DNN-Driven Adaptive GA for Resource Allocation in Private 5G Cybernetic Avatar Networks

Arif Dataesatu, Zie Eya Ekolle, and Takeshi Matsumura
Wireless Systems Laboratory, Wireless Networks Research Center,
National Institute of Information and Communications Technology (NICT), Yokosuka, Kanagawa, Japan
E-mail: arif.dataesatu@nict.go.jp

***Abstract*—Cybernetic avatars (CAs) require high-capacity uplink for multimodal sensory streaming in Private 5G networks, creating resource allocation challenges. Genetic algorithms (GAs) effectively optimize resource allocation by exploring solution spaces, but common implementations use fixed parameters (population size, generation count, mutation/crossover rates) regardless of network complexity. This causes inefficiencies: small scenarios waste resources on slow exhaustive search, while large deployments terminate prematurely with suboptimal solutions. This paper proposes a deep neural network (DNN)-driven adaptive GA framework where a neural network trained on diverse scenarios predicts optimal GA parameters based on network state features, and the GA applies these parameters to optimize resource allocation. This approach automatically scales algorithm complexity to match problem requirements across varying network loads. Evaluation comparing resource allocation outcomes between DNN-predicted parameters and representative fixed-parameter baselines (minimal, balanced, extensive configurations) demonstrates that the proposed approach achieves 54% faster convergence than balanced configurations and 72% faster than extensive configurations while maintaining comparable throughput and QoS, demonstrating effective balance between convergence speed and solution quality.**

***Index Terms*—Deep Neural Network, Genetic Algorithm, Adaptive Parameter Selection, Resource Allocation, Cybernetic Avatar, Private 5G Network**

I. Introduction

Cybernetic avatar (CA) technology enables remote human presence through robotic surrogates, with teleoperators (TOs) controlling distant machines for applications spanning telemedicine, disaster response, and industrial automation [1], [2]. Japan's Moonshot R&D programs target seamless human-machine integration requiring ultra-reliable low-latency wireless connections for bidirectional control and sensory data streams. To support such demanding applications, Japan has allocated dedicated spectrum through Local 5G frameworks [3], enabling private network deployments with controlled interference and guaranteed capacity.

Multi-teleoperator multi-CA deployments present unique resource allocation challenges. Unlike conventional mobile broadband where downlink dominates traffic, CA systems generate asymmetric uplink-heavy patterns as robots continuously stream multimodal sensory data to remote operators. This uplink congestion intensifies in scenarios with multiple simultaneous CA operations, where resource block (RB) contention directly impacts control responsiveness and quality of service (QoS). Recent work [4] addressed this challenge through dynamic multi-layer service area adjustment, proposing a heuristic method that adaptively shifts coverage boundaries to improve fairness in resource allocation. While this approach demonstrates effectiveness in balancing QoS across CAs, heuristic methods rely on predetermined adjustment rules that may not discover globally optimal boundary configurations, particularly in complex multi-CA scenarios requiring exploration of large solution spaces.

Genetic algorithms (GAs) offer a promising alternative for boundary optimization. Extensively applied to wireless resource allocation [5], [6], GAs employ population-based search that escapes local optima through crossover and mutation operations, unlike heuristic approaches that perform incremental adjustments [4]. However, GA performance critically depends on parameter selection: population size, generation count, crossover/mutation rates, and selection strategies [7], [8]. Common implementations employ fixed parameters tuned through trial-and-error, creating fundamental inefficiencies when network complexity varies. Sparse deployments require minimal exploration while dense scenarios demand extensive search, but identical parameters cause wasted computation for simple cases or premature termination for complex ones. Machine learning for algorithm configuration [9], [10] addresses this by learning mappings from problem characteristics to effective parameters, enabling automated tuning that adapts to specific instances.

This paper proposes a DNN-driven adaptive GA framework that addresses this limitation through learned parameter adaptation. While parameter tuning via machine learning is an established concept [9], [10], our work specifically applies this approach to wireless resource allocation in CA networks, providing a lightweight deterministic alternative to more complex reinforcement learning-based radio access network (RAN) intelligence methods. The main contributions are:

- DNN-driven framework: A two-stage approach where a deep neural network predicts GA parameters based on real-time network features, enabling automatic algorithm complexity adaptation without the training overhead of online reinforcement learning methods
- Comprehensive evaluation comparing resource allocation outcomes between DNN-predicted parameters and three representative fixed-parameter baselines (minimal, bal-

This research has been supported by JST Moonshot R&D Goal 1, Grant Number JPMJMS2216.

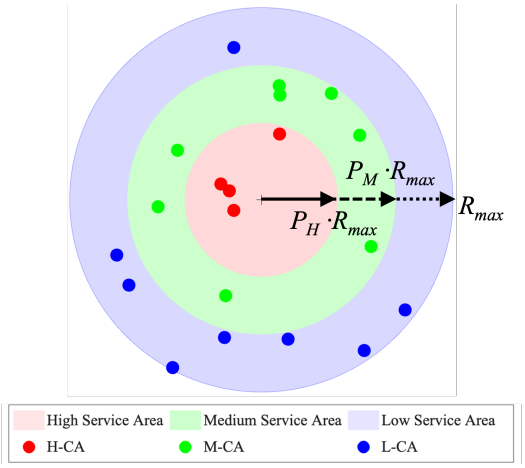

Fig. 1: Multi-tier service area framework with concentric priority areas.

anced, extensive) across diverse network loads
- Demonstration that the proposed approach achieves faster convergence while maintaining superior solution quality, effectively overcoming the inherent speed-quality tradeoff of fixed configurations

The remainder of this paper proceeds as follows. Section II presents the system model and GA-based optimization framework. Section III describes the proposed DNN-driven parameter adaptation mechanism. Section IV details the experimental setup. Section V presents performance evaluation results. Section VI concludes the paper.

II. System Model and GA-Based Optimization

This section describes the multi-tier service area framework for CA networks and formulates the resource allocation optimization problem addressed by genetic algorithms.

A. Multi-Tier Service Area and Priority-Based Resource Scheduling Framework

We adopt the multi-tier service area architecture from [4], which organizes coverage into three concentric areas surrounding the base station (BS): High service area ($0 \leq d \leq P_H \cdot R_{max}$), Medium service area ($P_H \cdot R_{max} < d \leq P_M \cdot R_{max}$), and Low service area ($P_M \cdot R_{max} < d \leq R_{max}$). Here, d represents the distance from BS to CA, P_H and P_M are normalized boundary parameters ($0 < P_H < P_M < 1$) determining the radial extent of High and Medium areas respectively, and R_{max} is maximum cell radius, as illustrated in Fig. 1.

Resource allocation follows priority-based scheduling where CAs in higher service areas receive preferential access to network resources. For each CA indexed by k at distance $d^{(k)}$ from the BS, priority values are assigned based on area membership determined by boundary parameters:

$$\omega^{(k)}(P_H, P_M) = \begin{cases} 3, & \text{if } d^{(k)} \leq P_H \cdot R_{max} \\ 2, & \text{if } P_H \cdot R_{max} < d^{(k)} \leq P_M \cdot R_{max} \\ 1, & \text{if } P_M \cdot R_{max} < d^{(k)} \leq R_{max} \end{cases} \tag{1}$$

where throughput requirement $T_{req}^{(k)}$ also varies by area (detailed in Section IV). This formulation explicitly shows how boundary parameters P_H and P_M control priority assignment for each CA, thereby determining resource allocation order.

These priority weights determine resource block allocation order, where CAs with higher $\omega^{(k)}$ receive allocation precedence when resources are constrained. The achieved throughput for CA k depends on allocated resource blocks through this priority-based scheduling mechanism:

$$T_{achieved}^{(k)} = \min(RB_{allocated}^{(k)} \cdot SE^{(k)} \cdot RB_{cap}, T_{req}^{(k)}) \tag{2}$$

where $RB_{allocated}^{(k)}$ is determined by the priority scheduler using weights from Eq. (1), $SE^{(k)}$ represents spectral efficiency, and RB_{cap} denotes throughput capacity per resource block.

B. GA-Based Boundary Optimization

While the priority-based framework in Eq. (1) assigns resources based on area membership, the boundary parameters P_H and P_M themselves critically determine network performance. Static boundary values fail to adapt to varying CA distributions and channel conditions, leading to suboptimal resource allocation. Fixed boundaries may over-prioritize nearby CAs while starving distant CAs, or conversely waste resources on low-priority areas. The resource allocation problem therefore seeks to determine optimal service area boundaries that maximize aggregate network throughput. The optimization objective is formulated as:

$$\text{maximize} \quad F_{TP}(\mathbf{P}) = \sum_{k \in \mathcal{U}} T_{achieved}^{(k)}(\mathbf{P}) \tag{3}$$

where $\mathbf{P} = \{P_H, P_M\}$ represents boundary parameters and $\mathcal{U}$ denotes the set of CAs served by the BS.

The GA approach [5], [6] encodes candidate solutions as boundary parameter vectors, initializes a population through uniform random sampling, and iteratively evolves the population through selection, crossover, and mutation operations. Fitness evaluation directly implements the throughput maximization objective in Eq. (3). Tournament selection chooses parents, arithmetic crossover creates offspring by blending boundary parameters, and Gaussian mutation introduces controlled perturbations. The complete optimization workflow is illustrated in Fig. 2, showing both traditional fixed-parameter GA and the proposed machine learning (ML)-adaptive approach.

Key GA parameters include:
- Population size N_{pop}: Number of candidate solutions maintained
- Maximum generations G_{max}: Iteration budget
- Crossover rate p_c: Probability of crossover operation

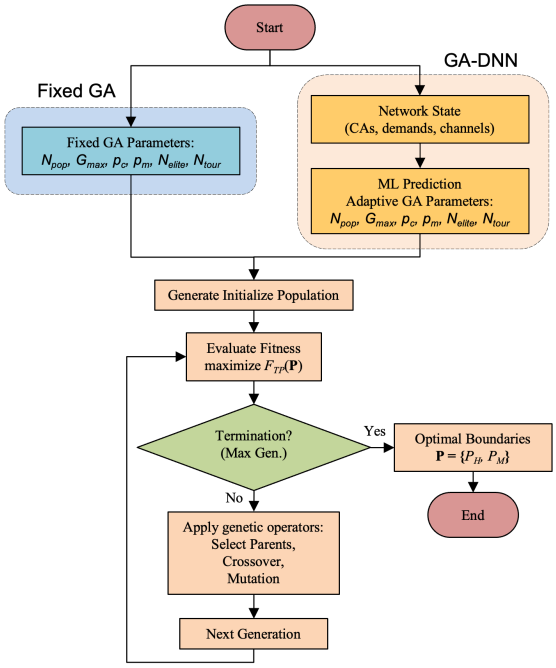

Fig. 2: Flow diagram of GA and GA_{DNN} optimization frameworks.

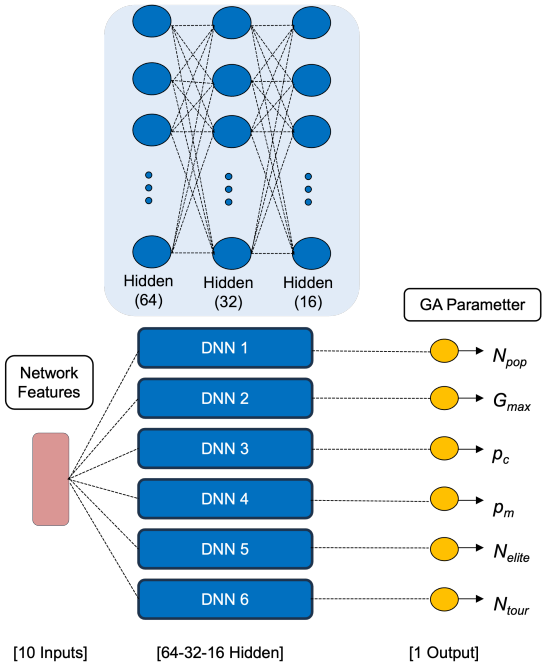

Fig. 3: Independent DNN architecture for GA parameter prediction.

- Mutation rate p_m: Probability of mutation
- Elite count N_{elite}: Top solutions preserved
- Tournament size N_{tour}: Selection pool size

III. Proposed DNN-Driven Adaptive GA

A. Motivation: Fixed Parameter Limitations

Fixed GA parameters cannot adapt to varying network complexity, as shown in the left branch of Fig. 2. This creates an inherent speed-quality tradeoff: minimal configurations achieve rapid convergence but may terminate prematurely for complex scenarios; extensive configurations ensure thorough exploration but waste computation on simple cases. An adaptive approach, as illustrated in the right branch of Fig. 2, can overcome this limitation.

B. ML-Based Parameter Prediction

Our approach employs a deep neural network that predicts optimal GA parameters based on real-time network features. The prediction pipeline consists of three stages, as illustrated in the right branch of Fig. 2.

1) Feature Extraction: Given current network state (CA positions, channel conditions, resource demands), we extract a 10-dimensional feature vector:

$$\mathbf{f} = [n_{CA}, \mu_{dist}, \sigma_{dist}, \mu_{SNR}, \sigma_{SNR}, D_{total}, \rho_{util}, p_H, p_M, p_L] \quad (4)$$

where n_{CA} is CA count, μ_{dist} and σ_{dist} capture spatial distribution statistics, μ_{SNR} and σ_{SNR} characterize signal quality, D_{total} is total resource demand, ρ_{util} is current utilization ratio, and p_H, p_M, p_L represent proportion of CAs in each service tier.

2) Neural Network Architecture: We employ an ensemble of 6 independent deep neural networks (DNNs), where each network predicts one GA parameter using identical feedforward architecture [10-64-32-16-1]. This independent regression approach offers several advantages over multivariate regression: (1) each network specializes in predicting a single parameter, simplifying the optimization landscape; (2) networks can be trained asynchronously in parallel, reducing training time; (3) parameter-specific architectures can be individually tuned if needed. Each network maps 10 input features to 1 output parameter through three hidden layers with progressive dimension reduction [64→32→16], employing linear activation (purelin) in the output layer for regression, as shown in Fig. 3. The networks are trained using scaled conjugate gradient optimization with mean squared error (MSE) loss function. During deployment, all 6 networks execute simultaneously to produce the complete parameter set $\{N_{pop}, G_{max}, p_c, p_m, N_{elite}, N_{tour}\}$.

Training data generation employs supervised learning with labels derived from exhaustive GA parameter exploration across diverse network conditions (5-50 CAs, varying spatial distributions, multiple channel realizations). For each of 1,000 network scenarios, we evaluate 100 GA parameter combinations sampled from practical ranges, execute complete optimization workflows, and measure convergence time, throughput, and QoS achievement. An efficiency score identifies the best-performing parameter set as the training label $(\mathbf{f}_i, \mathbf{p}_i^*)$. This data-driven approach spanning 100,000 total executions ensures the DNN learns genuinely optimal parameters rather than arbitrary selections. The 10-dimensional feature vector

abstracts network state into generalizable representations (network density, spatial heterogeneity, service distribution) that transcend individual CA placements, enabling generalization to unseen scenarios.

3) Parameter Prediction and Application: The right branch of Fig. 2 illustrates the $\mathrm{GA_{DNN}}$ adaptive workflow. During deployment, for each optimization instance:

1) Network State: Extract network features $\mathbf{f}$ from current CA distribution, channel conditions, and resource demands
2) ML Prediction: Normalize features to [0,1] using min-max scaling parameters from training data, forward propagate through neural network, and denormalize predicted parameters to obtain $\{N_{pop}, G_{max}, p_c, p_m, N_{elite}, N_{tour}\}$
3) Apply constraints (minimum population size, generation bounds, rate ranges)
4) GA Execution: Execute GA with predicted parameters through standard evolutionary cycle shown in the flowchart (initialize population, evaluate fitness, check termination, apply genetic operators, advance to next generation)

As shown in Fig. 2, following ML prediction, both traditional GA and $\mathrm{GA_{DNN}}$ execute identical optimization loops. The key distinction is in parameter selection: traditional GA uses fixed parameters for all scenarios, while $\mathrm{GA_{DNN}}$ dynamically adjusts parameters based on network state. This enables lightweight configurations for simple networks and extensive search for complex deployments, achieving the efficiency-performance balance demonstrated in Section V.

IV. Simulation Setup

A. Network Configuration

System-level simulations are implemented in MATLAB. Evaluation employs a Local 5G deployment with 50-m cell radius. The base station operates with 273 RBs (100 MHz bandwidth, 30 kHz subcarrier spacing). CAs are uniformly distributed within the coverage area with throughput requirements for real-time teleoperation. Key simulation parameters are summarized in Table I.

TABLE I: Key Simulation Parameters

Parameter	Value
Network Deployment	
Coverage area	50 m radius
Carrier frequency	4.85 GHz
Bandwidth	100 MHz (273 RBs)
Subcarrier spacing	30 kHz
Number of CAs	5–50
Throughput requirements	H: 25; M: 7; L: 3 Mbit/s
Fixed GA Parameters	
$\mathrm{GA_{30}}$	$N_{pop} = 30$, $G_{max} = 20$
$\mathrm{GA_{300}}$	$N_{pop} = 300$, $G_{max} = 100$
$\mathrm{GA_{500}}$	$N_{pop} = 500$, $G_{max} = 150$
Crossover rate	0.75–0.85
Mutation rate	0.10–0.20
ML Parameters	
Architecture	6 independent DNNs [10-64-32-16-1]
Output activation	Linear (purelin)
Loss function	Mean Squared Error (MSE)
Training algorithm	Scaled Conjugate Gradient
Training samples	Diverse network scenarios
Data split	70/15/15 train/val/test

B. Comparative Methods

To evaluate the proposed $\mathrm{GA_{DNN}}$ adaptive approach, we compare against static allocation and three representative fixed-parameter GA configurations spanning the speed-quality spectrum:

- Static: Baseline priority-based resource allocation without optimization
- $\mathrm{GA_{30}}$: Minimal configuration for rapid convergence with $N_{pop} = 30$, $G_{max} = 20$, $p_c = 0.8$, $p_m = 0.25$, $N_{elite} = 3$, and $N_{tour} = 3$
- $\mathrm{GA_{300}}$: Moderate configuration balancing speed and quality with $N_{pop} = 300$, $G_{max} = 100$, $p_c = 0.8$, $p_m = 0.3$, $N_{elite} = 50$, and $N_{tour} = 30$
- $\mathrm{GA_{500}}$: Extensive configuration for exhaustive search with $N_{pop} = 500$, $G_{max} = 150$, $p_c = 0.8$, $p_m = 0.25$, $N_{elite} = 100$, and $N_{tour} = 50$
- $\mathrm{GA_{DNN}}$: Adaptive configuration with DNN-predicted parameters

These fixed variants demonstrate the inefficiency of non-adaptive parameter selection across varying network loads. Performance is assessed through Monte Carlo simulation across CA counts to ensure statistical reliability.

V. Performance Evaluation

Table II presents representative $\mathrm{GA_{DNN}}$ parameter predictions across varying network scales, demonstrating the framework's automatic adaptation to problem complexity. The predicted parameters exhibit several notable patterns: population sizes (N_{pop}) remain relatively stable (123–148) across all scales, suggesting that solution space dimensionality in boundary optimization does not scale dramatically with CA count; generation budgets (G_{max}) show minimal variation (46–48), indicating consistent convergence requirements; crossover rates (p_c) gradually decrease from 0.75 to 0.71 as network density increases, favoring more conservative reproduction for complex scenarios; mutation rates (p_m) progressively increase from 0.12 to 0.14, promoting greater exploration in dense deployments; elite preservation (N_{elite}) grows from 14 to 19, retaining more high-quality solutions as problem complexity increases; tournament sizes (N_{tour}) incrementally increase from 11 to 13, reflecting larger selection pools for denser networks. These adaptive patterns contrast sharply with fixed-parameter approaches that maintain identical settings regardless of network conditions.

A. Performance vs. Network Load

Performance across varying network loads reveals distinct characteristics among the optimization approaches. Fig. 4 shows throughput evolution as CA count increases from 5 to

TABLE II: GA_{DNN} Predicted Parameters Across Network Scales

CAs	N_{pop}	G_{max}	p_c	p_m	N_{elite}	N_{tour}
5	148	46	0.75	0.12	14	11
10	144	46	0.75	0.12	15	11
20	133	47	0.74	0.12	15	11
30	125	47	0.73	0.13	16	11
40	123	47	0.72	0.13	17	12
50	124	48	0.71	0.14	19	13

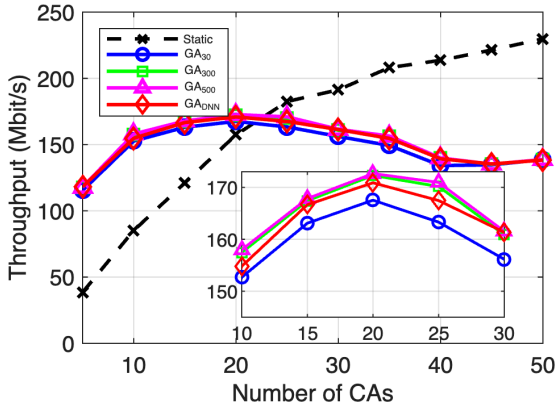

Fig. 4: Total throughput comparison across varying CA densities.

50. Static allocation appears to achieve competitive throughput, but this reflects serving fewer CAs due to QoS failures rather than genuine performance. Among GA-based approaches, GA_{30} exhibits consistently lower throughput across all network loads due to premature convergence that finds suboptimal boundaries, while GA_{300} and GA_{500} achieve higher throughput through more extensive exploration. The proposed GA_{DNN} intelligently adapts between these extremes, matching GA_{300} throughput quality while achieving significantly faster convergence.

Convergence time behavior, illustrated in Fig. 5, demonstrates the adaptive nature of the proposed approach. The fixed-parameter baselines show predictable patterns: GA_{30} maintains sub-20 ms convergence across all loads due to its minimal configuration, while GA_{300} and GA_{500} exhibit linear growth with CA count reflecting their larger population sizes and generation budgets. GA_{DNN} demonstrates intelligent adaptation, operating near GA_{30} speed for small networks (5-15 CAs) where simple parameter settings suffice, then transitioning toward GA_{300} parameters for larger deployments (30-50 CAs) where more extensive search becomes beneficial.

QoS achievement rates, presented in Fig. 6, highlight the critical advantage of optimization-based approaches. Static allocation maintains 100% QoS only up to 20 CAs, then degrades sharply to 24.8% at 50 CAs as resource contention intensifies. All GA-based approaches sustain near-perfect QoS (100% for 5-35 CAs, 90.2% at 50 CAs), with identical performance across fixed and adaptive configurations. This demonstrates that GA_{DNN} preserves solution quality while

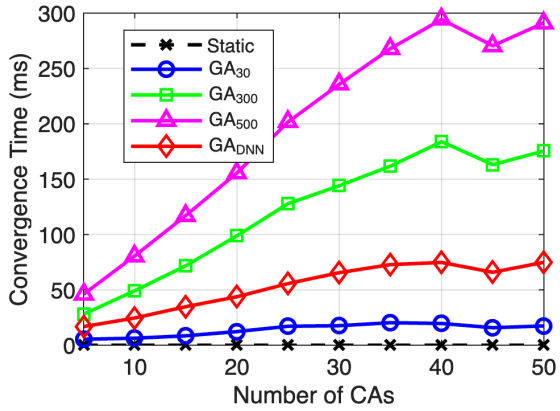

Fig. 5: Convergence time comparison showing GA_{DNN} adaptive scaling.

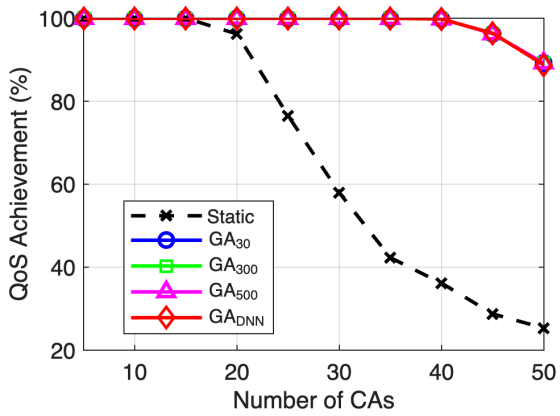

Fig. 6: QoS achievement comparison showing GA superiority over static allocation.

improving convergence efficiency through adaptive parameter selection, maintaining robustness across varying network densities through constraint enforcement for extreme scenarios.

Resource utilization patterns, shown in Fig. 7, further demonstrate optimization effectiveness. GA-based approaches achieve higher utilization (96-98%) compared to static allocation (83%), indicating more efficient resource usage through dynamic boundary adaptation. The consistent utilization across all GA variants confirms that adaptive parameter selection does not compromise resource efficiency.

B. Overall Performance Summary

Table III presents average metrics across all scenarios. The fixed-parameter baselines expose a fundamental speed-quality tradeoff: GA_{30} achieves fast convergence (14.02 ms) but reduced throughput (147.51 Mbit/s), GA_{300} improves throughput to 151.59 Mbit/s but requires 8.6× longer convergence (120.42 ms), while GA_{500} yields only marginal 0.21% throughput gain with 63% more convergence time, demonstrating diminishing returns.

Fig. 8 visualizes this tradeoff relationship. The proposed GA_{DNN} breaks this tradeoff, achieving comparable throughput to GA_{300} (150.74 Mbit/s vs 151.59 Mbit/s) while delivering

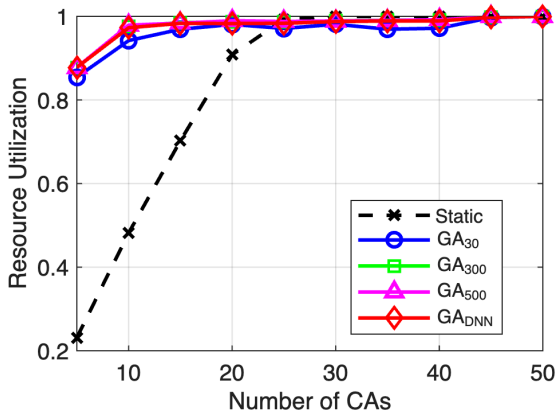

Fig. 7: Resource utilization comparison across all methods.

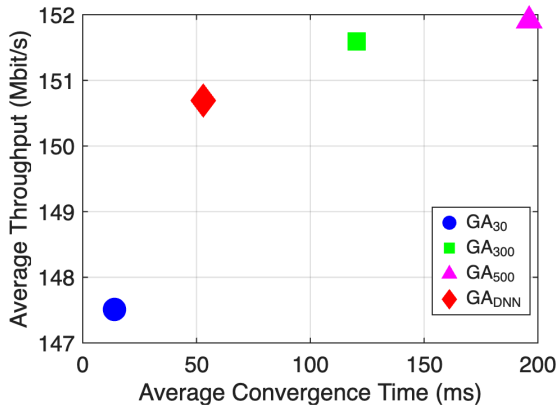

Fig. 8: Time vs performance tradeoff showing $\mathrm{GA}_{\mathrm{DNN}}$ optimal balance.

TABLE III: Overall Average Performance

Method	Avg Throughput (Mbit/s)	Avg QoS (%)	Avg Util	Avg Time (ms)
Static	164.90	66.31	0.83	0.00
GA_{30}	147.51	98.53	0.96	14.02
GA_{300}	151.59	98.53	0.98	120.42
GA_{500}	151.91	98.53	0.98	196.11
$\mathrm{GA}_{\mathrm{DNN}}$	150.74	98.53	0.98	55.72

54% and 72% faster convergence than GA_{300} and GA_{500} respectively (55.72 ms vs 120.42 ms and 196.11 ms). Compared to GA_{30}, $\mathrm{GA}_{\mathrm{DNN}}$ achieves 2.19% higher throughput while maintaining identical QoS and resource utilization. This demonstrates that adaptive parameter selection effectively matches algorithm complexity to problem requirements, automatically selecting lightweight configurations for simple scenarios while scaling up for complex deployments.

VI. Conclusion

This paper presented a deep neural network-driven adaptive GA framework for resource allocation in cybernetic avatar networks. The approach employs a neural network to predict optimal GA parameters based on real-time network features, which the GA then applies to optimize resource allocation. Evaluation across diverse network loads (5-50 CAs) demonstrated that the proposed approach achieves 54% and 72% faster convergence than balanced and extensive fixed configurations respectively, while maintaining comparable throughput (150.74 Mbit/s vs 151.59 Mbit/s) and superior resource utilization. Compared to minimal fixed configurations, $\mathrm{GA}_{\mathrm{DNN}}$ achieves 2.19% higher throughput, demonstrating that the DNN-driven framework effectively matches algorithm complexity to problem requirements by automatically adapting parameters based on network conditions. While effective for scenarios within the training distribution, the current implementation assumes offline training and may require periodic retraining for deployments experiencing novel conditions. The DNN prediction overhead (approximately 8 ms) remains negligible compared to GA convergence time (55.72 ms average).

Future directions include extending the framework to multi-objective optimization, incorporating mobility-aware prediction, investigating transfer learning across deployment environments, and developing hybrid approaches combining DNN predictions with online adaptation mechanisms.

References

[1] T. Matsumura, H. Murakami, A. Wakayama, and K. Ibuka, "Toward Reliable End-to-End Communication for Global Cybernetic Avatar Teleoperation," in *2025 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2025.

[2] N. Hagita, R. Kanai, H. Ishiguro, K. Minamizawa, F. Arai, F. Shimpo, T. Matsumura, and Y. Yamanishi, "Cybernetic Avatars: Teleoperation Technologies from In-Body Monitoring to Social Interaction," *Science Robotics*, vol. 9, no. 96, 2024.

[3] Ministry of Internal Affairs and Communications (MIC), "Japan's Efforts to Promote the Prevalence of Local 5G." [Online]. Available: https://www.soumu.go.jp

[4] A. Dataesatu, A. Wakayama, K. Ibuka, H. Murakami, and T. Matsumura, "Fair Resource Allocation with Dynamic Multi-Layer Service Area Management in Local Networks Supporting Cybernetic Avatars," in *2025 IEEE 101st Vehicular Technology Conference (VTC2025-Spring)*, 2025, pp. 1–7.

[5] X. Liu, W. Zhang, and H. Wang, "Energy Efficient Resource Allocation for 5G Heterogeneous Networks Using Genetic Algorithm," in *2021 IEEE Globecom Workshops*. IEEE, 2021, pp. 1–6.

[6] J. Li, Q. Zhang, and W. Liang, "Improved Genetic Algorithm Based Intelligent Resource Allocation in 5G Ultra Dense Networks," in *2018 IEEE International Conference on Communications*. IEEE, 2018, pp. 1–6.

[7] G. Karafotias, M. Hoogendoorn, and A. E. Eiben, "Parameter Control in Evolutionary Algorithms: Trends and Challenges," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 2, pp. 167–187, 2015.

[8] A. Aleti and I. Moser, "A Systematic Literature Review of Adaptive Parameter Control Methods for Evolutionary Algorithms," *ACM Computing Surveys*, vol. 49, no. 3, pp. 1–35, 2016.

[9] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential Model-Based Optimization for General Algorithm Configuration," in *Learning and Intelligent Optimization (LION 2011)*, ser. Lecture Notes in Computer Science, vol. 6683. Springer, 2011, pp. 507–523.

[10] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter, "Efficient and Robust Automated Machine Learning," p. 2755–2763, 2015.