation: a path to Zero Trust Security in the modern enterprise," *Int. J. Sci. Res. (IJSR)*, vol. 12, no. 2, pp. 1766–1772, 2023, doi: 10.21275/SR230204114347.

[23] D. Bakhronova, "Intelligent Information Security System for Language and History Education Using Machine Learning-based Intrusion Detection Algorithm," *J. Internet Serv. Inf. Secur.*, vol. 15, no. 1, pp. 520–529, 2025, doi: 10.58346/JISIS.2025.I1.034.

[24] Y. Saidu, S. M. Shuhidan, D. A. Aliyu, I. Abdul Aziz, and S. Adamu, "Convergence of Blockchain, IoT, and AI for Enhanced Traceability Systems: a comprehensive review," *IEEE Access*, vol. 13, pp. 16838–16865, 2025, doi: 10.1109/ACCESS.2025.3528035.

[25] Abdurraziq, "Cloud Computing Performance Metrics," dataset, Kaggle, 2023. [Online]. Available: https://www.kaggle.com/datasets/abdurraziq01/cloud-computing-performance-metrics

[26] B. Yousefimehr, M. Ghatee, and R. Razavi-Far, "Multi-Stage Self-Distillation for Class-Imbalanced Anomaly Detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, 2026, doi: 10.1109/TETCI.2026.3671131.

[27] B. Yousefimehr, M. Ghatee, and R. Razavi-Far, "Multi-teacher knowledge distillation framework for lightweight anomaly detection," *Neural Netw.*, vol. 195, p. 108267, 2026, doi: 10.1016/j.neunet.2025.108267.

[28] M. Kozhanov, C. Mako, V. Poser, G. Eigner, and A. Mosavi, "Knowledge Augmented Generation for Curriculum Planning," *Eurasian J. Math. Comput. Appl.*, vol. 14, no. 1, pp. 17–34, 2026, doi: 10.32523/2306-6172-2026-14-1-17-34.

Shamim Yousefi
Department of Computer Engineering,
University of Mohaghegh Ardabili,
Ardabil, Iran, Email: Sh.yousefi@uma.ac.ir.

Alimzhan Igenbayev
Obuda University, Budapest, Hungary.

Zahraa Haseebsalman
Department of Computer Engineering,
University of Mohaghegh Ardabili, Ardabil, Iran,

Zhasulan Akylaev
Farabi University, Almaty, Kazakhstan

Received 04.05.2025 , Accepted 02.04.2026 Available online 30.06.2026.