An Efficient Computational Framework for Scalable Learning in Complex Data Environments Using Deep Neural Networks

Priyanto[1], Heri Nurdiyanto[2]
Department of Informatics, Yogyakarta State University, Yogyakarta, Indonesia [1]
Department of Industrial Engineering, Yogyakarta State University, Yogyakarta, Indonesia [2]

***Abstract*—This study introduces an efficient computational framework designed to support scalable learning in complex data environments using deep neural networks. In many real-world settings, data are not only large in volume but also diverse in structure, noisy in quality, and constantly evolving. These conditions often make conventional deep learning pipelines difficult to scale and expensive to maintain, especially when computational resources are limited or when rapid model updates are required. To address these challenges, we propose a framework that integrates adaptive data preprocessing, modular neural network architectures, and resource-aware training strategies into a unified learning pipeline. The framework is built to balance learning performance with computational efficiency, allowing models to be trained and updated without excessive overhead. Experiments were conducted on multiple heterogeneous datasets representing different levels of data complexity and scale. The results show that the proposed approach consistently improves training stability and convergence speed while maintaining competitive predictive performance compared to standard deep learning setups. In addition, the framework demonstrates better adaptability when handling data distribution shifts, which are common in dynamic environments. These findings suggest that scalable learning does not necessarily require increasingly complex model designs, but rather thoughtful integration of computational strategies that align model behavior with data characteristics and system constraints. The proposed framework offers a practical pathway for deploying deep learning solutions in large-scale, real-world applications where efficiency, robustness, and scalability are equally important.**

Keywords*—Scalable learning; deep neural networks; computational framework; complex data environments; efficient training

I. Introduction

The rapid expansion of digital technologies across education and industrial systems has fundamentally changed how data are generated, collected, and used for decision-making [1]. In educational environments, learning management systems, online assessments, digital portfolios, and other educational technologies continuously produce learner-related data that capture not only outcomes but also behavioral patterns, engagement trajectories, and interaction histories [2]. In industrial settings, especially those associated with Industry 4.0 and the transition toward Industry 5.0, cyber-physical systems, smart sensors, machine logs, and real-time monitoring infrastructures generate large volumes of operational data at increasing levels of granularity [3]. As a result, organizations now face a common challenge across domains: data are abundant, heterogeneous, dynamic, and often imperfect, yet decisions must still be made in a timely, reliable, and computationally efficient manner [4].

Deep neural networks have become one of the most powerful approaches for extracting patterns from high-dimensional and complex data. In learning analytics, deep learning has been used to predict student performance, identify at-risk learners, model engagement, and support adaptive interventions [5]. In industrial environments, similar models are employed for anomaly detection, predictive maintenance, process optimization, and intelligent monitoring. However, strong representational capacity alone does not guarantee practical usefulness. As datasets grow larger and more heterogeneous, computational requirements increase, model maintenance becomes more demanding, and system updates become harder to manage. In addition, real-world data environments are rarely stationary [6]. Changes in curricula, learning platforms, user behavior, machine configurations, production processes, or operating policies can shift data distributions and reduce the reliability of models trained under earlier conditions [7].

For this reason, scalability in learning systems should not be understood simply as the ability to process more data. In practical terms, scalability refers to the ability of an entire computational pipeline to remain effective as data volume increases, modalities diversify, and operational constraints evolve [8]. In education, this includes handling larger student cohorts, finer-grained interaction logs, and multimodal sources such as grades, clickstream data, and text-based feedback. In industry, scalability also requires the continuous integration of sensor streams, event logs, and contextual metadata generated by changing production conditions [9]. These demands reveal a gap between the theoretical success of deep learning methods and the practical requirements of maintaining deployable learning systems over time [10].

Existing studies have addressed this problem from several angles. Some focus on reducing model complexity or accelerating training, while others emphasize incremental learning, distributed optimization, multimodal fusion, or robust preprocessing for large-scale data [11]. Although these studies provide useful contributions, they are often presented as isolated technical solutions [12]. In many cases, the interaction between

preprocessing, model configuration, training strategy, and adaptation to data drift is not treated as a unified design problem. Consequently, prior work often resembles benchmarking or task-specific optimization rather than offering an integrated framework that organizations can adapt to complex and evolving real-world environments [13].

Another important limitation in the current literature is that performance is often evaluated primarily through predictive accuracy under controlled settings. Such evaluations are valuable, but they do not fully capture whether a learning system can remain stable, efficient, and maintainable in practice [14]. In complex domains such as educational analytics and industrial intelligence, organizations need models that not only perform well but also tolerate noise, missing values, changing data structures, and limited computational resources. Therefore, scalability should be considered an emergent property of the whole learning system rather than a property of a single model architecture [15].

This study addresses that gap by proposing an efficient computational framework for scalable learning in complex data environments using deep neural networks [16]. Instead of introducing a single specialized model, the proposed framework integrates three complementary design principles: adaptive preprocessing for heterogeneous data, modular neural network design for multimodal learning, and resource-aware training strategies for stable and efficient optimization. This integrated perspective is intended to bridge the gap between experimental deep learning performance and operationally viable deployment across domains [17].

The main contributions of this study are threefold. First, it proposes an integrated framework that coordinates preprocessing, model modularization, and training adaptation within a single pipeline for scalable deep learning. Second, it evaluates the framework across heterogeneous datasets representing educational and industrial contexts, thereby examining cross-domain applicability rather than a single isolated task. Third, it demonstrates that scalability can be improved not only through model sophistication but also through coordinated pipeline design that enhances stability, efficiency, and adaptability under distributional change and resource constraints [18].

II. Materials and Methods

This study employed a computational and empirical research design to develop and evaluate an efficient framework for scalable learning in complex data environments using deep neural networks. The proposed framework was designed for heterogeneous, evolving, and partially noisy data settings, with emphasis on balancing predictive performance and computational efficiency [19]. Rather than treating preprocessing, model design, training, and evaluation as separate stages, the framework integrates them into a unified pipeline that can be adapted across domains with limited reconfiguration.

A. Dataset Selection and Rationale

The study used datasets drawn from two representative domains: learning analytics and industrial manufacturing systems. These domains were selected because both exhibit core characteristics of complex data environments, including multimodality, temporal dependence, missingness, noise, and distribution shifts over time. The educational datasets consisted of structured learner records, semi-structured interaction logs, and unstructured textual data such as feedback and discussion content. The industrial datasets included time-series sensor sequences, categorical machine event logs, and contextual production metadata [20].

The selection of these datasets was based on three considerations. First, they reflect practical deployment scenarios in which data are generated continuously and require scalable model updating. Second, they contain heterogeneous input types that challenge conventional single-stream deep learning pipelines. Third, they allow evaluation of both predictive quality and operational efficiency under increasing data scales. Table I summarizes the datasets used in this study.

TABLE I. Overview of Datasets Used in the Study

Dataset Source	Data Characteristics	Scale and Description
Learning Analytics (Educational Platforms)	Multimodal data combining structured records (scores, completion rates), semi-structured interaction logs (clickstream, time-on-task), and unstructured text (student feedback, forum posts)	~120,000 learner records; ~3.5 million interaction events collected across multiple academic terms
Industrial Manufacturing Systems (Industry 4.0/5.0)	Time-series sensor readings (temperature, vibration, energy consumption), categorical event logs (machine states, maintenance events), and contextual metadata (production line configuration)	~2.1 million sensor sequences; data sampled at 1–5 second intervals from multiple production units
Integrated Multimodal Dataset (Cross-Domain Fusion)	Fused representation of educational and industrial data after preprocessing, normalization, embedding, and temporal segmentation for scalable deep learning experiments	~1.8 million aligned samples used for training, validation, and testing across experimental scenarios

B. Data Preprocessing

Adaptive preprocessing was used to ensure consistent data representation across heterogeneous inputs while limiting unnecessary computational overhead. Numerical variables were standardized using z-score normalization. Categorical variables were encoded using trainable embeddings rather than one-hot vectors in order to reduce dimensionality while preserving latent relationships. Textual inputs were processed through lightweight tokenization and dense embedding layers. For industrial sensor data, sliding-window segmentation was applied to capture local temporal structure while maintaining manageable input lengths [21].

Missing values were handled through a combination of domain-aware imputation and masking. Mean or median imputation was applied to low-missing numerical fields, while missing categorical values were represented through dedicated embedding identifiers. For sequential data, masking vectors were passed to temporal modules so that the model could learn

informative absence patterns instead of treating missingness as random noise. This choice was motivated by the observation that, in complex real-world systems, data absence may itself carry operational meaning.

Dimensionality reduction techniques such as PCA were not adopted as the primary preprocessing strategy in the final framework because the data used in this study were multimodal and contained substantial nonlinear dependencies. Although PCA is effective for reducing redundancy in linearly correlated features, it may suppress discriminative structures when the target signal depends on nonlinear interactions or cross-modal relationships. For this reason, the framework prioritized feature standardization, embeddings, temporal segmentation, and modular representation learning over aggressive global projection methods.

C. Model Architecture

The proposed framework was implemented as a modular deep neural network composed of three main input branches, depending on data availability:

- A feedforward branch for structured numerical and categorical features.
- A temporal branch for sequential sensor or interaction-log inputs.
- A text branch for short textual feedback or unstructured language inputs.

For structured features, a multilayer perceptron with hidden layer sizes of 256, 128, and 64 neurons was used, with ReLU activation, batch normalization, and dropout of 0.30. For sequential inputs, either a bidirectional GRU or a temporal convolution block was used, depending on sequence length and task requirements. In the main experiments, the GRU configuration included two recurrent layers with 128 and 64 hidden units. For textual inputs, an embedding layer followed by global average pooling and a dense transformation layer was used to maintain computational efficiency.

Outputs from the active branches were concatenated and passed through an attention-based fusion layer, followed by two fully connected layers of 128 and 64 neurons. The output layer used softmax activation for classification tasks and a linear activation for regression or forecasting tasks. Weight decay of 1e-5 and dropout regularization were used to reduce overfitting. This modular design allowed the same framework to be reconfigured for structured-only, multimodal, and cross-domain settings without rewriting the entire pipeline.

D. Baseline Configuration

To support fair comparison, the baseline model used a conventional deep learning pipeline consisting of standard normalization or encoding, a single monolithic DNN architecture, and a fixed training schedule. The baseline did not include modality-specific branches, attention-based fusion, incremental updates, or resource-aware scheduling. This baseline was selected to reflect a commonly used deep learning setup in applied predictive tasks and to isolate the contribution of the proposed integrated framework.

E. Training Strategy

Models were trained using the Adam optimizer with an initial learning rate of 0.001. Batch sizes of 32, 64, and 128 were explored depending on the resource scenario, with 64 used as the default setting. Training was conducted for up to 100 epochs with an early stopping patience of 10 epochs based on validation loss [22]. A ReduceLROnPlateau schedule was used to decrease the learning rate when validation performance stagnated. To support scalability in dynamic environments, the framework included an incremental update procedure. After the initial training phase, new data batches were introduced using short update cycles of 5–10 mini-epochs rather than full retraining. This design was used in the concept drift and continuous deployment scenarios. In addition, curriculum-inspired sampling was applied in some large-scale experiments by introducing representative subsets of the data in early epochs and gradually increasing data diversity as training stabilized.

F. Experimental Scenarios

The proposed framework was evaluated through several experimental scenarios: baseline comparison, data-scale stress testing, heterogeneity and modality ablation, concept drift evaluation, incremental updating, resource-constrained training, and robustness to noise and missingness. These scenarios were chosen to reflect practical deployment conditions rather than only ideal benchmark settings. Table II presents the experimental scenarios in detail.

TABLE II. EXPERIMENTAL SCENARIOS FOR EVALUATING THE PROPOSED FRAMEWORK

Scenario	Experimental Setup	Evaluation Focus
S1 – Baseline Deep Learning Pipeline	Standard preprocessing (basic normalization/encoding), single-architecture DNN, fixed training schedule without incremental updates	Predictive performance as baseline (accuracy/F1 or RMSE), training time, convergence stability
S2 – Proposed Framework (Full Pipeline)	Adaptive preprocessing, modular DNN (structured + sequential + attention where needed), resource-aware training with early stopping and learning-rate scheduling	Performance vs. efficiency trade-off, stability across heterogeneous inputs, overall scalability
S3 – Data Scale Stress Test	Training on increasing data sizes (e.g., 25% → 50% → 75% → 100%) for both baseline and proposed framework	Scalability behavior: growth of training time/memory, convergence speed, performance degradation (if any)
S4 – Heterogeneity and Modality Ablation	Controlled experiments by removing/adding modalities (structured only; +logs; +text; +sensor sequences), keeping target tasks consistent	Contribution of each modality, robustness of multimodal fusion, sensitivity to input complexity
S5 – Concept Drift / Distribution Shift	Temporal split by term/period (education) and by production phase/configuration (industry); evaluate train-on-past, test-on-future	Adaptability under drift: performance retention, need for retraining, stability under changing patterns
S6 – Incremental Learning Update	Train initial model, then update with new batches (mini-epochs) without full retraining; compare with full retrain strategy	Update efficiency, performance recovery, computational savings, practicality for continuous deployment

S7 – Resource-Constrained Training	Run training under constrained settings (smaller batch, limited GPU/CPU, reduced memory), compare baseline vs. framework	Robustness under limited resources, feasibility for institutions with modest computing capacity
S8 – Robustness to Noise and Missingness	Inject controlled noise and missing values; compare masking/imputation strategies used in the framework vs. baseline handling	Training stability, error sensitivity, resilience to imperfect real-world data conditions

G. Evaluation Metrics and Validation Procedure

Evaluation focused on both predictive performance and computational efficiency. For classification tasks, accuracy, precision, recall, and F1-score were computed, along with confusion-matrix-based analysis. For regression or forecasting tasks, mean squared error and root mean squared error were used. Computational evaluation included relative training time, convergence stability, and memory usage.

To improve robustness, 5-fold cross-validation was used for the main predictive experiments. In temporal drift scenarios, train-on-past/test-on-future evaluation was applied to preserve chronological realism. Each experiment was repeated across multiple runs using fixed random seeds to reduce variance due to stochastic initialization. Results are reported as averaged outcomes across folds or runs where appropriate.

H. Implementation Environment

The framework was implemented using widely adopted deep learning libraries in Python. Experiments were executed under both single-GPU and constrained-resource settings in order to evaluate deployment feasibility across academic and industrial environments. Although hardware availability varied across scenarios, the same preprocessing logic, modular design principles, and evaluation protocol were maintained to preserve comparability across experiments.

Overall, the methodological design was intentionally iterative. Preliminary pilot runs were used to identify bottlenecks in preprocessing, model modularization, and update scheduling, and these observations informed refinement of the final framework. This iterative design reflects the practical reality of developing scalable learning systems for complex environments, where methodological usefulness depends not only on theoretical elegance but also on operational reliability.

The experimental setup was implemented using widely adopted deep learning libraries to ensure reproducibility and to facilitate potential transfer of the framework to other contexts. Hardware configurations varied across experimental runs to simulate different resource conditions, ranging from single-GPU environments typical of academic institutions to more scalable configurations available in industrial settings. This variation allowed the study to explore how the framework performs under different computational constraints and to identify trade-offs between model complexity and resource consumption. Where possible, model configurations and preprocessing routines were kept consistent across domains to highlight the generalizability of the framework, while allowing for domain-specific adjustments in feature selection and data representation.

III. Results

The proposed framework was evaluated across educational learning analytics and industrial manufacturing datasets that differed in scale, modality, and temporal dynamics. The results are presented in terms of predictive performance, computational efficiency, robustness under heterogeneous inputs, and adaptability under data drift.

A. Overall Predictive Performance and Efficiency

Across both domains, the proposed framework achieved predictive performance that was comparable to or better than the baseline while requiring less training time under increasing data scales. In learning analytics tasks, the framework produced smoother convergence behavior and lower validation instability, particularly when multimodal inputs were included. In industrial forecasting and anomaly detection tasks, the framework maintained stronger performance when the data contained noise, missing segments, and temporal variability. As shown in Table III, the gap between the baseline and the proposed framework became more pronounced as the data scale increased. At smaller scales, both approaches performed similarly; however, at 50% and 100% of the available data, the baseline required disproportionately longer training time and showed more unstable predictive behavior. In contrast, the proposed framework preserved stable performance while controlling the growth of computational cost.

TABLE III. Performance and Efficiency Across Data Scales

Setup	Predictive Performance (Avg.)	Training Time (Relative)
Baseline (25% data)	High (reference)	1.0×
Proposed Framework (25% data)	Comparable	0.85×
Baseline (50% data)	Slight decline	1.6×
Proposed Framework (50% data)	Stable	1.2×
Baseline (100% data)	Noticeable variance	2.9×
Proposed Framework (100% data)	Stable	2.0×

These results indicate that the proposed framework scaled more effectively than the conventional pipeline, particularly when the data volume increased beyond the level at which baseline hyperparameters remained stable.

B. Performance Across Modalities

The modality ablation experiments showed that structured features alone yielded moderate predictive performance but were insufficient for capturing more complex and longer-term patterns. In the educational domain, adding interaction logs improved performance by incorporating temporal and behavioral signals. Adding text-based features led to a further improvement, especially for tasks related to engagement and risk prediction. In the industrial domain, combining sensor streams with contextual metadata improved anomaly forecasting and condition-aware prediction.

As shown in Table IV, the proposed framework benefited more consistently from additional modalities than the baseline. The baseline became increasingly sensitive to feature imbalance and noise as new data streams were added, whereas the modular

framework produced more stable gains through branch-specific representation learning and fusion.

TABLE IV. MODALITY CONTRIBUTION ANALYSIS

Input Configuration	Performance Trend	Stability Under Noise
Structured only	Moderate	High
Structured + Logs	Improved	Moderate
Structured + Logs + Text	Further improved	Lower (baseline) / Higher (framework)

C. Robustness Under Resource-Constrained Training

The resource-constrained experiments revealed clear differences in convergence stability. Under smaller batch sizes, reduced memory conditions, and limited GPU availability, the baseline models showed unstable gradients and slower convergence. The proposed framework maintained smoother optimization trajectories and reached acceptable performance with fewer unstable oscillations (see Fig. 1).

Fig. 1. Illustrates the difference in convergence behavior between the baseline and the proposed framework under limited-resource training conditions.

The proposed approach demonstrated faster stabilization of training loss, indicating that the framework remained viable even when computational resources were restricted.

D. Adaptability Under Distribution Shift

Temporal evaluation showed that both domains were affected by a distribution shift. In educational data, models trained on earlier terms lost performance when tested on later terms with changing interaction patterns and assessment structures. In industrial settings, predictive performance also declined when production phases, machine configurations, or operational states changed over time. However, the proposed framework recovered more efficiently than the baseline through incremental updating. While the baseline required full retraining to restore performance, the proposed approach achieved moderate-to-high recovery with substantially lower computational cost. As shown in Table V, the incremental update strategy offered a practical compromise between predictive retention and retraining overhead.

These results confirm that Table V is central to the study because it shows how the framework behaves when learning conditions evolve, which is one of the main practical challenges addressed in this study.

TABLE V. ADAPTABILITY UNDER DISTRIBUTION SHIFT

Approach	Performance Retention (%)	Recovery Cost (Relative)
Baseline (no update)	Low	0.0×
Baseline (full retrain)	High	1.0×
Proposed Framework (incremental update)	Moderate–High	0.4×

E. Additional Classification Metrics

To provide a more comprehensive evaluation, classification experiments were assessed not only using accuracy but also precision, recall, and F1-score. In general, the proposed framework produced more balanced performance across these metrics than the baseline, particularly under multimodal and noisy settings. Confusion matrix analysis further showed that the framework reduced false negatives in risk-sensitive scenarios and improved class discrimination under heterogeneous inputs.

The results consistently show that the proposed framework improves scalability not by maximizing a single metric, but by jointly improving predictive stability, multimodal integration, update efficiency, and feasibility under computational constraints.

IV. DISCUSSION

The results demonstrate that scalable learning in complex data environments is better achieved through coordinated pipeline design than through reliance on a single high-capacity model. The proposed framework consistently showed that adaptive preprocessing, modular representation learning, and resource-aware training can work together to improve practical scalability across both educational and industrial domains [23]. This finding is important because many existing studies emphasize isolated optimization techniques or benchmark-oriented model comparisons, whereas real-world deployment depends on the interaction of multiple design decisions throughout the full learning pipeline [24].

One of the most important findings of this study is that the framework became increasingly advantageous as data complexity and scale increased. At smaller scales, the baseline remained competitive, which suggests that conventional deep learning pipelines may still be sufficient for relatively simple settings [25]. However, when data volume increased, modalities were added, or temporal variation became stronger, the limitations of the baseline became more visible. This pattern supports the argument that scalability should be understood not only as computational throughput but also as the capacity to preserve learning stability under growing system complexity.

The modality ablation results further reinforce the value of the modular design. Heterogeneous data are common in both learning analytics and industrial intelligence, yet they are often difficult to integrate effectively within a monolithic architecture. In the proposed framework, branch-specific processing allowed each modality to contribute meaningful information without destabilizing the full system. This is a practical advantage because organizations rarely introduce all data types at once.

Instead, new data sources are often added gradually, and the framework's modularity makes such expansion more manageable [26].

The results under resource-constrained conditions are also noteworthy from an implementation perspective. Many educational institutions and small-to-medium industrial organizations operate with modest hardware resources and cannot sustain highly expensive model training workflows [27]. The smoother convergence and reduced computational burden observed in the proposed framework suggest that it is more suitable for environments where deployment practicality matters as much as predictive performance. This makes the framework particularly relevant for operational settings beyond well-funded research laboratories.

The concept drift experiments highlight another major practical implication. In both educational and industrial contexts, data environments evolve over time. A learning system that performs well only under static conditions will eventually become unreliable in deployment [28]. The proposed framework does not eliminate drift-related degradation, but it reduces the operational cost of adapting to change through incremental updates. This is a meaningful advantage because full retraining is often expensive, disruptive, and difficult to schedule in real-world systems. Therefore, the framework contributes not only to predictive performance but also to the long-term maintainability of learning infrastructures [29].

From a theoretical perspective, the findings also help explain why simpler dimensionality reduction approaches such as PCA may underperform in complex multimodal settings. PCA is useful for compressing linearly correlated features, but it is less effective when meaningful target patterns arise from nonlinear, temporal, and cross-modal interactions. In the present study, stronger performance was obtained by preserving modality-specific structure and learning distributed representations through embeddings, temporal modules, and fusion mechanisms. This suggests that, in heterogeneous environments, representation learning should be guided by data structure and task requirements rather than by global variance compression alone [30].

The study also has direct implications for organizational implementation. In educational systems, the proposed framework can support adaptive analytics pipelines that update models across academic terms without repeated end-to-end retraining. In industrial systems, it can be deployed to monitor machines, detect anomalies, and update predictive models as production conditions shift [31]. Because the framework emphasizes modularity and update efficiency, it can be integrated gradually into existing analytics infrastructures rather than requiring full architectural replacement. This makes the approach more feasible for institutions seeking incremental digital transformation.

Despite these strengths, several limitations should be acknowledged. First, the modular architecture introduces additional design decisions, and effective module selection still requires domain knowledge. Second, although the framework is more efficient than the baseline, it may not outperform highly specialized architectures designed for a single narrow task [32]. Third, while the present evaluation covers heterogeneous and dynamic conditions, broader validation on additional real-world domains would strengthen claims of generalizability. These limitations indicate that the framework should be viewed as a flexible design strategy rather than a fixed universal solution. Future work can build on this study in several directions [33]. One promising extension is the integration of explainable AI mechanisms to improve transparency and user trust, especially in education and high-stakes industrial decision contexts. Another is the development of automated configuration strategies that recommend modules, hyperparameters, and update schedules based on observed data characteristics. Further studies may also compare the present framework with transformer-based architectures, continual learning models, and edge-deployable systems under stricter resource constraints [34].

V. Conclusion

This study proposed an efficient computational framework for scalable learning in complex data environments using deep neural networks. The framework integrates adaptive preprocessing, modular neural network design, and resource-aware training strategies into a unified pipeline intended for heterogeneous, evolving, and computationally constrained settings. The empirical results show that the proposed approach maintains competitive predictive performance while improving convergence stability, training efficiency, multimodal robustness, and adaptability under distribution shift. The findings indicate that scalability should not be interpreted only as the ability to process larger datasets. Instead, it should be understood as the ability of a learning system to remain reliable, efficient, and maintainable as data volume, modality diversity, and operational complexity increase. In this sense, the contribution of this work lies not in introducing a single specialized model but in demonstrating that coordinated pipeline design can substantially improve the practical deployment of deep learning systems across domains such as learning analytics and Industry 4.0/5.0 applications. From an applied perspective, the framework offers a realistic pathway for organizations that need to update models continuously, integrate new data sources gradually, and operate under non-ideal computational conditions. At the same time, the study acknowledges that no single framework can fully address every domain-specific challenge. Future research should, therefore, extend this work by incorporating explainability mechanisms, automated configuration support, broader cross-domain validation, and comparisons with newer continual-learning and transformer-based architectures. These directions will help further strengthen the operational and scientific value of scalable deep learning in real-world environments.

References

[1] Y. Zhang, J. Auriol, and H. Yu, "Neural-Operator Control for Traffic Flow Models with Stochastic Demand," IFAC-PapersOnLine, vol. 59, no. 8, pp. 1–6, Jun. 2025, doi: 10.1016/j.ifacol.2025.08.057.

[2] O. Sobeyko and L. Mönch, "Heuristic approaches for scheduling jobs in large-scale flexible job shops," Comput. Oper. Res., vol. 68, pp. 97–109, Apr. 2016, doi: 10.1016/j.cor.2015.11.004.

[3] Y. Wang et al., "Spider silk inspired polymer electrolyte with well bonded interface and fast kinetics for solid-state lithium-ion batteries," Materials Today, vol. 76, pp. 1–8, Jul. 2024, doi: 10.1016/j.mattod.2024.05.001.

[4] L. Wang et al., "Dynamic job-shop scheduling in smart manufacturing using deep reinforcement learning," Computer Networks, vol. 190, May 2021, doi: 10.1016/j.comnet.2021.107969.

[5] Y. J. Wang, J. Li, and G. G. Wang, "Fuzzy correlation entropy-based NSGA-II for energy-efficient hybrid flow-shop scheduling problem," Knowl. Based. Syst., vol. 277, Oct. 2023, doi: 10.1016/j.knosys.2023.110808.

[6] M. Sakawa and R. Kubota, "Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy duedate through genetic algorithms," Eur. J. Oper. Res., vol. 120, no. 2, pp. 393–407, Jan. 2000, doi: 10.1016/S0377-2217(99)00094-6.

[7] D. Rooyani and F. M. Defersha, "An efficient two-stage genetic algorithm for flexible job-shop scheduling," IFAC-PapersOnLine, vol. 52, no. 13, pp. 2519–2524, Sep. 2019, doi: 10.1016/j.ifacol.2019.11.585.

[8] D. Müller, M. G. Müller, D. Kress, and E. Pesch, "An algorithm selection approach for the flexible job shop scheduling problem: Choosing constraint programming solvers through machine learning," Eur. J. Oper. Res., vol. 302, no. 3, pp. 874–891, Nov. 2022, doi: 10.1016/j.ejor.2022.01.034.

[9] C. Özgüven, L. Özbakir, and Y. Yavuz, "Mathematical models for job-shop scheduling problems with routing and process plan flexibility," Appl. Math. Model., vol. 34, no. 6, pp. 1539–1548, Jun. 2010, doi: 10.1016/j.apm.2009.09.002.

[10] I. Essafi, Y. Mati, and S. Dauzère-Pérès, "A genetic local search algorithm for minimizing total weighted tardiness in the job-shop scheduling problem," Comput. Oper. Res., vol. 35, no. 8, pp. 2599–2616, Aug. 2008, doi: 10.1016/j.cor.2006.12.019.

[11] Y. Lei, Q. Deng, M. Liao, and S. Gao, "Deep reinforcement learning for dynamic distributed job shop scheduling problem with transfers," Expert Syst. Appl., vol. 251, Oct. 2024, doi: 10.1016/j.eswa.2024.123970.

[12] M. M. Mafarja and S. Mirjalili, "Hybrid Whale Optimization Algorithm with simulated annealing for feature selection," Neurocomputing, vol. 260, pp. 302–312, Oct. 2017, doi: 10.1016/j.neucom.2017.04.053.

[13] R. Chen, B. Yang, S. Li, and S. Wang, "A self-learning genetic algorithm based on reinforcement learning for flexible job-shop scheduling problem," Comput. Ind. Eng., vol. 149, Nov. 2020, doi: 10.1016/j.cie.2020.106778.

[14] B. Chen and T. I. Matis, "A flexible dispatching rule for minimizing tardiness in job shop scheduling," Int. J. Prod. Econ., vol. 141, no. 1, pp. 360–365, Jan. 2013, doi: 10.1016/j.ijpe.2012.08.019.

[15] V. Kaplanoʇlu, "An object-oriented approach for multi-objective flexible job-shop scheduling problem," Expert Syst. Appl., vol. 45, pp. 71–84, Mar. 2016, doi: 10.1016/j.eswa.2015.09.050.

[16] O. A. Ghrayeb, "A bi-criteria optimization: Minimizing the integral value and spread of the fuzzy makespan of job shop scheduling problems," Applied Soft Computing Journal, vol. 2, no. 3, pp. 197–210, 2003, doi: 10.1016/S1568-4946(02)00069-8.

[17] S. Jia and Z. H. Hu, "Path-relinking Tabu search for the multi-objective flexible job shop scheduling problem," Comput. Oper. Res., vol. 47, pp. 11–26, Jul. 2014, doi: 10.1016/j.cor.2014.01.010.

[18] F. Neri and C. Cotta, "Memetic algorithms and memetic computing optimization: A literature review," Swarm Evol. Comput., vol. 2, pp. 1–14, 2012, doi: 10.1016/j.swevo.2011.11.003.

[19] J. M. Novas, "Production scheduling and lot streaming at flexible job-shops environments using constraint programming," Comput. Ind. Eng., vol. 136, pp. 252–264, Oct. 2019, doi: 10.1016/j.cie.2019.07.011.

[20] A. Allahverdi, "The third comprehensive survey on scheduling problems with setup times/costs," Eur. J. Oper. Res., vol. 246, no. 2, pp. 345–378, Oct. 2015, doi: 10.1016/j.ejor.2015.04.004.

[21] P. J. Lai and H. C. Wu, "Evaluate the fuzzy completion times in the fuzzy flow shop scheduling problems using the virus-evolutionary genetic algorithms," Applied Soft Computing Journal, vol. 11, no. 8, pp. 4540–4550, Dec. 2011, doi: 10.1016/j.asoc.2011.08.012.

[22] R. M. Devadas, V. Hiremani, Preethi, S. T, S. R, and P. Gujjar, "Hypercomplex neural networks: Exploring quaternion, octonion, and beyond in deep learning," MethodsX, vol. 15, Dec. 2025, doi: 10.1016/j.mex.2025.103644.

[23] R. Li, W. Gong, and C. Lu, "A reinforcement learning based RMOEA/D for bi-objective fuzzy flexible job shop scheduling," Expert Syst. Appl., vol. 203, Oct. 2022, doi: 10.1016/j.eswa.2022.117380.

[24] W. Yang, Z. Wang, and G.-G. Wang, "Optimizing fuzzy job shop scheduling using graph neural networks and deep reinforcement learning," Eng. Appl. Artif. Intell., vol. 174, p. 114506, Jun. 2026, doi: 10.1016/j.engappai.2026.114506.

[25] S. Afsar, C. R. Vela, J. J. Palacios, and I. González-Rodríguez, "Mathematical models and benchmarking for the fuzzy job shop scheduling problem," Comput. Ind. Eng., vol. 183, Sep. 2023, doi: 10.1016/j.cie.2023.109454.

[26] M. Ulicny, V. A. Krylov, and R. Dahyot, "Harmonic convolutional networks based on discrete cosine transform," Pattern Recognit., vol. 129, Sep. 2022, doi: 10.1016/j.patcog.2022.108707.

[27] M. Rojc and I. Mlakar, "An LSTM-based model for the compression of acoustic inventories for corpus-based text-to-speech synthesis systems," Computers and Electrical Engineering, vol. 100, May 2022, doi: 10.1016/j.compeleceng.2022.107942.

[28] J. Lin, L. Ma, and Y. Yao, "A Fourier domain acceleration framework for convolutional neural networks," Neurocomputing, vol. 364, pp. 254–268, Oct. 2019, doi: 10.1016/j.neucom.2019.06.080.

[29] J. A. Stuchi, N. G. Canto, R. R. de F. Attux, and L. Boccato, "A frequency-domain approach with learnable filters for image classification[Formula presented]," Appl. Soft Comput., vol. 155, Apr. 2024, doi: 10.1016/j.asoc.2024.111443.

[30] S. Liu et al., "Improve cognition of depressive patients through the regulation of basal ganglia connectivity: Combined medication using Shuganjieyu capsule," J. Psychiatr. Res., vol. 123, pp. 39–47, Apr. 2020, doi: 10.1016/j.jpsychires.2020.01.013.

[31] Y. Zhang, K. I. Kou, Y. Zhang, and L. Liu, "Fast fixed-/preassigned-time synchronization of Clifford-valued neural networks for medical image encryption," Neurocomputing, vol. 651, Oct. 2025, doi: 10.1016/j.neucom.2025.130984.

[32] I. Y. Alshareef et al., "Lightweight and high-precision spectral convolutional neural network model for custom 94-class ASCII character recognition," Neural Comput. Appl., vol. 37, no. 21, pp. 16883–16904, Jul. 2025, doi: 10.1007/s00521-025-11376-2.

[33] J. Gu et al., "Recent advances in convolutional neural networks," Pattern Recognit., vol. 77, pp. 354–377, May 2018, doi: 10.1016/j.patcog.2017.10.013.

[34] S. O. Ayat, M. Khalil-Hani, A. A. H. Ab Rahman, and H. Abdellatef, "Spectral-based convolutional neural network without multiple spatial-frequency domain switchings," Neurocomputing, vol. 364, pp. 152–167, Oct. 2019, doi: 10.1016/j.neucom.2019.06.094.