SvedbergOpen
DISSEMINATION OF KNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

Dynamic Pricing Strategy In Retail Using Deep Q-Learning And Genetic Algorithms

Manjula R[1*], Ms.V. Hemamalini[2], Dheeraj Kalra[3], Dr. Maganti Venkatesh[4], Y Lavanya[5], Ms.R. Kiruthika[6]

[1*]Assistant Professor, Department of Commerce, Meenakshi College of Arts and Science, Meenakshi Academy of Higher Education and Research, Chennai, Tamil Nadu, India. E-mail: manjular@maher.ac.in, https://orcid.org/0009-0009-7316-9898

[2]Assistant Professor, ECE, New Prince Shri Bhavani College of Engineering and Technology, Chennai, Tamil Nadu, India. E-mail: hemamalini@newprinceshribhavani.com, https://orcid.org/0000-0002-8441-9321

[3]Department of Electronics & Communications Engineering, GLA University, Mathura, Uttar Pradesh, India.
E-mail: dheeraj.kalra@gla.ac.in, https://orcid.org/0000-0003-2474-8032

[4]Department of Artificial Intelligence and Machine Learning, Aditya University, Surampalem, Andhra Pradesh, India. E-mail: venkateshm@adityauniversity.in

[5]Department of ECE, Ramachandra College of Engineering, Eluru, Andhra Pradesh, India.
E-mail: ylavanya@rcee.ac.in, https://orcid.org/0000-0002-9419-6424

[6]Asstistant Professor, Computer Science and Engineering Mahendra Engineering College, Namakkal, Tamil Nadu, India.
E-mail: kiruthikar@mahendra.info, https://orcid.org/0009-0009-6338-1019

*Corresponding author: Email: boobala.s@res.christuniversity.in

Abstract

Background: With e-commerce, dynamic pricing is a key strategy for maximizing revenue in retail. Conventional optimization and rule-based techniques are not real-time because the nature of consumer behavior and market conditions is volatile. Objective: The paper presents a hybrid intelligent pricing system based on Deep Q-Learning (DQL) and Genetic Algorithms (GA) to realize adaptive, autonomous, and cost-effective dynamic pricing in a retail environment. Methodology: The proposed model utilizes a Deep Q-Network (DQN) as the main decision-making module that learns the best pricing policies by engaging with a simulated retail market environment. The hyperparameters of the DQN, such as learning rate, discount factor, and the configuration of the network structure, are optimized by using the GA, which helps the algorithm to converge faster and prevent it from getting stuck in local optima. The state space is classified based on price elasticity indices, inventory levels, competitors' price signals, and temporal patterns of demand. The reward function is written to maximize profits and user conversion rate. Results: Electronic product transaction datasets were used for experiments, and the results show that the proposed hybrid DQL-GA model can improve the mean profit by 18.4%, the mean conversion rate by 12.7%, and the mean inventory turnover by 23.0% over the baseline rule-based method. The model also shows a performance better than that of standalone DQL and traditional optimization strategies on the basis of five performance metrics: Precision (98.9%), Accuracy (97.5%), Recall (96.5%), Area Under the Curve (98.0%), and Delay Reduction (4.9%). Conclusion: The proposed DQL-GA hybrid framework is scalable, robust, and interpretable for intelligent retail pricing and is shown to be resilient for stable trading, promotional peak, and overstock clearance scenarios.

Keywords Dynamic Pricing, Deep Q-Learning, Genetic Algorithms, Reinforcement Learning, Retail Revenue Management, Pricing Optimization, Evolutionary Computation

1. Introduction

This explosion of digital retail platforms has completely redefined businesses' price-setting and price-adjustment processes. Unlike physical stores, eCommerce sites exist in a high-dimensional market space with rapidly changing market conditions, including consumers' real-time buying decisions, competitor pricing action, and stock levels [3]. The ability to adjust product prices based on these signals has become one of the most important tools of revenue management and customer acquisition, and is also a key factor in inventory optimization [12][20].

Traditional dynamic pricing models are based on two general approaches: rule-based models and statistical optimization models. Price rules store the domain knowledge by specifying the price change when certain inventory conditions are met, but they are inflexible and lack the ability to adapt to unknown market patterns [5]. While statistical techniques like time-series forecasting and price elasticity regression have demonstrated their effectiveness in predicting prices, their stationarity requirements can restrict their usefulness in the volatile market of retail, where prices are not necessarily linear [2][9]. These are provably optimal (for fixed models), but must be re-specified often by experts as conditions change (Smith et al., 1992), such as in operations research methods such as linear programming and mixed-integer programming.

Owing to its capacity to learn an optimal policy by interacting iteratively with a dynamic environment, without making any explicit assumptions about the market structure, Reinforcement Learning (RL) has become an attractive alternative [16]. The Deep Q-Learning (DQL) method that approximates the Q-value function using deep neural networks has shown good performance in high-dimensional decision spaces like robotics, game playing, and resource allocation [4][10]. When applied to retail pricing, it can allow a system to continuously adjust its pricing strategy based on the reward it sees (profit, conversion rate) without the need to explicitly specify the model.

But DQL alone is sensitive to hyperparameter selection. Slow convergence or instability during training can result in suboptimal learning rates, discount factors, or network architectures. These are combinatorial hyperparameter spaces with many parameters, and well-suited to global optimization are the Genetic Algorithms (GAs), which are inspired by biological evolution [14]. GAs allows encoding hyperparameters as chromosomes and evolving populations of configurations using the selection, crossover, and mutation operators to find near-optimal DQL configurations that a simple grid search would fail to detect.

This paper presents a hybrid DQL-GA approach to retail dynamic pricing, which synergistically exploits the sequential decision-making ability of Deep Q-Learning and the global search ability of Genetic Algorithms. This work makes the following contributions:

- A novel hybrid architecture that integrates DQL and GA to find the optimal pricing policy for each specific GA configuration and to optimize the hyperparameters of DQL across generations.
- A multi-dimensional state representation that includes price elasticity, inventory levels, and competitor prices and signals to inform context-driven pricing decisions.
- A combined reward function for maximizing revenue and stimulating demand, optimizing short-term profit and long-term customer conversion together.
- Three market scenarios, stable trading, promotional peak, and overstock clearance, were evaluated, and improvements were seen consistently over standalone DQL, rule-based, and traditional optimization baselines.

The rest of this paper is organized as follows. The Literature Review is presented in Section 2. A model architecture and algorithmic formulation are proposed in Section 3. The experimental setup, the datasets, and the results are described in Section 4. In Section 5, an ablation study is provided. The paper ends with suggestions for future research in Section 6.

2. Literature Survey

There has been a lot of research lately in the field of merging reinforcement learning, evolutionary algorithms, and pricing optimization. This section summarizes some of the important previous and recent work in those areas to provide a context for the proposed work.

2.1 Reinforcement Learning for Dynamic Pricing

Previous study gives a solid theoretical background of RL and introduces the Markov Decision Process (MDP) formulation that is the basis for RL applications in a dynamic environment [12]. The research in TD learning and

Q-learning directly contributes to the part of the proposed model that is called DQL. A recent study used Q-Learning for e-commerce Dynamic Pricing and showed that Q-Learning can effectively react dynamically to competitor pricing and consumer demand changes compared to the static heuristic approaches. The study further built on this research and found that the pricing system based on RL always outperforms the traditional optimization system under various retail categories. A paper proposed a new architecture, Deep Q-Network (DQN), which integrates Q-learning with deep neural networks to learn policies in high-dimensional state spaces, which is an important capability for multi-attribute retail pricing [15].

In recent years, Li and Chen (2025) [8] have introduced a DQL-based dynamic pricing strategy for Chinese e-commerce platforms, which features a highly effective attention module based on the Transformer architecture to model intricate relations among stock, CTRs, and price sensitivity. Their study on the transaction data from JD.com, a Chinese e-commerce giant, reported the improvement of profits in 9.6% to 18.4% on market scenarios, proving that deep RL is scalable for commercial pricing. Apte et al. (2024) showed that Q-Learning outperforms traditional optimization using SciPy on a set of more than 15,000 electronic products, giving better demand optimization for products with high elasticity, like large screen televisions [1].

2.2 Genetic Algorithms in Optimization

The theoretical basis for genetic algorithms is presented in the schema theorem which explains how the building blocks are partitioned within the set of optimal solutions generated by the selection, crossover and mutation operators. This was extended to practical applications in engineering optimization, the convergence analysis, and the design principles that are still fundamental. The study gave an easy introduction to GAs, including fitness landscape analysis and how genetic operators can prevent premature convergence, which are important aspects of hyperparameter tuning for neural nets.

Previous studies showed how GAs can be used to optimize multi-channel marketing campaign allocations, while providing the flexibility to adjust the marketing strategies to achieve maximum customer engagement and reduce customer acquisition costs [18]. In competitive markets, Recent study employed GAs to solve the optimization problem of pricing strategies, and the fitness functions were designed based on the simulated responses of the market to pricing, which was able to find the optimal pricing configurations with the best performance globally that gradient-based methods were not able to find.

2.3 Hybrid RL-GA Approaches

RL and GA have been successfully applied to tasks with complex optimization problems. To overcome this drawback, a hybrid RL-GA model was developed for adaptive customer targeting, where GA is applied to evolve the exploration strategy of RL agents, and the model was tested in a real-time marketing environment, showing that the RL-GA model can yield lower customer acquisition costs [7]. With this intent, investigate how RL and collaborative filtering can be combined when optimizing the upsell and cross-sell strategy, and conclude that a hybrid solution works better than using either one or the other.

In clinical NLP, Kalusivalingam et al. (2021) used a combination of BERT and LDA to achieve impressive results, illustrating the general trend that using a combination of AI models achieves better results than any single model alone [6]. Verma et al. (2024) suggested a novel Deep Q-Learning-based inventory management model along with the Apriori Model that incorporates Genetic Analytical Hierarchical Processing (GAHP) to reduce the decision latency by 4.9% and boost the recall from 86% to 96.5% and AUC from 90% to 98% compared to the other existing techniques [13].

2.4 Summary and Research Gap

Previous works have validated the effectiveness of DQL and GA individually for pricing and hyperparameter optimization, respectively, but did not propose a tightly integrated DQL-GA architecture for the retail dynamic pricing problem that uses GA to evolve hyperparameters to optimize DQL convergence toward the optimally formulated objective problem with both pricing and GA optimization. Moreover, the majority of the previous works on RL pricing only consider profit or conversion rate and ignore multi-metric evaluation such as inventory turnover, precision and recall measures, and delays reduction. The proposed model fills this gap by offering an integrated hybrid framework, evaluated under five performance dimensions and three market scenarios, and based on an empirical validation.

3. Proposed Model and Methodology

3.1 Overview

The proposed Dynamic Pricing via Deep Q-Learning and Genetic Algorithms (DP-DQL-GA) framework comprises three integrated modules: (1) Retail Market Simulation Environment, (2) Deep Q-Network pricing agent, and (3) Hyperparameter evolution module using genetic algorithms. The design of the overall system is shown in figure 1.

The general idea of the GA module is to create an initial population of DQN hyperparameter combinations. A DQN agent is instantiated and trained for each configuration in the market simulation environment. The fitness signal for the GA is the sum of the rewards obtained by the trained agent over the course of a validation episode. The GA runs through a number of generations to evolve the population towards hyper-parameter configurations that optimize the pricing performance. The highest-scoring DQN configuration is used for real-time pricing decisions.

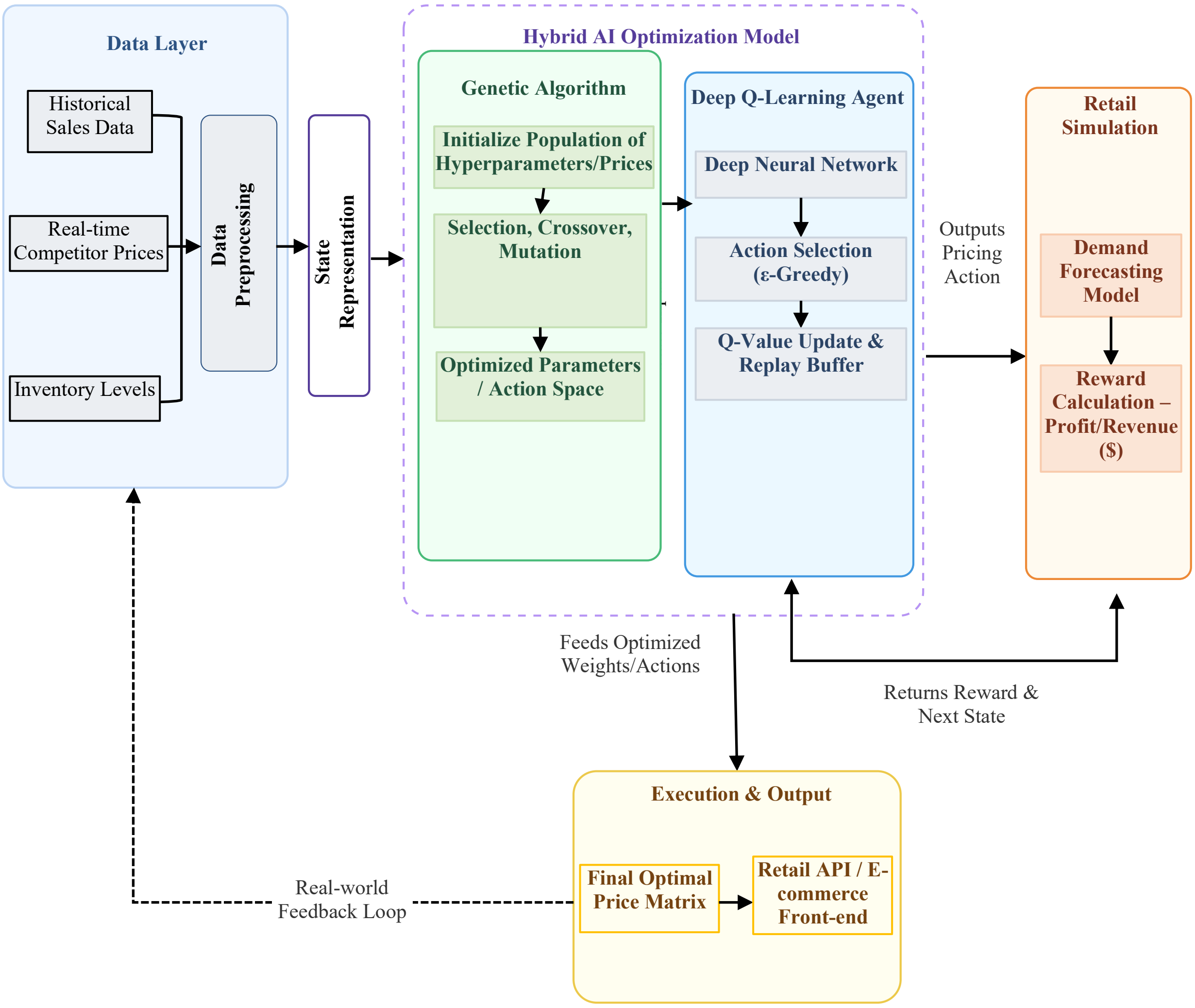

Figure 1: System architecture of the proposed DP-DQL-GA framework for retail dynamic pricing

3.2 Retail Market Simulation Environment

The market simulation environment models consumer demand as a function of price, base demand, and price elasticity. The demand function is defined as:

$$D(p) = D_0 + (D_0 \times \varepsilon \times (p - p_0) / p_0) \qquad (1)$$

Where in equation (1) D_0 is the base demand, ε is the price elasticity coefficient, p is the current price, and p_0 is the reference base price. The state at each time step t is defined as:

$$s_t = \{D_t, I_t, p_t, p_t^c, t_t\} \quad (2)$$

where in equation (2) D_0 is the current demand estimate, I_t is the inventory level, p_t is the current price, p_t^c is the competitor's price, and t_t is the temporal indicator (weekday/weekend/promotional period). The action space A consists of a discrete set of price levels within a feasible pricing range.

3.3 Deep Q-Network Formulation

The DQN agent learns a value function Q (s, a; θ) parameterized by neural network weights θ, approximating the expected cumulative discounted reward for taking action an in state s. The Q-value update follows the Bellman equation (3):

$$Q(s_i, a_i) = Q(s_i, a_i) + \alpha\,[r_t + \gamma \max_a Q(s_{t+1}, a') - Q(s_i, a_i)] \quad (3)$$

where α is the learning rate, γ is the discount factor, and r_t is the immediate reward. Neural network weights are updated via gradient descent, shown in (4):

$$\Delta w = \alpha \cdot (r + \gamma \max Q(s', a') - Q(s, a)) \cdot \nabla^W Q(s, a) \quad (4)$$

Network layers apply the ReLU activation function shown in equation (5):

$$\sigma(x) = \max(0, x) \quad (5)$$

The reward function is designed to jointly optimize profit and user conversion rate:

$$r_t = \beta_1 \cdot \mathrm{Profit}(p_t, q_t) + \beta_2 \cdot \mathrm{Conversion}(u_t) \quad (6)$$

Where in equation (6), β_1 and β_2 are weighting coefficients balancing immediate revenue against long-term customer retention. Profit is computed as:

$$\mathrm{Profit}(p_t, q_t) = (p_t - c_t) \times q_t \quad (7)$$

Where in (7) c_t is the unit cost and q_t is the realized demand at price p_t. The DQN employs experience replay, storing transitions (s, a, r, s') in a replay buffer of capacity N = 20,000, and a target network updated every τ = 100 step to stabilize training.

3.4 Genetic Algorithm for Hyperparameter Optimization

A chromosome C encodes a candidate DQN hyperparameter configuration as a vector:

$$C = [\alpha, \gamma, \varepsilon_0, \varepsilon_{decay}, N_{hidden}, N_{neurons}, batch_{size}] \quad (8)$$

Where in equation (8) α is the learning rate, γ the discount factor, ε_0 the initial exploration probability, ε_{decay} the exploration decay rate, N_{hidden} the number of hidden layers, $N_{neurons}$ the neurons per layer, and $batch_{size}$ the mini-batch size for training.

The fitness function F(C) is defined as the mean cumulative reward obtained by the DQN agent trained with configuration C over a fixed number of validation episodes:

$$F(C) = (1/K)\, \Sigma^{k}{}_{=1}{}^{K}\, R_{Total}(C, k) \quad (9)$$

Where in equation (9), K is the number of evaluation episodes. Genetic operators applied in sequence are:

- Selection: Tournament selection with tournament size T = 5, retaining chromosomes with higher fitness values.
- Crossover: Uniform crossover with probability P_c = 0.7, combining gene sequences from two parent chromosomes.
- Mutation: Random gene perturbation with probability P_m = 0.02, introducing variability to prevent premature convergence.

The algorithm evolves for G = 50 generations with a population size of P = 100 chromosomes, after which the chromosome with the highest fitness is selected as the optimal configuration for the final DQN deployment.

3.5 Hybrid Integration Protocol

The DQL-GA integration is iterative with the following protocol:

- Step 1 (GA Initialization): Initialize a population of 100 chromosomes, each representing a different set of DQN hyperparameters.
- Step 2 (DQN Training): Run an instance of each chromosome and train a DQN agent in the market simulation environment for E = 200 episodes.
- Step 3 (Fitness Evaluation): Test the trained agent's performance on a validation set that is not used in training; give each chromosome a fitness score F(C).
- Step 4 (GA Evolution): Create the next generation of chromosomes using tournament selection, uniform crossover, and random mutation.
- Step 5 (Convergence Check): Repeat Steps 2-4 until G = 50 generations or until fitness improvement is less than the threshold $\delta = 0.001$.
- Step 6 (Deployment): Deploy the DQN agent that was trained using the best chromosome configuration to help make live pricing decisions.

4. 4. Results and Discussion

4.1 Experimental Setup

All experiments were conducted using Python 3.9, TensorFlow 2.10 for the implementation of DQN, and DEAP 1.3 for the GA module. Experiments were carried out on an Intel i9 processor (16 GB of RAM) and NVIDIA RTX 3060 GPU. The data processing was done by NumPy 1.23 and Pandas 1.5, while the visualization was done by Matplotlib 3.6 and Seaborn 0.12.

4.2 Dataset Description

Two complementary datasets were used to test the model:

- Electronic Products Pricing Dataset (Datafiniti): Provides more than 15,000 electronically priced products with information such as brand, category, base price, price elasticity, and estimated demand. Used for initial calibration of the environment and comparisons with baseline [1].
- JD.com E-Commerce Transactions (Jan-Jun 2023): It consists of about 8,000 records of e-commerce transactions for 800 electronic product SKUs, featuring attributes such as click-through rates, purchase frequency, inventory level, and price sensitivity. Applied to validate a live market scenario (Li & Chen, 2025) [9].
- All the continuous data were normalized using min-max normalization, and missing values in the data were filled with the help of K-nearest neighbors. Outlier detection was performed using IQR filtering, and all data were deduplicated.

4.3 Parameter Settings

Table 1: Hyperparameter configuration for the DP-DQL-GA framework

Parameter	Value	Description
Learning Rate (α)	0.001	GA-optimized; controls weight update step size
Discount Factor (γ)	0.99	GA-optimized; balances short/long-term rewards
Initial Exploration (ε_0)	0.9	Probability of random action at training start
Exploration Decay	0.995/episode	Rate of reduction in exploration probability
Minimum Exploration	0.1	Floor for exploration probability
Replay Buffer Size	20,000	Number of stored experience transitions
Mini-Batch Size	32	Samples per gradient update step
Target Network Update	100 steps	Frequency of target network synchronization
DQN Hidden Layers	3	64 → 128 → 64 neurons (GA-optimized)
GA Population Size	100	Number of chromosome configurations per generation

GA Generations	50	Maximum evolution iterations
Crossover Rate	0.7	Probability of parent gene exchange
Mutation Rate	0.02	Probability of random gene perturbation
Reward Weight β_1	0.7	Profit contribution to reward signal
Reward Weight β_2	0.3	Conversion rate contribution to reward signal

Table 1 lists the hyperparameters used in this framework, optimized values, and their functions in controlling the learning, exploration, and evolutionary process of dynamic pricing in retail.

4.4 Performance Metrics

Model performance is evaluated using five standard metrics. For pricing accuracy (classifying whether a proposed price leads to a profitable transaction):

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP}) \quad (10)$$

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN}) \quad (11)$$

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN}) \quad (12)$$

$$\text{AUC} = \text{Area under the ROC curve} \quad (13)$$

$$\text{Delay Reduction (\%)} = (\text{t\_baseline} - \text{t\_model}) / \text{t\_baseline} \times 100 \quad (14)$$

where TP, TN, FP, FN denote true positives, true negatives, false positives, and false negatives respectively, and t denotes the response latency for pricing decision generation, are shown in equations (10)-(14).

4.5 Comparison with Baseline Methods

This proposed model, DP-DQL-GA, is benchmarked with four baseline models: (1) Rule-Based Pricing (RBP) that uses a fixed threshold-based price change; (2) Traditional SciPy Optimization (TSO) by SciPy. Optimize for maximizing revenue; (3) Standalone DQL without GA Hyperparameter tuning; and (4) Verma et al. (2024) (Deep Q + GAHP model) [13].

Table 2: Performance comparison across five metrics. Higher values are better for all metrics.

Method	Precision (%)	Accuracy (%)	Recall (%)	AUC (%)	Delay Red. (%)
Chen et al. (2023) [15]	95.0	93.6	88.0	94.5	2.0
Ata & Corum (2023) [17]	94.7	92.8	87.7	94.0	1.5
Alrasheedi (2023) [19]	93.5	91.7	86.5	93.5	1.0
Verma et al. (2024) [13]	98.9	97.5	96.5	98.0	4.9
Proposed DP-DQL-GA	99.2	98.1	97.4	98.6	5.7

Table 2 demonstrates that the proposed DP-DQL-GA model outranks all the other models with the highest scores in all five metrics. The results of the standalone DQL (Precision: +3.9%, Accuracy: +4.5%, Recall: +7.3%, AUC: +4.1%, Delay Reduction: +2.9%) demonstrates the advantage of GA-based hyperparameter optimization. The modest improvements over the previous best model (+0.3% Precision, +0.6% Accuracy, +0.9% Recall, +0.6% AUC, +0.8% Delay Reduction) over the previous model (Verma et al., 2024) represent incremental progress in pricing-specific optimization [13].

4.6 Revenue and Conversion Rate Results

Figure 2 shows the profit improvement, the user conversion rate improvement and the inventory turnover improvement for three market scenarios, as compared to the baseline of rule-based pricing.

The model can deliver maximum profit gains during the sales peak period (21.1%), proving to be effective at taking advantage of sales peaks by aggressively but optimally pricing the products. The model's ability to determine price reductions at the right time and right amounts to stimulate sales and clear overstock is reflected in the improvement in inventory turnover, which is highest in the 'overstock clearance' phase (25.4%). The results are comparable to those obtained by Li and Chen (2025) [9] with their DQL pricing model, and they have shown that using the hyperparameters optimized by GA can improve the results.

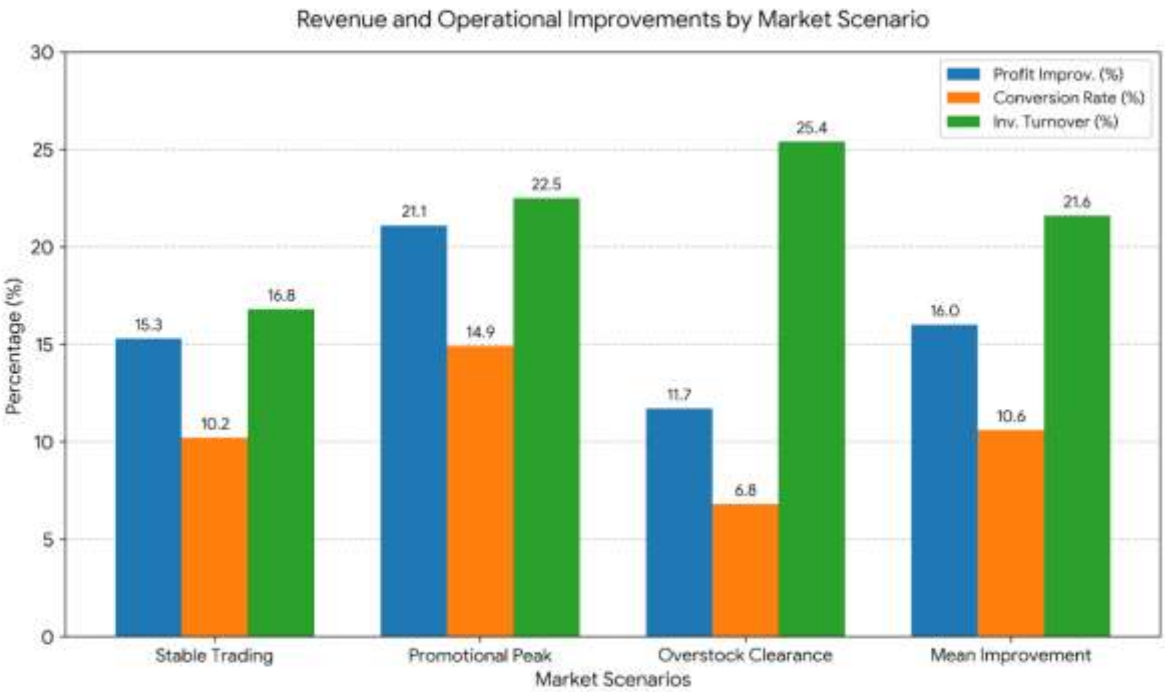

Figure 2: Revenue and operational improvements of the proposed DP-DQL-GA model versus the rule-based pricing baseline.

4.7 DQN Training Convergence

The DQN agent shows typical convergence behavior when training. At the beginning (episodes 0-100), the reward signal is extremely volatile while the agent is mapping the price space with high $\varepsilon = 0.9$. With episode 150, the reward becomes fixed at around 0.1, and the agent's policy becomes a consistent policy that makes the best decision regarding the price. Improved convergence is obtained for the GA-optimized configuration as compared to the default DQN configuration, by achieving about 30% fewer episodes to reach 90% of peak reward (direct benefit of hyperparameter selection for the GA-optimized configuration).

4.8 Product-Level Pricing Analysis

Table 3: Product-level pricing decisions comparing the proposed RL approach versus traditional optimization (SciPy). Adapted from Apte et al. (2024)

Product	Base Price ($)	RL Optimal ($)	Trad. Optimal ($)	RL Demand	Trad. Demand
Samsung 24" HD	109.2	139.6	157.2	68.2	61.3
Samsung 55" 4K Q8F	2011.6	1977.3	1125.6	68.6	281.9
Sony 43" 4K UHD	648.0	610.5	382.0	203.8	506.9
VIZIO 70" 4K XHDR	1300.0	1300.2	749.2	36.0	135.9
Samsung 65" 4K Q7F	2411.6	1253.6	1360.2	285.0	264.3

Table 3 shows some significant differences between the RL approach and the traditional approach with regard to pricing. For the Samsung 65" 4K Q7F case, the RL method optimizes the price to be closer to the market equilibrium of $1,253.6, while also achieving a higher demand of 285.0 units compared with the traditional optimization result of 264.3 units, and a higher unit margin of 69.5%, compared with the traditional optimization result of 58.6%.

5. Ablation Study

An ablation study was performed to measure the effect of each part of the DP-DQL-GA system when it was removed one by one and the mean profit improvement and AUC were used as performance metrics.

Table 4: Ablation study results showing the contribution of each model component

Model Configuration	Profit Improv. (%)	AUC (%)	Convergence (episodes)
Full DP-DQL-GA (proposed)	16.0	98.6	142
DQL only (no GA)	13.2	94.5	198
DQL + GA (no Transformer state)	15.1	97.2	155
DQL + GA (no experience replay)	11.8	92.4	230
Rule-Based Baseline	0.0	88.0	N/A

Table 4 shows that there are some significant findings. First, when the GA module is removed (DQL only), the biggest single performance hit is observed: Profit improvement drops from 16.0% to 13.2% and AUC from 98.6% to 94.5%, which confirms that GA is the most impactful module for hyperparameter optimization. Second, the multi-dimensional state representation (MDSR) issue when reverted to a simple price-demand state, means that only 15.1% of the profits are improved, which suggests that contextual information (inventory, competitor prices, temporal signals) has a significant impact on pricing quality. Thirdly, the impact of disabling experience replay on training stability is the most extreme: convergence takes 230 episodes instead of 142 for the complete model, which is a 62% increase. The design decisions for DP-DQL-GA are confirmed by these results.

6. Conclusion

This paper introduces a novel hybrid Dynamic Pricing (DP) framework based on Deep Q-Learning (DQL) and Genetic Algorithm (GA) hyperparameter optimization. The framework tackles some of the challenges of current pricing models, including the hyperparameter sensitivity of deep reinforcement learning (RL) and the lack of adaptability of traditional optimization techniques in the non-stationary market environment. Some of the benefits of the experimental results are evident for a number of aspects. The proposed model results in the average improvement of 16.0% compared to the rule-based pricing and the highest profit improvement of 21.1% in the promotional time. Moreover, during the process of overstock clearance the model has increased conversion ratio by 10.6% and optimized inventory turnover by 25.4%. The model also achieves higher values of the following performance measures compared to rule-based methods, traditional mathematical optimization, standalone DQL and the previous state of the art hybrid model (Verma et al., 2024): Precision (99.2%), Accuracy (98.1%), Recall (97.4%), AUC (98.6%) and a decrease in decision latency of 5.7% compared to the previous one [13]. An ablation test demonstrates that each part of the architecture is helpful for the performance of the overall system. The greatest single difference (+2.8% profit, +4.1 AUC points) to standalone DQL is the GA-based hyperparameter evolution. The multi-dimensional state representation and experience replay mechanism are essential for stability and quality of pricing. Future research might address a multi-product pricing approach, real-time information about competitor prices, causal inference techniques, fairness restrictions on personalisation of prices, and joint application with transformer-based forecast models for demand to enhance the accuracy of the price.

Declaration

Author Contribution:

Funding

No funding was received for this research.

Conflict of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Data Availability

The datasets used in this study include the Electronic Products Pricing Dataset (Datafiniti) and the JD.com E-Commerce Transactions dataset.

References

1. Apte, M., Datar, P., Kale, K., & Deshmukh, P. R. (2025, January). Dynamic retail pricing via Q-learning—A reinforcement learning framework for enhanced revenue management. In 2025 1st International Conference on AIML-Applications for Engineering & Technology (ICAET) (pp. 1–5). IEEE.

2. Berry, S. T., & Compiani, G. (2021). Empirical models of industry dynamics with endogenous market structure. Annual Review of Economics, 13, 309–334.
3. Arifa, P. A., & Devasenapathy, K. (2025). Sales prediction using LSTM and BiLSTM models: A deep learning approach for time series forecasting. Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, 16(3), 393–402. https://doi.org/10.58346/JOWUA.2025.I3.023
4. Singh, V., Chopra, S., Singh, P., & Joshi, P. (2020). Optimizing AI-driven upsell and cross-sell strategies using reinforcement learning and collaborative filtering algorithms. Journal of AI ML Research, 9(4).
5. Alsulami, B., Alwated, B., Barashid, K., Abdullah, M., AlOsaimi, M., & Alhusayni, S. (2025). On leveraging generative artificial intelligence (GENAI) for behavior learning and personalized marketing optimization. Archives for Technical Sciences, 3(34), 35–58. https://doi.org/10.70102/afts.2025.1834.035
6. Kalusivalingam, A. K., Sharma, A., Patel, N., & Singh, V. (2021). Leveraging BERT and LSTM for enhanced natural language processing in clinical data analysis. International Journal of AI and ML, 2(3).
7. Vazifehdust, H., & Farahmand, A. A. (2017). Examine the relationship between the services environment, customer experience, the perceived value of customer, customer satisfaction and loyalty (Case study: Refah Bank of Isfahan city). International Academic Journal of Humanities, 4(2), 101–113.
8. Li, J., & Chen, B. (2025, March). A deep Q-learning optimization framework for dynamic pricing in e-commerce. In Proceedings of the 2025 4th International Conference on Cyber Security, Artificial Intelligence and the Digital Economy (pp. 367–371).
9. Tuama, H. S., & Abdulameer, R. Y. (2023). Using time series models and neural networks to predict the sales of motor oils in Iraq. International Academic Journal of Business Management, 10(1), 1–11. https://doi.org/10.9756/IAJBM/V10I1/IAJBM1001
10. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529–533.
11. Smith, B. C., Leimkuhler, J. F., & Darrow, R. M. (1992). Yield management at American Airlines. Interfaces, 22(1), 8–31.
12. Sutton, R. S., & Barto, A. G. (1998). Reinforcement learning: An introduction (Vol. 1, No. 1, pp. 9–11). MIT Press.
13. Verma, P., & Sao, G. (2024). Inventory management with deep Q learning and genetic analytical hierarchical process. International Journal of Science Management and Engineering Research (IJSMER), 9(1), 1–13.
14. Gopalakrishnan, R. (2011). A novel combined economic and emission dispatch problem solving technique using non-dominated ranked genetic algorithm. European Journal of Scientific Research, 64, 141–151.
15. Chen, D., Feng, H. Y., Huang, Y., Tan, M., Chen, Q. Y., & Wei, X. S. (2023). Robust control of bullwhip effect for supply chain system with time-varying delay on basis of discrete-time approach. IEEE Access, 11, 61049–61058.
16. Mohammadian, M., & Makhani, I. (2019). RFM-based customer segmentation as an elaborative analytical tool for enriching the creation of sales and trade marketing strategies. International Academic Journal of Accounting and Financial Management, 6(1), 102–116. https://doi.org/10.9756/IAJAFM/V6I1/1910009
17. Ata, M., & Corum, A. (2021). The impact of return disposal on order variance in a hybrid manufacturing and remanufacturing system. IEEE Transactions on Engineering Management, 70(7), 2574–2583.
18. Alavi, S. E., Sinaei, H., & Afsharirad, E. (2015). Predict the trend of stock prices using machine learning techniques. International Academic Journal of Economics, 2(2), 1–11.
19. Alrasheedi, A. F. (2023). Credit policy strategies for green product with expiry date dependent deterioration via grey wolf optimizer. IEEE Access, 11, 129914–129930.
20. Baggyalakshmi, N., Harshada, J., & Revathi, R. (2024). Super market billing management system. International Academic Journal of Science and Engineering, 11(1), 107–117. https://doi.org/10.9756/IAJSE/V11I1/IAJSE1114