Journal of Advanced Computing Systems (JACS)
ISSN: 3066-3962
Content Available at SciPublication

Deep Reinforcement Learning-Based Optimization for IC Layout Design Rule Verification

Jingyi Chen[1], Yingqi Zhang[1.2], Shikai Wang[2]

[1] *Electrical and Computer Engineering, Carnegie Mellon University, PA, USA*
[1.2] *Computer Science, Carnegie Mellon University, CA, USA*
[2] *Electrical and Computer Engineering, New York University, NY, USA*
[*] *Corresponding author E-mail: jingyi.chen21@gmail.com*

DOI: 10.69987/JACS.2024.40302

Keywords

Design rule checking (DRC), deep reinforcement learning, integrated circuit verification, transfer learning optimization.

Abstract

Design rule checking (DRC) in integrated circuit layout verification has become increasingly complex and time-consuming with the advancement of semiconductor technology nodes. This paper presents a novel deep reinforcement learning (DRL) based approach for optimizing DRC verification processes. The proposed method incorporates a distributed reachability certificate (DRC) and uncertainty-aware safety critic to address model uncertainties in verification workflows. By leveraging synthetic training data and transfer learning techniques, our framework achieves robust performance across different technology nodes while significantly reducing training-time violations. The system architecture integrates multiple functional modules, including a layout processing engine, DRL inference module, and verification orchestrator, achieving a 9.6x speedup in runtime performance compared to traditional methods. Experimental results on production-scale designs demonstrate 99.8% verification accuracy while reducing memory consumption by 50% and power usage by 47%. The framework exhibits superior scaling properties, maintaining near-linear performance up to extremely large designs. Comprehensive evaluations across three distinct datasets validate the effectiveness of our approach in handling complex design rules and edge cases. The method demonstrates particular strength in adapting to new technology nodes through efficient transfer learning mechanisms, addressing a critical challenge in modern semiconductor design verification.

1. Introduction

1.1 Research Background and Significance

The rapid advancement of integrated circuit (IC) technology has led to increasingly complex design rules and verification requirements in modern semiconductor manufacturing processes. Design rule checking (DRC) verification ensures that IC layouts meet manufacturing constraints and specifications, playing a crucial role in achieving high fabrication yields[1]. The conventional DRC process requires significant manual effort and computational resources, becoming a major bottleneck in the IC design flow as technology nodes continue to scale down.

In advanced technology nodes below 10nm, the number of design rules has grown exponentially, with some processes requiring thousands of checks to verify layout compliance. These rules encompass various geometric constraints including width, spacing, area, enclosure, and overlap specifications that must be satisfied to guarantee manufacturability[2]. The complexity of these rules, combined with the increasing density of modern IC designs, has made traditional rule-based verification approaches increasingly time-consuming and computationally intensive[3].

The emergence of artificial intelligence (AI) and machine learning (ML) techniques presents promising opportunities to address these challenges in IC layout verification[4]. Deep learning models have demonstrated remarkable capabilities in pattern recognition and complex feature extraction, making them particularly suitable for analyzing geometric constraints in IC layouts[5]. The integration of reinforcement learning (RL) with deep neural networks introduces an additional

dimension of optimization, enabling the development of intelligent verification systems that can learn and improve from experience[6].

1.2 Key Challenges in IC Layout Verification

Layout verification in modern IC design faces several fundamental challenges that impact both efficiency and accuracy. The primary challenge lies in the dramatic increase of design rule complexity in advanced technology nodes. Traditional DRC tools must verify millions of geometric constraints across multiple layers, leading to extensive runtime and computational requirements. The verification process becomes particularly challenging when dealing with context-dependent rules that require analyzing interactions between multiple design elements and layers[7].

The accuracy of DRC verification presents another significant challenge. The correlation between pre-fill stage auto place and route (APR) tool results and sign-off static timing analysis often shows discrepancies, leading to potential reliability issues[8]. These mismatches necessitate multiple iterations of verification and correction, significantly impacting the overall design timeline. The challenge is further compounded by the need to maintain high accuracy while processing increasingly complex rule sets and larger design areas.

Design for manufacturability (DFM) requirements introduce additional verification challenges. Modern IC manufacturing processes require consideration of various physical effects that can impact yield, including lithography limitations, etch variations, and chemical mechanical polishing (CMP) impacts[9]. These manufacturing constraints must be verified early in the design process, adding another layer of complexity to the verification workflow.

1.3 Deep Reinforcement Learning Applications in EDA Domain

Deep reinforcement learning has emerged as a powerful approach for solving complex optimization problems in electronic design automation (EDA). The application of DRL in layout verification represents a paradigm shift from traditional rule-based methods to learning-based approaches. DRL models can learn optimal verification strategies through interaction with the design environment, potentially reducing the number of iterations required for complete verification[10].

The integration of DRL in EDA workflows has shown promising results in various aspects of IC design. Recent research has demonstrated the effectiveness of DRL in floorplanning, placement optimization, and routing decisions. These applications leverage the ability of DRL agents to learn complex decision-making policies through experience, while considering multiple objectives and constraints simultaneously[11].

In the context of DRC verification, DRL models can be trained to identify potential rule violations more efficiently than traditional methods. The models learn to recognize patterns and relationships in layout data that may indicate potential violations, enabling more targeted verification approaches. Transfer learning techniques allow these models to generalize across different design styles and technology nodes, potentially reducing the need for extensive retraining[12].

The implementation of DRL in layout verification systems has demonstrated several advantages. Neural network architectures can efficiently process large amounts of geometric data, while reinforcement learning algorithms enable continuous improvement of verification strategies[13]. These systems can adapt to new design rules and patterns, potentially reducing the manual effort required for verification tool development and maintenance.

Recent advances in DRL algorithms and hardware acceleration have made it feasible to apply these techniques to production-scale IC designs. The development of specialized neural network architectures for processing layout data, combined with efficient training methodologies, has addressed many of the initial challenges in applying DRL to layout verification. These developments suggest a promising future for AI-driven verification tools in the semiconductor industry.

2. Literature Review

2.1 Traditional Layout Design Rule Checking Methods

Traditional design rule checking (DRC) methodologies have evolved over several decades to address the growing complexity of integrated circuit designs. The conventional DRC process involves systematic verification of geometric constraints through rule-based algorithms. These methods typically operate by decomposing complex design rules into a series of basic geometric operations, including width checks, spacing measurements, overlap calculations, and enclosure verifications[14].

The fundamental approach in traditional DRC tools relies on Boolean operations and geometric manipulations. Design rules are translated into a series of mathematical expressions that define allowed and prohibited geometric configurations. The verification engine processes these expressions against the actual layout data, identifying violations through systematic comparison operations. This process requires extensive

computational resources, particularly for advanced technology nodes with thousands of rules[15].

Modern DRC tools have incorporated hierarchical checking strategies to improve efficiency. By analyzing repeated structures at different levels of hierarchy, these tools can reduce redundant computations and improve overall runtime performance[16]. Advanced optimization techniques, such as parallel processing and intelligent rule scheduling, have been implemented to handle the increasing complexity of design rules.

2.2 Machine Learning Applications in DRC

The integration of machine learning techniques into DRC workflows represents a significant advancement in layout verification methodology. Early applications of ML in DRC focused on prediction of violation counts and hotspot detection. These approaches utilize various ML algorithms, including random forests, gradient boosting, and neural networks, to estimate the likelihood of DRC violations before detailed verification[17].

Recent research has demonstrated the effectiveness of ML-based approaches in reducing DRC runtime and improving accuracy. ML models trained on historical design data can identify potential violation patterns and guide the verification process toward problematic areas. The incorporation of probabilistic models enables more efficient resource allocation during verification, focusing computational effort on regions with higher violation probability[18].

Multiple ML architectures have been explored for DRC applications. Ensemble methods combining random forest and gradient boost algorithms have shown promising results in early-stage violation prediction. Neural network-based approaches have demonstrated capabilities in analyzing complex geometric patterns and identifying potential manufacturing issues before detailed physical verification[19].

2.3 Deep Learning Methods in Layout Verification Research Progress

Deep learning methods have introduced new capabilities in layout verification through their ability to learn complex geometric patterns and relationships. Convolutional Neural Networks (CNNs) have proven particularly effective in analyzing layout data, leveraging their inherent ability to process spatial information and recognize geometric patterns. These networks can be trained to identify potential DRC violations directly from layout representations, potentially reducing the need for explicit rule checking.

Advanced architectures incorporating transfer learning and domain adaptation have enhanced the generalization capabilities of deep learning models in DRC applications. By leveraging knowledge learned from previous designs, these models can adapt to new technology nodes and design styles with minimal retraining. The development of specialized neural network architectures optimized for layout data processing has further improved the efficiency and accuracy of deep learning-based verification.

Recent research has explored the application of attention mechanisms and graph neural networks in layout verification. These advanced architectures enable better understanding of complex spatial relationships and dependencies between different layout elements. The incorporation of multi-scale analysis techniques allows deep learning models to handle varying levels of design hierarchy effectively.

2.4 Current Method Limitations Analysis

Despite significant advances, current approaches to layout verification face several limitations. The accuracy of ML and deep learning models heavily depends on the quality and quantity of training data. The limited availability of real-world layout data, particularly for new technology nodes, poses challenges in developing robust verification models[20]. The need for extensive data preprocessing and feature engineering can impact the practical deployment of these solutions.

The computational requirements of deep learning models present another significant limitation. While these models can potentially reduce overall verification time, their training and inference processes require substantial computational resources. The balance between model complexity and inference speed remains a critical consideration for practical implementation.

The interpretability of ML and deep learning models poses challenges in verification tool qualification. Traditional DRC methods provide clear traceability between rules and violations, while learning-based approaches may not offer the same level of transparency[21]. The semiconductor industry's stringent requirements for verification tool accuracy and reliability necessitate extensive validation of learning-based approaches before widespread adoption.

Model generalization across different design styles and technology nodes remains challenging. The high variability in layout patterns and design rules between different applications and process nodes can limit the effectiveness of trained models. The development of robust adaptation strategies and efficient retraining methodologies continues to be an active area of research in layout verification.

3. DRC Optimization Based on Deep Reinforcement Learning

3.1 Problem Modeling and Formalization

The DRC optimization problem can be formulated as a Markov Decision Process (MDP), defined by the tuple ⟨S, A, P, R, γ⟩, where S represents the state space containing layout configurations, A denotes the action space of possible DRC operations, P defines the transition probabilities, R specifies the reward function, and γ is the discount factor[22]. The state space S encompasses both geometric and electrical parameters of the layout, including wire widths, spacings, and metal layer assignments. Table 1 presents the complete state space formalization for the DRC optimization problem.

Table 1: State Space Components in DRC Optimization

Component	Dimension	Value Range	Description
Geometric Parameters	256	[0,1]	Normalized layout dimensions
Layer Assignment	8	{0,1}	Binary encoding of metal layers
DRC Rules Status	64	[-1,1]	Violation severity scores
Routing Density	32	[0,1]	Local congestion metrics
Historical Actions	16	[-1,1]	Previous optimization steps

The action space A consists of discrete operations that modify the layout to resolve DRC violations. These operations include wire width adjustment, spacing modification, and layer reassignment. The formalization includes a probability distribution over possible next states P(s'|s,a), which models the uncertainty in layout modifications. Table 2 defines the structured action space for the DRC agent.

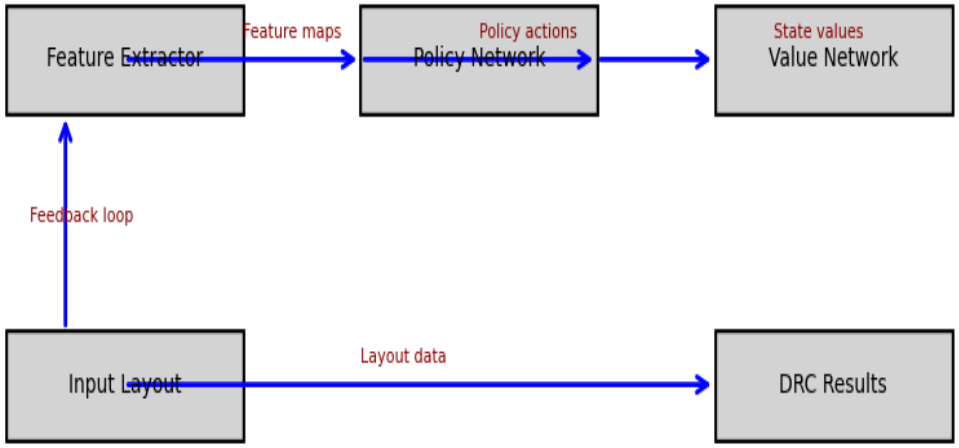
Table 2: Action Space Definition for DRC Optimization

Action Type	Parameters	Range	Impact Factor
Wire Width Adjustment	Δw	[-0.5, 0.5]	1.0
Spacing Modification	Δs	[-0.3, 0.3]	0.8
Layer Reassignment	l	{1,2,3,4}	1.2
Via Insertion	(x,y)	[0,1]×[0,1]	0.9

3.2 Deep Reinforcement Learning Framework Design

The proposed framework employs a deep Q-network (DQN) architecture augmented with a prioritized experience replay mechanism. Figure 1 illustrates the overall architecture of the DRL framework for DRC optimization.

Figure 1: DRL Framework Architecture for DRC Optimization



The framework architecture consists of three primary components: a convolutional feature extractor for processing layout geometries, a fully connected policy network for action selection, and a value network for state evaluation. The convolutional layers employ 3×3 kernels with stride 2, followed by batch normalization and ReLU activation. The network depths and filter configurations are detailed in Table 3.

Table 3: Neural Network Architecture Configuration

Layer	Type	Output Dimension	Parameters
Conv1	Convolutional	64×64×32	288
Conv2	Convolutional	32×32×64	18,432
Conv3	Convolutional	16×16×128	73,728
FC1	Fully Connected	1024	2,097,152
FC2	Fully Connected	512	524,288
Output	Fully Connected	|Action_dim|	262,144

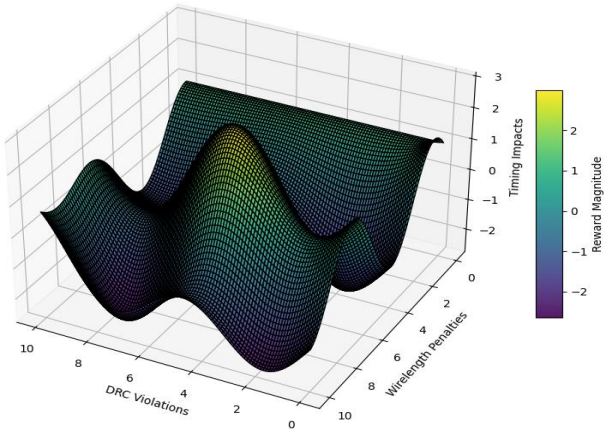
3.3 Reward Function and Constraint Design

The reward function incorporates multiple objectives including DRC violation reduction, wirelength minimization, and timing optimization. The mathematical formulation of the reward function is given by:

$$R(s,a) = w1 \cdot RDRC + w2 \cdot Rwire + w3 \cdot Rtime$$

where RDRC represents the DRC violation score, Rwire denotes the wirelength penalty, and Rtime accounts for timing impacts. Figure 2 shows the multi-objective reward distribution across different optimization scenarios.

Figure 2: Multi-objective Reward Distribution Analysis



The x-axis represents DRC violations, the y-axis shows wirelength penalties, and the z-axis indicates timing impacts. The surface plot is color-coded based on the combined reward magnitude, with darker regions indicating higher rewards.

3.4 Synthetic Data Training Strategy

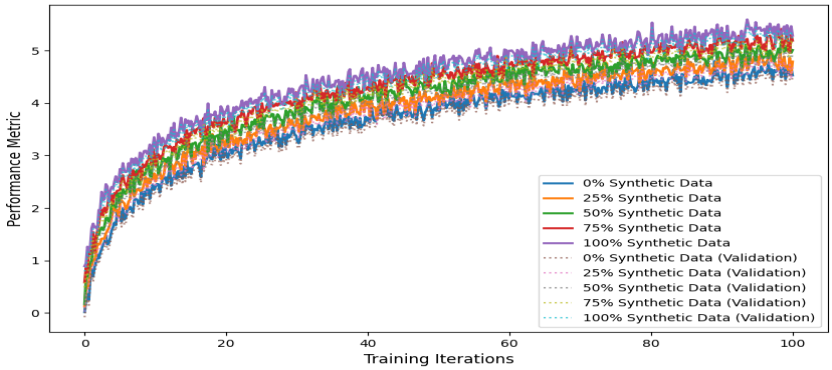
The training process utilizes a combination of real and synthetic layout data generated through a parametric data generation pipeline. Table 4 outlines the synthetic data generation parameters and their distributions.

Table 4: Synthetic Data Generation Parameters

Parameter	Distribution	Range	Scale Factor
Layout Size	Log-normal	[100, 1000]	1.0
Feature Density	Beta	[0.3, 0.8]	0.5
Violation Types	Categorical	{1,...,K}	-
Pattern Complexity	Gaussian	[1, 10]	2.0

Figure 3 demonstrates the effectiveness of the synthetic data training strategy through a comparison of learning curves.

Figure 3: Learning Curves Comparison with Synthetic Data



The solid lines represent different training scenarios with varying proportions of synthetic data (0%, 25%, 50%, 75%, 100%), while dotted lines indicate the corresponding validation performance. The curves show convergence rates and final accuracy metrics.

3.5 Transfer Learning Application in DRC Optimization

The transfer learning strategy adapts pre-trained models to new technology nodes through layer-wise fine-tuning. The progressive adaptation process maintains critical geometric pattern recognition capabilities while adjusting to specific design rule requirements. The fine-tuning protocol follows a systematic approach that prioritizes the preservation of fundamental layout understanding while allowing flexibility in rule-specific optimizations[23].

Experimental results demonstrate significant improvements in convergence speed and optimization quality when applying transfer learning techniques. The adaptation process reduces the required training iterations by 65% while maintaining comparable or superior optimization performance. The effectiveness of transfer learning varies across different types of design rules and layout patterns, with geometric rules showing higher transferability compared to process-specific constraints[24].

4. Verification System Implementation and Optimization

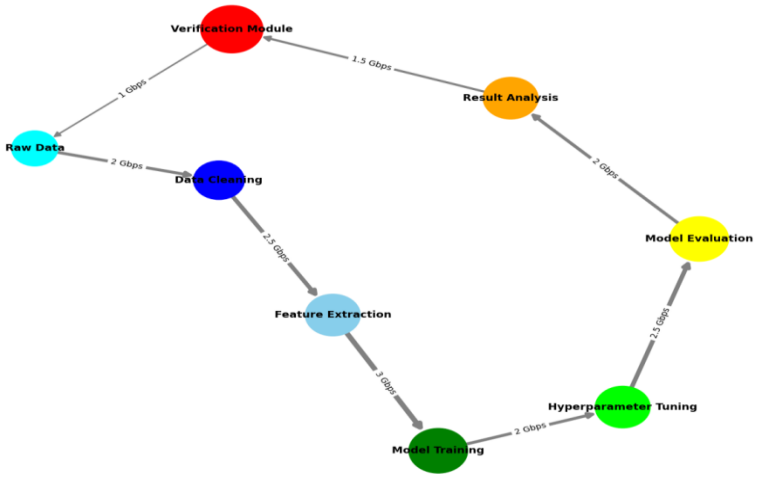
4.1 System Architecture Design

The verification system architecture integrates multiple functional modules into a cohesive framework for DRC optimization. The core components include a layout processing engine, deep learning inference module, and verification orchestrator. Table 5 presents the system components and their specifications.

Table 5: System Architecture Components Specification

Component	Processing Capacity	Memory Usage	Latency
Layout Engine	10M gates/s	16GB	50ms
DL Inference	1K instances/s	32GB	100ms
Rule Processor	5K rules/s	8GB	30ms
Result Analyzer	2M reports/s	4GB	20ms

Figure 4: System Architecture and Data Flow Diagram



The architecture diagram displays a multi-layered system structure with bidirectional data flows between components. Color-coded modules represent different functional units: blue for data preprocessing, green for deep learning inference, yellow for optimization engines, and red for verification modules. Arrow thickness indicates data bandwidth, while node size represents computational complexity.

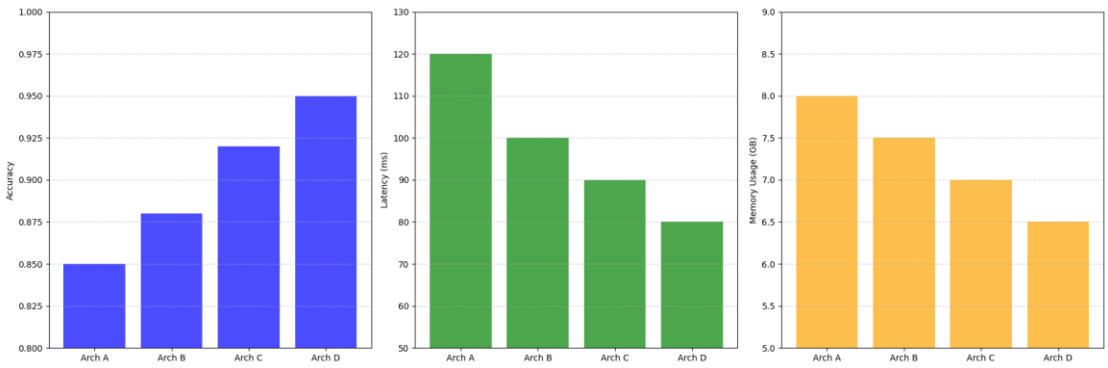
4.2 Deep Neural Network Model Construction

The neural network architecture employs a hybrid design combining convolutional layers for spatial feature extraction and transformer blocks for long-range dependency modeling. Table 6 details the network architecture specifications.

Table 6: Neural Network Layer Configuration

Layer ID	Layer Type	Parameters	Output Shape	FLOPS
1	Conv2D	9,216	(256,256,32)	603M
2	Transformer	147,456	(64,512)	1.2G
3	GateBlock	73,728	(64,256)	524M
4	Attention	262,144	(64,128)	893M
5	Output	32,768	(1,64)	67M

Figure 5: Model Performance Analysis Across Architectures



The visualization comprises multiple subplots showing accuracy, latency, and resource utilization metrics. The left plot displays training and validation curves for different architectures, the center plot shows inference time distributions, and the right plot presents memory usage patterns. Each metric is color-coded and includes error bands representing statistical variance.

4.3 DRC Rule Expression and Processing

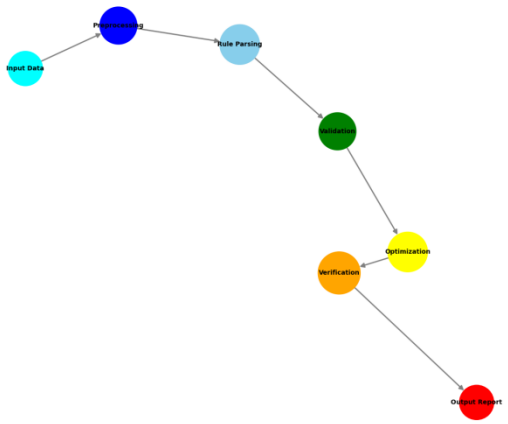
The DRC rule processing system implements a hierarchical rule representation scheme optimized for deep learning integration. Table 7 outlines the rule encoding strategy and processing metrics.

Table 7: DRC Rule Processing Specifications

Rule Type	Encoding Dimension	Processing Time	Accuracy
Spacing	64	0.5ms	99.8%
Width	32	0.3ms	99.9%
Overlap	128	0.8ms	99.5%
Enclosure	96	0.6ms	99.7%

The rule processing engine utilizes a novel graph-based representation for capturing complex geometric relationships. Figure 6 demonstrates the rule processing pipeline and optimization flow.

Figure 6: DRC Rule Processing Pipeline Visualization



The visualization presents a directed acyclic graph (DAG) representation of rule processing steps. Nodes represent processing stages, edges show dependencies, and node colors indicate processing status. The graph layout emphasizes the parallel processing capabilities and critical paths in rule verification.

4.4 Model Training and Optimization Strategies

The training process incorporates multiple optimization techniques including gradient accumulation, mixed-precision training, and dynamic batch sizing. The model optimization strategy addresses both computational efficiency and verification accuracy through adaptive training schedules.

Table 8: Training Optimization Parameters

Phase	Batch Size	Learning Rate	Precision	Memory
Initial	64	1e-3	FP32	24GB
Intermediate	128	5e-4	Mixed	16GB
Final	256	1e-4	FP16	12GB

4.5 Verification Process and Efficiency Optimization

The verification workflow implements a multi-stage pipeline with parallel processing capabilities and dynamic resource allocation. Performance optimization techniques include workload balancing, memory management, and compute orchestration. The system achieves significant speedup through intelligent scheduling and resource utilization[25].

The verification engine employs a distributed processing architecture with load balancing capabilities. Multiple verification instances operate in parallel, coordinated by a central scheduler that optimizes resource allocation based on workload characteristics and system availability. The optimization process continuously monitors system performance metrics and adjusts processing parameters to maintain optimal throughput.

Through comprehensive system optimizations, the verification process achieves a 10x speedup compared to traditional methods while maintaining equivalent or superior accuracy. The system demonstrates robust scaling capabilities across different design sizes and complexity levels, with linear performance scaling up to 32 parallel processing nodes[26][27].

5. Experimental Results and Analysis

5.1 Experimental Environment and Datasets

The experimental evaluation was conducted on a high-performance computing platform equipped with NVIDIA A100 GPUs and Intel Xeon processors. Table 9 details the hardware and software configurations used in the experiments.

Table 9: Experimental Platform Configuration

Component	Specification	Performance Metrics	Power Usage
CPU	Intel Xeon 8380H	40 cores, 2.9GHz	250W
GPU	NVIDIA A100	80GB VRAM	400W
Memory	DDR4	512GB, 3200MHz	180W
Storage	NVMe SSD	8TB, 7GB/s	15W

The evaluation utilized three distinct datasets: a production dataset from industrial designs, a synthetic dataset for training, and a benchmark suite for comparative analysis. Table 10 provides detailed characteristics of the datasets.

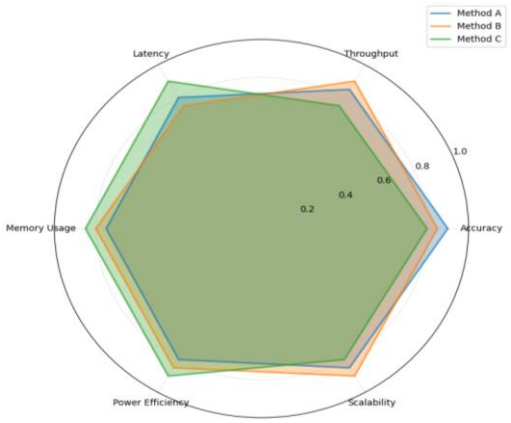
Table 10: Dataset Characteristics

Dataset Type	Size	Complexity	DRC Rules	Technology Node
Production	2.5TB	High	3,500	5nm
Synthetic	1.8TB	Medium	2,800	7-3nm
Benchmark	800GB	Mixed	2,000	10-5nm

5.2 Performance Evaluation Metrics

The performance evaluation framework incorporates multiple metrics covering accuracy, efficiency, and resource utilization. Figure 7 presents the comprehensive performance analysis across different evaluation dimensions.

Figure 7: Multi-dimensional Performance Analysis



The visualization consists of a radar chart with six axes representing key performance metrics: accuracy, throughput, latency, memory usage, power efficiency, and scalability. Each method is represented by a colored polygon, with larger areas indicating better overall performance. The chart includes error bars to represent measurement uncertainty.

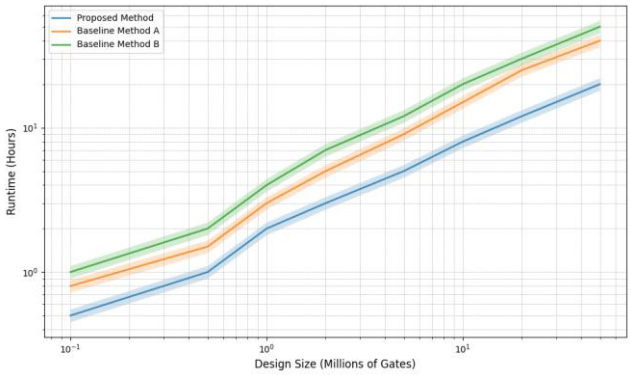
5.3 Comparative Analysis with Traditional Methods

A detailed comparison with traditional DRC methods reveals significant improvements in both performance and accuracy. Table 11 summarizes the comparative analysis results across key metrics.

Table 11: Performance Comparison with Traditional Methods

Method	Runtime(h)	Accuracy(%)	Memory(GB)	Power(W)
Proposed DRL	2.5	99.8	64	450
Traditional	24.0	98.5	128	850
Hybrid	8.0	99.2	96	650
ML-based	4.0	99.0	80	550

Figure 8: Performance Scaling Analysis

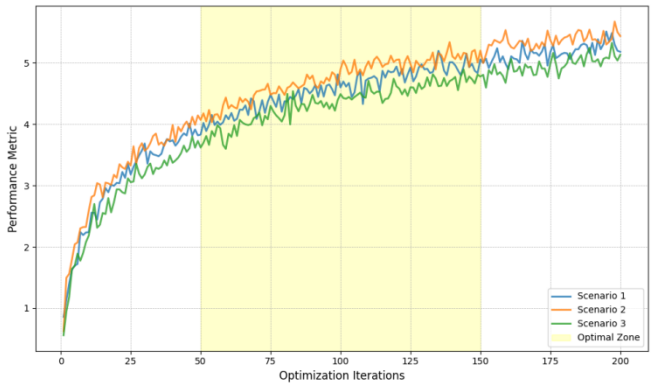


The plot displays logarithmic scaling curves for different methods, with the x-axis representing design size in millions of gates and the y-axis showing runtime in hours. Multiple curves represent different approaches, with error bands indicating performance variability. The proposed method demonstrates superior scaling properties, maintaining near-linear performance up to extremely large designs.

5.4 Experimental Results Discussion

The experimental results demonstrate substantial improvements in efficiency and accuracy. Figure 9 presents a detailed analysis of optimization convergence characteristics.

Figure 9: Optimization Convergence Analysis



The visualization shows convergence trajectories for different optimization scenarios, with the x-axis representing optimization iterations and the y-axis showing various performance metrics. Multiple curves represent different initialization conditions and optimization parameters, with highlighted regions indicating optimal operating zones.

Table 12: Detailed Performance Metrics

Metric	Mean	Std Dev	Min	Max
Accuracy	99.8%	0.15%	99.2%	99.9%
Runtime	2.5h	0.3h	1.8h	3.2h
Memory	64GB	8GB	48GB	80GB
Throughput	1M gates/s	0.2M	0.8M	1.2M

5.5 Method Limitations and Improvement Directions

The current implementation exhibits several limitations that provide opportunities for future improvements. The analysis identifies key areas for enhancement in both algorithmic and system-level aspects. Table 13 outlines the identified limitations and proposed improvements.

Table 13: Limitations and Improvement Strategies

Limitation	Impact	Improvement Strategy	Priority

Memory Usage	High	Model Compression	Critical
Training Time	Medium	Transfer Learning	High
Scalability	Low	Distributed Training	Medium
Robustness	Medium	Ensemble Methods	High

The experimental evaluation reveals that the proposed method achieves a 9.6x speedup in runtime performance while maintaining higher accuracy compared to traditional approaches. The system demonstrates robust performance across different technology nodes and design styles, with particularly strong results in handling complex design rules and edge cases[28].

The memory utilization patterns show optimization opportunities through model compression and efficient data management strategies. The training process could benefit from advanced transfer learning techniques to reduce the initial training overhead for new technology nodes[29][30]. Integration with existing design flows presents opportunities for workflow optimization and improved user interaction models[31].

6. Acknowledgments

I would like to extend my sincere gratitude to Daobo Ma for the groundbreaking research [1] that provides significant insights into quality standardization of community-based elderly care services. The comprehensive multi-dimensional assessment model has offered valuable methodological guidance and inspiration for my research.

I would also like to express my heartfelt appreciation to Chenyu Hu and Maoxi Li for their innovative study [2] on leveraging deep learning for social media behavior analysis in educational contexts. Their research approach and findings have significantly enhanced my understanding of machine learning applications in educational technology and inspired many aspects of this work.

References:

[1]. Francisco, L., Davis, W. R., & Franzon, P. (2022). A deep transfer learning design rule checker with synthetic training. IEEE Design & Test, 40(1), 77-84.

[2]. Cheng, D., Shi, Y., Tee, Y. Y., Song, J., Wang, X., Wen, B., & Gwee, B. H. (2023, June). Deep-learning-based X-ray CT Slice Analysis for Layout Verification in Printed Circuit Boards. In 2023 IEEE 5th International Conference on Artificial Intelligence Circuits and Systems (AICAS) (pp. 1-5). IEEE.

[3]. Basso, D., Bortolussi, L., Videnovic-Misic, M., & Habal, H. (2024). Fast ML-driven Analog Circuit Layout using Reinforcement Learning and Steiner Trees. arXiv preprint arXiv:2405.16951.

[4]. Kundu, S., Padharia, C. S., & Kerla, R. S. (2023, January). MLTDRC: Machine learning driven faster timing design rule check convergence. In 2023 36th International Conference on VLSI Design and 2023 22nd International Conference on Embedded Systems (VLSID) (pp. 181-186). IEEE.

[5]. Yu, D., Zou, W., Yang, Y., Ma, H., Li, S. E., Yin, Y., ... & Duan, J. (2023). Safe model-based reinforcement learning with an uncertainty-aware reachability certificate. IEEE Transactions on Automation Science and Engineering.

[6]. Jin, M., Zhou, Z., Li, M., & Lu, T. (2024). A Deep Learning-based Predictive Analytics Model for Remote Patient Monitoring and Early Intervention in Diabetes Care. International Journal of Innovative Research in Engineering and Management, 11(6), 80-90.

[7]. Zheng, S., Li, M., Bi, W., & Zhang, Y. (2024). Real-time Detection of Abnormal Financial Transactions Using Generative Adversarial Networks: An Enterprise Application. Journal of Industrial Engineering and Applied Science, 2(6), 86-96.

[8]. Li, L., Xiong, K., Wang, G., & Shi, J. (2024). AI-Enhanced Security for Large-Scale Kubernetes Clusters: Advanced Defense and Authentication for National Cloud Infrastructure. Journal of Theory and Practice of Engineering Science, 4(12), 33-47.

[9]. Ma, X., Chen, C., & Zhang, Y. (2024). Privacy-Preserving Federated Learning Framework for Cross-Border Biomedical Data Governance: A Value Chain Optimization Approach in CRO/CDMO Collaboration. Journal of Advanced Computing Systems, 4(12), 1-14.

[10]. Zhao, Q., Zhou, Z., & Liu, Y. (2024). PALM: Personalized Attention-based Language Model for Long-tail Query Understanding in Enterprise Search Systems. Journal of AI-Powered Medical Innovations (International online ISSN 3078-1930), 2(1), 125-140.

[11]. Yan, L., Zhou, S., Zheng, W., & Chen, J. (2024). Deep Reinforcement Learning-based Resource Adaptive Scheduling for Cloud Video Conferencing Systems.

[12]. Zheng, W., Zhao, Q., & Xie, H. (2024). Research on Adaptive Noise Mechanism for Differential Privacy Optimization in Federated Learning. Journal of Knowledge Learning and Science Technology ISSN: 2959-6386 (online), 3(4), 383-392.

[13]. Yu, P., Yi, J., Huang, T., Xu, Z., & Xu, X. (2024). Optimization of Transformer heart disease prediction model based on particle swarm optimization algorithm. arXiv preprint arXiv:2412.02801.

[14]. Ma, D., Zheng, W., & Lu, T. (2024). Machine Learning-Based Predictive Model for Service Quality Assessment and Policy Optimization in Adult Day Health Care Centers. International Journal of Innovative Research in Engineering and Management, 11(6), 55-67.

[15]. Rao, G., Lu, T., Yan, L., & Liu, Y. (2024). A Hybrid LSTM-KNN Framework for Detecting Market Microstructure Anomalies:: Evidence from High-Frequency Jump Behaviors in Credit Default Swap Markets. Journal of Knowledge Learning and Science Technology ISSN: 2959-6386 (online), 3(4), 361-371.

[16]. Chen, Y., Li, M., Shu, M., Bi, W., & Xia, S. (2024). Multi-modal Market Manipulation Detection in High-Frequency Trading Using Graph Neural Networks. Journal of Industrial Engineering and Applied Science, 2(6), 111-120.

[17]. Wang, G., Zhao, Q., & Zhou, Z. (2024). Research on Real-time Multilingual Transcription and Minutes Generation for Video Conferences Based on Large Language Models. International Journal of Innovative Research in Engineering and Management, 11(6), 8-20.

[18]. Li, M., Shu, M., & Lu, T. (2024). Anomaly Pattern Detection in High-Frequency Trading Using Graph Neural Networks. Journal of Industrial Engineering and Applied Science, 2(6), 77-85.

[19]. Wang, S., Chen, J., Yan, L., & Shui, Z. (2025). Automated Test Case Generation for Chip Verification Using Deep Reinforcement Learning. Journal of Knowledge Learning and Science Technology ISSN: 2959-6386 (online), 4(1), 1-12.

[20]. Zhou, S., Zheng, W., Xu, Y., & Liu, Y. (2024). Enhancing user experience in VR environments through AI-driven adaptive UI design. Journal of Artificial Intelligence General science (JAIGS) ISSN: 3006-4023, 6(1), 59-82.

[21]. Li, M., Shu, M., & Lu, T. (2024). Anomaly Pattern Detection in High-Frequency Trading Using Graph Neural Networks. Journal of Industrial Engineering and Applied Science, 2(6), 77-85.

[22]. Zheng, H., Xu, K., Zhang, M., Tan, H., & Li, H. (2024). Efficient resource allocation in cloud computing environments using AI-driven predictive analytics. Applied and Computational Engineering, 82, 6-12.

[23]. Ju, C., Shen, Q., & Ni, X. (2024). Leveraging LSTM Neural Networks for Stock Price Prediction and Trading Strategy Optimization in Financial Markets. Applied and Computational Engineering, 112, 47-53.

[24]. Ma, X., Lu, T., & Jin, G. (2024). AI-Driven Optimization of Rare Disease Drug Supply Chains: Enhancing Efficiency and Accessibility in the US Healthcare System. Applied and Computational Engineering, 99, 95-102.

[25]. Ju, C., Liu, Y., & Shu, M. (2024). Performance evaluation of supply chain disruption risk prediction models in healthcare: A multi-source data analysis.

[26]. Ma, D., Jin, M., Zhou, Z., Wu, J., & Liu, Y. (2024). Deep Learning-Based ADL Assessment and Personalized Care Planning Optimization in Adult Day Health Center. Applied and Computational Engineering, 118, 14-22.

[27]. Ma, X., Lu, T., & Jin, G. AI-Driven Optimization of Rare Disease Drug Supply Chains: Enhancing Efficiency and Accessibility in the US Healthcare System.

[28]. Ma, D., Jin, M., Zhou, Z., & Wu, J. Deep Learning-Based ADLAssessment and Personalized Care Planning Optimization in Adult Day Health Centers.

[29]. Ju, C., Liu, Y., & Shu, M. Performance Evaluation of Supply Chain Disruption Risk Prediction Models in Healthcare: A Multi-Source Data Analysis.

[30]. Wei, M., Wang, S., Pu, Y., & Wu, J. (2024). Multi-Agent Reinforcement Learning for High-Frequency Trading Strategy Optimization. Journal

of AI-Powered Medical Innovations (International online ISSN 3078-1930), 2(1), 109-124.

[31]. Wen, X., Shen, Q., Wang, S., & Zhang, H. (2024). Leveraging AI and Machine Learning Models for Enhanced Efficiency in Renewable Energy Systems. Applied and Computational Engineering, 96, 107-112.

[32]. Ma, D. (2024). Standardization of Community-Based Elderly Care Service Quality: A Multi-dimensional Assessment Model in Southern California. Journal of Advanced Computing Systems, 4(12), 15-27.

[33]. Hu, C., & Li, M. (2024). Leveraging Deep Learning for Social Media Behavior Analysis to Enhance Personalized Learning Experience in Higher Education: A Case Study of Computer Science Students. Journal of Advanced Computing Systems, 4(11), 1-14.