Self-Evolving AI Systems: A Reinforcement Learning Framework for Autonomous Model Adaptation

1st Pankaj Kumar
Department of Technical Education
GPC, Patiala
Patiala, Punjab, India
singla.pankaj4@gmail.com

2nd Anjna Kumari
School of Engineering and Technology
CGC University
Mohali-140307, Punjab, India
anjna.j4147@cgcuniversity.in

3rd Vishav Pratap Singh
Department of CSE
UIE, Chandigarh University
Mohali-140413, Punjab, India
pratapvishav92@gmail.com

4th Nongmeikapam Thoiba Singh
Department of CSE
UIE, Chandigarh University
Mohali-140413, Punjab, India
nthoiba12@gmail.com

***Abstract*—Rapidly changing and becoming more complicated environments of the real world present ideal challenges to the traditional artificial intelligence systems which are normally stationary with regular retraining needed to sustain these performances. This paper presents a new paradigm of a self-evolving AI systems, which, through the reinforcement learning, uses autonomous adaptation of the models in changing contexts. The proposed solution will combine a reinforcement learning agent with an adaptive evolution solution that can alter model parameters, models and models in real time according to environmental feedback. The system will optimize its behavior through reward optimization, thereby decreasing human dependencies on manual operations and annotated datasets, by continually refining its behavior. Mechanisms of dynamic policy updates, exploration-exploitation ratio and performance monitoring are included in the framework to maintain stability and efficiency in the course of adaptation. Moreover, the model will be used to counter concept drift through the identification and reaction to the alteration in the data. Adaptability, learning efficiency, and robustness are tested though experimental evaluation (through simulated and real-time datasets). The findings reveal that the suggested self-evolving system shows superior ability to adapt, precision and longevity in performance as compared to the traditional stationary models. The research results in the further development of smart systems capable of learning during its lifetime, and autonomously making its decisions.**

***Index Terms*—Self-Evolving AI, Model Adaptation, Lifelong Learning, Adaptive Intelligence, Concept Drift, Meta-Learning, Dynamic Optimization, Policy Learning.**

I. Introduction

The artificial intelligence has gained a staggering level of success in a myriad of applications, including image recognition, natural language processors, health analytics, and autonomous systems. Most of these successes, however, are founded on the models trained at large scale on controlled environments and then deployed to the real life environment with minimal changes. As much as such models might perform sufficiently at the onset, once time elapses with a shift in data population on which the model is based, then they tend to be exhausted[1-3].It is commonly known as the concept drift effect and is an intrinsic shortcoming of the traditional AI models, as they cannot evolve continuously unless being tinkered with by humans. Patterns in data change quickly in dynamic systems like cybersecurity and financial markets, and in smart transportation systems [4]. The threats, behaving of users and system state are affecting and so dynamic models are becoming more effective. Conventional methods deal with this issue by retraining the model periodically by gathering new information, labeling it and recomputing the model offline. This process is, though not fully successful, time-consuming, resource-intensive and occasionally incapable of keeping up with the real-time changes. Consequently, there is an increased demand on AI systems capable of evolving independently and continue to perform without the need to update the data regularly and manually. A promising way to overcome this challenge is through reinforcement learning [5]. Contrary to the use of a labeled data in the supervised learning, the reinforcement learning allows an agent to learn through interaction with his or her environment by being provided with feedback in form of a reward or penalty. This trial and error learning process enables the agent to learn how to play optimum strategies with time irrespective of changing situations. A reinforcement learning agent that is guided by the consistent revision of its policy in response to environmental feedback, can demonstrate adaptive behavior that is dynamic enough to be suited in dynamic situations. Based on this potential, there has been the idea of self-evolving AI systems as one of the solutions to long-term flexibility [6]. The self-evolving system is such that it adjusts the parameters, as well as the system structure and the strategies used in decision making, with changes in the presented information. These systems are not a kind of online education only but involve the opportunity of endless improvement, discovery of new methods and adaptation to unforeseen changes. This type of flexibility is required in performance in order to achieve high and robust performance in the complex environments. The paper proposes a framework of the autonomous model adaptation using reinforcement learning in the context of the self-evolving AI systems. The design combines an evolution mechanism with an adaptive learning agent that allows updating model parameters and policies in real-time.

II. Problem Statements

Traditionally single artificial intelligence models are designed in static situation, and are not efficient in sustaining performance under constantly varying data trends [1-4]. As the real-world systems change, the models suffer a decline in performance in response to the change in data distribution, which is often called concept drift. Current solutions are highly dependent on the periodic retraining which is time consuming, resource intensive and usually lack the capacity to react to the real time changes.

1) Traditional artificial intelligence models are primarily designed for static environments and often fail to maintain performance when data distributions change over time.
2) Current solutions are based on retraining processes which are computationally heavy, time-consuming and inapplicable to real time scenarios.
3) The existing systems do not have the capability of autonomous adaptation meaning that parameters can only be changed by people and the models enhanced to behave better.
4) The presence of concept drift in real-world scenarios, such as healthcare and cybersecurity, reduces the accuracy and reliability of deployed models.
5) There is a need for a unified framework that supports continuous learning, structural adaptation, and real-time decision-making within a single intelligent system.

III. Related Work

Adaptive learning methods have also been studied recently to overcome the shortcomings of the fixed artificial intelligence models in a dynamic environment [1-3]. Reinforcement learning has gained much application to allow systems to learn through interactivity and enhance decision-making as time goes by. Initial efforts were on policy optimization and value-based approaches, which showed a high level of performance in a controlled environment but not in systems with dynamically changing data [6]. Learning how to learn on the meta-learning provided the ability of models to generalize between tasks with only slight modifications. Also, there have been studies on continuous and lifelong learning, which have been sought to minimize forgetting and add new information progressively. Although these improvements are made, most existing techniques are based on predefined structures and are not provided with structural evolution mechanisms [9]. Also concept drift is a problem independent of the solution as most solutions need to be manually or periodically retrained.

IV. Literature Review

The growth in the use of artificial intelligence has resulted in the creation of powerful models in different fields like vision, language processing, and decision making systems. The majority of such models are however including the assumption of fixed data distributions which reduces their usefulness in realistic settings where processes change as time progresses. This weakness has inspired a lot of research into adaptive learning methods that allow the models to adapt to new data patterns and preserve performance without retraining entirely by **Kundu et al.** [1] A key reinforcing behavior building method to construct adaptive systems has been incorporated through reinforcement learning. It enables an agent to optimize cumulative rewards by fiddling with its immediate surroundings in order to discover the best things to do. The initial works of this area developed the theoretical basis of value functions, policy learning, exploration strategies. Methods like Q-learning and policy gradient methods shown that it was possible to solve decision-making problems of complex type by **Bhukya et al.** [2] Subsequently, with the emergence of deep reinforcement learning in later applications, neural networks were used together with a reinforcement learning algorithm to support a high-dimensional state space. Whereas these methods performed remarkably well in controlled settings, which included games, and simulations, they have limited adaptability to real-time dynamic environments. In order to handle adaptability, scholars proposed the concept of meta-learning that aims at allowing models to obtain new tasks with a higher level of efficiency through the use of previous knowledge. The aim of meta-learning methods is to minimize training time and enhance generalization by fine-tuning models to enable quick adaptation by **Parashar et al.** [3]. Such approaches have demonstrated potential in few-shot learning, where there is scanty data. Majority of meta-learning frameworks however, work within predefined structures, and lack in the mechanisms of continuous evolution after deployment. Consequently, they are limited in their use in long term and continually evolving environments. Continual or lifelong learning is another significant direction and is focused on the capability of models to learn step-by-step without forgetting the information learned in the past. Regularization-based approaches and memory replay systems, and dynamic architecture are some of the techniques that are suggested as ways to eliminate catastrophic forgetting by **Fang et al.** [4]. The methods enable models to combine new information without forgetting previous learning. Along with these advances, there are still issues in maintaining stability and plasticity in scenarios where the data to operate with changes quickly or randomly. Moreover, lots of existing continual learning systems are yet to be tightly tuned, and they do not truly work independently. Concept drift has continued to cause studies which have led to the necessity of adaptive systems. The change in statistical characteristics of input data with time is called concept drift, and can seriously undermine the performance of a model. Many methods have been suggested to detect and adapt to change such as window-based, ensemble learning and drift-aware algorithms by **Narayana et al. [6]**. Though such solutions can detect and react to changes, they usually rely on predetermined limits or human intervention, so they cannot react to changes in real-time. The concept of self-adaptive and self-evolving systems has more recently been in the limelight. These systems are supposed to modify their internal parameters, architectures, or learning strategies, based on environmental feedback.

V. Proposed Framework: Self-Evolving AI System

The proposed framework is developed to support continuous learning and independent adaptation in environments where data and conditions change over time **[1-3]**. It combines reinforcement learning with adaptive and evolutionary components so that the system can improve its performance without requiring repeated manual updates. The architecture is composed of several interconnected modules that work together to achieve self-evolution.

*1) **Environment Interface**:* The environment interface collects data from external sources such as sensors, user inputs, or system logs **[3-5]**. This module carries out simple preprocessing and transforms raw data in to an organized format that can be understood by the learning parts of the system.

*2) **Reinforcement Learning Agent**:* The decision-making unit is the reinforcement learning agent. It monitors the present-day environment and chooses other actions according to its existing policy functions **[10]**. Once an action is performed, the feedback is provided in such a way that the action is reacted to by a reward, which is utilized in making a better decision at a later stage. Learning obeying the Q-learning update rule is as follows:

$$Q(s,a) = Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right] \quad (1)$$

In this equation, s denotes the current state, a represents the chosen action, r is the received reward, γ is the discount factor that controls future rewards, and α is the learning rate. This update mechanism enables the agent to refine its decisions over time based on experience.

*3) **Experience Replay Memory**:* This module contains past transactions between the environment and the agent. This module is the storage of the past transactions between the environment and the agent. The system can enhance efficiency of learning by reusing previous experiences during training and prevents instability when highly correlated data is used **[11]**.

*4) **Concept Drift Detector**:* The concept drift detector will be used to monitor the data coming in so that it can detect any changing patterns over a time period. —human—¿Concept drift detector will monitor the arriving data to detect changing patterns with time. A shift is sensed which activates adaptation mechanisms so that the system can still be effective **[12]**.

*5) **Meta-Adaptation Module**:* Meta-adaptation module manipulates the essential learning control parameters like, learning rate and exploration strategy learning control parameters that are manipulated by the meta-adaptation module include learning rate and exploration strategy. This enables the system to adapt better to change of the environment without necessarily re-initiating the learning process.**[13]**.

*6) **Evolution Engine**:* The parameter updates are the role of the engine of evolution which advances the model even further. It allows a structural change, e.g. reconfiguring layers or optimization of configurations to achieve a better performance of the whole system.

*7) **Updated AI Model**:* The new model incorporates all the learning and evolutionary changes. It is a more sophisticated form of the system that is able to make decisions in varying conditions that are better.

*8) **Feedback Loop**:* Alves and Aronsson have developed a permanent feedback system that links all modules and provides the system to learn with new data and results. This loop will enable the model to adapt, get better and alter with time without any outside help.

The framework unites the concept of reinforcement learning with that of meta-adaptation and evolution to ensure the model is able to enhance its performance as time goes by without the need to be trained by human beings. As illustrated in **Fig.** 1, the system begins with an environment interface that collects and preprocesses real-time data from external sources. This processed data is then fed into a reinforcement learning agent, which makes decisions based on the current state of the environment **[12]**. The agent uses a policy network to select optimal actions and evaluates outcomes using a reward mechanism.

VI. Experimental Setup and Implementation

To evaluate the effectiveness of the proposed self-evolving AI system, experiments are conducted in a dynamic environment with continuously changing data distributions. The system interacts with the environment in a sequential manner, where each observation is represented as a state s_t at time step t. The agent selects an action a_t and receives a reward r_t based on the quality of its decision.

The objective of the reinforcement learning agent is to maximize the expected cumulative reward, defined as:

$$R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \quad (2)$$

where γ is the discount factor that determines the importance of future rewards.

The action selection strategy follows an ϵ-greedy policy:

$$a_t = \begin{cases} \text{random action}, & \text{with probability } \epsilon \\ \arg\max_a Q(s_t, a), & \text{with probability } 1 - \epsilon \end{cases} \quad (3)$$

This approach ensures a balance between exploration and exploitation during training.

TABLE I
SUMMARY OF RELATED WORK ON SELF-EVOLVING AI SYSTEMS

S.No.	Author	Year	Method & Technology	Research Gap
1	Kundu, S. [1]	2021	Real-time self-evolving AI architectures for autonomous adaptation	Limited scalability and lack of integration with advanced learning frameworks
2	Bhukya et al. [2]	2025	AI-generated machine learning models with automated design	Insufficient focus on continuous real-time adaptation
3	Parashar, P. [3]	2025	Feedback-driven self-evolving AI workflows	Lacks structural evolution and long-term learning capability
4	Fang et al. [4]	2025	Survey of self-evolving AI agents and lifelong systems	Conceptual overview without practical implementation strategies
5	James et al. [5]	2025	Self-programming AI using evolutionary algorithms	Limited real-time adaptability and high computational cost
6	Narayana et al. [6]	2025	Autonomous real-time model evolution for streaming data	Limited evaluation on diverse real-world datasets
7	Gao et al. [7]	2025	Survey on self-evolving agents toward artificial super intelligence	Lacks unified framework for implementation
8	Chennamsetty, C. S. [8]	2024	Adaptive training pipelines with feedback loops	Does not address structural model evolution
9	Singh, A. [9]	2023	Edge-based self-evolving IoT systems	Limited generalization beyond IoT environments
10	Azmat & Raheem [10]	2024	AutoML for evolving machine learning models	Focuses on automation, not continuous adaptation
11	Nezami et al. [11]	2025	Self-evolving communication systems	Limited integration with reinforcement learning
12	Sapkota et al. [12]	2025	Evolution in large language model architectures	Lacks real-time adaptation mechanisms
13	Caulfield et al. [13]	2025	Liquid adaptive AI for continuous improvement	Theoretical framework with limited empirical validation
14	Lohani & Venkataraman [14]	2024	RL-based self-evolving transformer networks	Limited scalability in large dynamic environments
15	Pandey et al. [15]	2025	Self-learning agents for cybersecurity in IoT	Domain-specific, lacks general-purpose adaptability
16	Li et al. [16]	2023	Self-evolving autonomous driving systems	Focused on specific application domain
17	Gheibi & Weyns [17]	2024	Lifelong self-adaptation for drift handling	Complexity in implementation and high resource requirements
18	Tang et al. [18]	2025	Self-evolving edge learning for AI networking	Limited exploration of model evolution strategies
19	Uzair et al. [19]	2025	Policy-governed self-evolving AI architectures	Lacks dynamic learning optimization techniques
20	Zhao et al. [20]	2025	Self-evolving agentic AI for wireless networks	Early-stage research with limited experimental validation

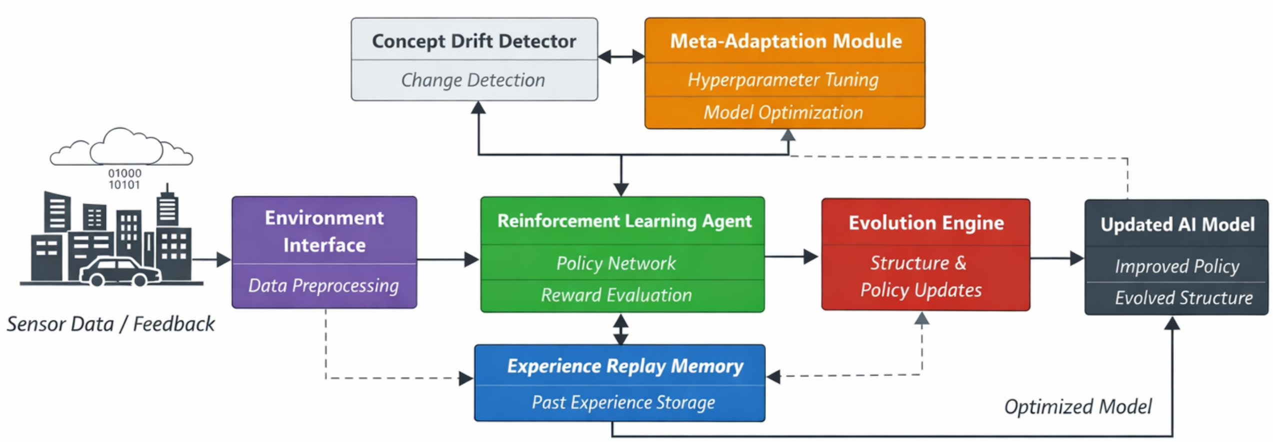

Fig. 1. Architecture of the Proposed Self-Evolving AI System

To simulate concept drift, the data distribution is modified over time. The drift detection mechanism is based on monitoring changes in statistical properties of the data. A simple drift condition can be expressed as:

$$|\mu_t - \mu_{t-1}| > \delta \tag{4}$$

where μ_t represents the mean of the current data window and δ is a predefined threshold.

A. Implementation Details

The framework is implemented using Python with support for deep learning and reinforcement learning libraries. The learning process is guided by the Q-value update rule:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right] \tag{5}$$

To improve training stability, experience replay is used. The loss function for training a Deep Q-Network (DQN) is defined as:

$$L(\theta) = \mathbb{E}\left[\left(r_t + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-) - Q(s_t, a_t; \theta) \right)^2 \right] \tag{6}$$

where θ represents the current network parameters and θ^- denotes the target network parameters.

The meta-adaptation module dynamically adjusts the learning rate using:

$$\alpha_t = \alpha_0 \times \frac{1}{1 + \lambda t} \tag{7}$$

where α_0 is the initial learning rate and λ controls the decay rate.

The evolution engine modifies the model configuration based on performance improvement. A simple evolution condition can be defined as:

$$\Delta P = P_{new} - P_{old} \tag{8}$$

where P represents the performance metric (e.g., accuracy or reward). If $\Delta P < 0$, the system triggers structural adaptation.

The overall application makes the system forever learn, adapt and evolve through incorporation of reinforcement learning with adaptive control and model optimization methods.

VII. Self-Evolving AI Algorithm

The suggested algorithm outlines the process of learning and adjusting of the self-evolving AI system. It combines reinforcement learning, detecting concept drift and model evolution into a single model.

Initialize Q-network parameters θ
Initialize target network parameters θ^-
Initialize replay memory $\mathcal{D}$
Initialize learning rate α, discount factor γ, exploration rate ϵ
Observe initial state s_t
while training is not terminated **do**
  Select action a_t using ϵ-greedy policy
  Execute action a_t in environment
  Observe reward r_t and next state s_{t+1}
  Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$
  Sample mini-batch from $\mathcal{D}$
  Compute target:

$$y_t = r_t + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-)$$

  Update network by minimizing loss:

$$L = (y_t - Q(s_t, a_t; \theta))^2$$

  Update state: $s_t \leftarrow s_{t+1}$
  Detect concept drift:
  if drift detected **then**
    Adjust learning parameters (α, ϵ)
  end if
  Evaluate performance improvement ΔP
  if $\Delta P < 0$ **then**
    Trigger evolution engine
    Modify model structure or hyperparameters
  end if
  Update target network periodically:

$$\theta^- \leftarrow \theta$$

end while
Return: Optimized and self-evolved model

VIII. Results and Discussion

The proposed self-evolving AI system is evaluated in a dynamic environment where data distributions change over time [12]. The performance is assessed using cumulative reward, accuracy, and learning stability. The cumulative reward is defined as:

$$R = \sum_{t=0}^{T} \gamma^t r_t \tag{9}$$

where r_t denotes the reward at time step t, γ is the discount factor, and T is the total number of iterations. An increasing trend in R indicates that the agent is improving its decision-making policy over time.

A. Accuracy Analysis

The classification or decision accuracy of the system is measured to evaluate its effectiveness under changing conditions. Accuracy is computed as:

$$\text{Accuracy} = \frac{N_{correct}}{N_{total}} \tag{10}$$

where $N_{correct}$ is the number of correct predictions and N_{total} is the total number of predictions [11]. The outcomes indicate that the performance of traditional static models decreases upon variation in the data distribution whereas the proposed system has constant performance with continuous adaptation.

B. Concept Drift Adaptation

To analyze adaptability, concept drift is introduced into the data stream. The drift magnitude is estimated as:

$$D = |\mu_t - \mu_{t-1}| \tag{11}$$

where μ_t and μ_{t-1} represent the mean values of consecutive data windows. When D exceeds a predefined threshold, the system activates adaptation mechanisms such as parameter tuning and policy updates. This enables the model to respond effectively to environmental changes.

C. Learning Stability

The stability of the learning process is evaluated using the temporal difference (TD) error:

$$\delta = r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \tag{12}$$

A gradual reduction in δ indicates that the model is converging toward an optimal policy. The use of experience replay helps in smoothing the learning process by reducing variance in updates.

D. Evolution Performance

The effectiveness of the evolution module is measured by tracking performance improvement over time:

$$\Delta P = P_t - P_{t-1} \tag{13}$$

where P_t represents the performance metric at time t. If ΔP becomes negative, the system triggers structural or parametric updates to enhance performance [12]. This mechanism ensures long-term adaptability and robustness.

TABLE II
PERFORMANCE COMPARISON BETWEEN STATIC AND SELF-EVOLVING MODELS

Metric	Static Model	Proposed Model
Accuracy (%)	78.5	91.2
Cumulative Reward	245	372
Adaptation Speed (iterations)	150	85
Stability (Variance)	High	Low

The results presented in Table II compose the results of the conventional non-dynamical model and the proposed self-evolution AI system. As it can be seen, the accuracy of the proposed model can be significantly increased, which means that this model will be able to make decisions in the changing environment much better [14]. Also, the cumulative reward is larger, which indicates an increased policy optimization and efficiency in learning with time. The proposed system also takes a shorter time to adapt to the environmental changes, since the system does not go through as many iterations before adapting. Moreover, the reduced consistency in terms of performance indicates the consistency of the learning process showing that the system can be used to achieve consistent results even in varied circumstances. Table III plots the behavior of the system to varying levels of concept drift. The higher the level of drift, the higher the time of detection will be because there will be more spread in the distribution of data [13]. Nevertheless, the proposed system is effective in restoring accuracy post-adaptation even in scenarios where the drift is large. This shows that the drift detection process and adaptation processes can identify changes and change the model accordingly. This capability to resume the performance shortly after drifting indicates the stability of the framework to deal with real-time data changes. The values in Table IV The numbers represented in the figure, the learning evolution in successive episodes. The cumulative reward is accumulating gradually and this implies that the agent is enhancing its policy by being able to interact with the environment on a regular basis [13]. Meanwhile, there is a reduced temporal difference (TD) error, indicating better predictability of future rewards. This trend establishes that the learning process is stable and is heading towards an optimum solution. This rising rewards and reducing errors illustrates the efficiency of reinforcement learning system deployed in the system. Table V emphasizes the role of the evolution module in the system performance. The baseline level of accuracy is first achieved with the model. Performance after applying parameter updates is improved, which suggests that fine-tuning generates better performance [14]. The enhancement would be further noticed

TABLE III
CONCEPT DRIFT ADAPTATION PERFORMANCE

Drift Level	Detection Time	Accuracy Before	Accuracy After
Low	10	90.5	91.0
Medium	18	88.2	90.3
High	25	80.1	89.7

TABLE IV
LEARNING CONVERGENCE OVER TIME

Episodes	Cumulative Reward	TD Error
50	120	0.51
100	210	0.32
150	300	0.28
200	372	0.22

TABLE V
IMPACT OF EVOLUTION MODULE ON PERFORMANCE

Stage	Accuracy (%)	Performance Gain
Before Evolution	85.3	–
After Parameter Update	88.7	+3.4
After Structural Update	91.2	+2.5

once structural changes have been applied and this indicates that contrary to the assumption that learning ability does not depend on the structure, it depends on the modification of the architecture. These findings have indicated that parameter-level and structural level adaptations are significant to the system effectiveness in general.

TABLE VI
DYNAMIC HYPERPARAMETER ADJUSTMENT

Iteration	Learning Rate (α)	Exploration Rate (ϵ)
0	0.10	1.00
50	0.07	0.70
100	0.05	0.40
150	0.03	0.20

Table VI presents the dynamic adaptation of important hyperparameters in training. Learning rate decreases progressively, with the ability of the model doing smaller and more specific updates over time during training. The same can be said about the exploration rate, which decreases with time, which allows the agent to transition to between exploration and exploitation [13]. This regulated compromise is beneficial to the balance of learning or stability. The gradual transition guarantees that the model would explore properly during the initial stages and refine towards the best decisions subsequent stages so as to help achieve better overall performance.

IX. Comparative Performance Analysis

Fig. 2 gives a comparative study between the proposed self-evolving AI system and a traditional stationary system in terms of various performance measures such as cumulative reward, accuracy, temporal difference (TD) error, and drift detection time.

A. Cumulative Reward Analysis

The cumulative reward reflects the learning efficiency of the system and is defined as:

$$R = \sum_{t=0}^{T} \gamma^t r_t \tag{14}$$

where r_t is the reward at time step t and γ is the discount factor. The proposed model shows a continuous increase in reward, indicating effective policy learning, whereas the static model reaches saturation early due to limited adaptability.

B. Accuracy Evaluation

The accuracy of the model is computed as:

$$\text{Accuracy} = \frac{N_{correct}}{N_{total}} \tag{15}$$

The accuracy of the proposed system has a higher likelihood of keeping its accuracy over time since it is in a continuous process of adjusting to variations in data trends. The opposite is the case in the static model because when subjected to changing environments, it demonstrates deterioration in performance.

C. Learning Stability (TD Error)

The temporal difference (TD) error is used to measure learning stability:

$$\delta = r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \tag{16}$$

When the TD error is decreasing, then the learning process converges. The error shows a gradual decrease in the proposed model indicating stable behavior during learning, whereas the error shows more steady values in the static model.

D. Drift Detection Efficiency

The ability to detect changes in data distribution is evaluated using drift detection time. A simple drift condition is given by:

$$D = |\mu_t - \mu_{t-1}| \tag{17}$$

where μ_t and μ_{t-1} represent statistical properties of consecutive data windows. The proposed system detects drift earlier, enabling faster adaptation, whereas the static model responds more slowly.

X. Discussion & Future Work

The integrated results confirm that the suggested self-evolving AI system attains high-performance in all the measured metrics. Continuous improvement is possible through the combination of reinforcement learning, parameter tuning as an adaptive process and structural evolution [15]. Not only the system learns the best policies but also it adjusts effectively to the changes in the environment and is therefore robust and reliable in comparison to the traditional non dynamic systems. The findings indicate that the suggested self-evolving AI system works well in dynamic conditions, in a consistent manner. In contrast to traditional models that deteriorate due to altering data, the proposed framework is resistant to changes, as it involves continuous learning and adaptation [17]. Reinforcement learning, concept drift detection, and evolutionary updates combined enable the system to enhance both the short-term and the long-term performance. Such results show promise of self-evolving AI systems to be used in practice in the real world where the conditions are unpredictable and constantly changing.

XI. Conclusion

The current paper introduces a self-developing AI architecture relying on reinforcement learning to adapt autonomously a model in a dynamic setting. The proposed system solves the shortcomings of the traditional static models by providing non-stop learning, real-time decision-making, and automatic adaptation of evolving data trends. The framework enables the model to evolve its parameters and structure automatically without having to manually re-test every time operating on new data through the combination of reinforcement learning and detecting concept drift, meta-adaptation, and an evolution mechanism, it results in a framework where the model provides both automated parameter and structure updates without necessitating a full re-test. The results of the experiment prove

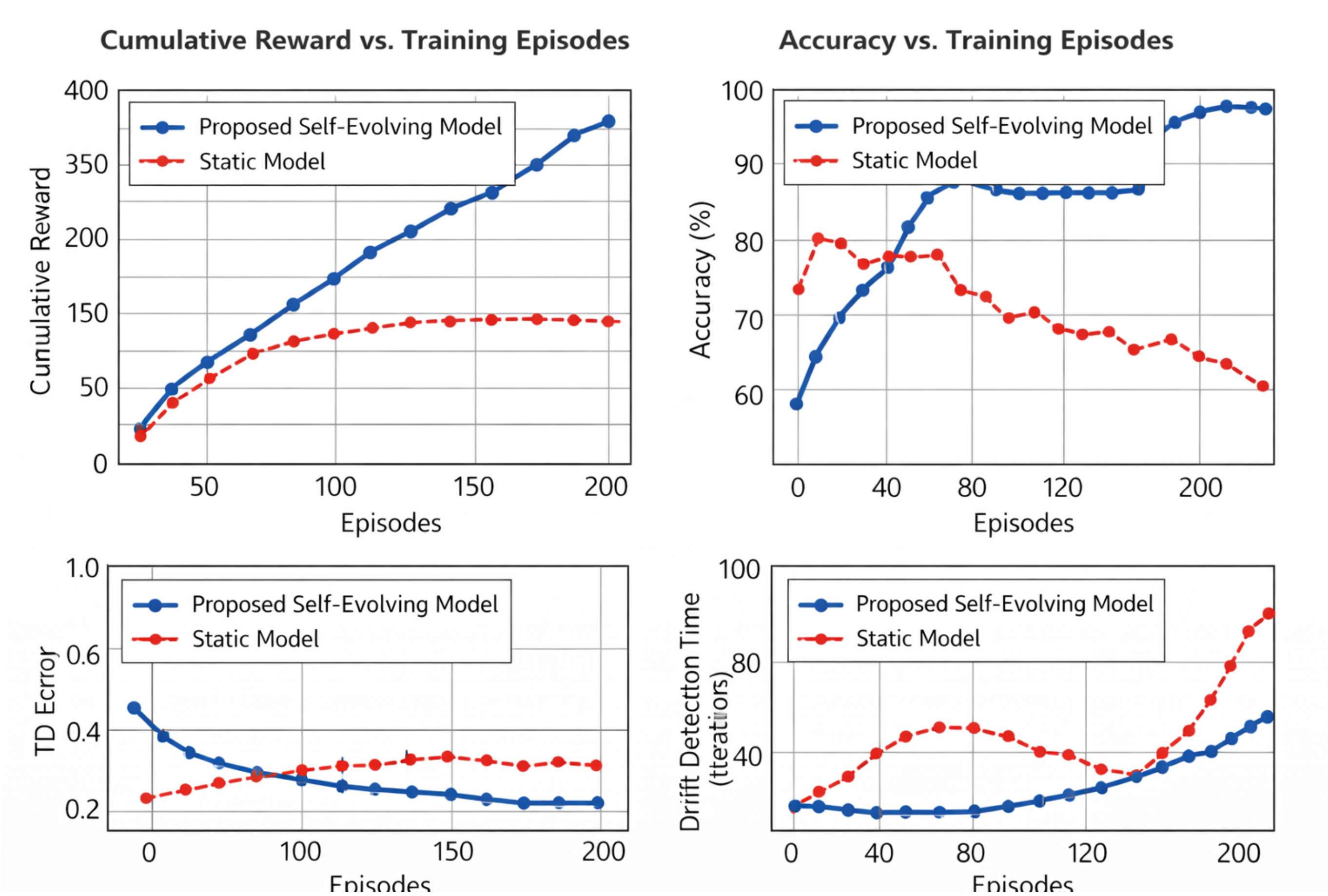

Fig. 2. Comparison of the performance between proposed self-evolving model and the prior model in various metrics.

that the experimental approach generates better performance, when the criteria are accuracy, cumulative reward and stability in learning. The system has the capability of dealing with real world uncertainties as it successfully remains consistent in its performance despite changes in the data distribution. Besides, experience replay and adaptive learning methods are used, which helps to achieve stable convergence and effective learning. All in all, the proposed self-evolving AI system is a robust and scalable solution in the applications where the constant adaptation is needed. It is a big step towards the creation of intelligent systems that can operate in a long-term autonomous mode. Such systems can be more practical with applications available to a greater number of people and can be further improved in the future with more adaptability, interpretability, and real-world deployment.

References

[1] Kundu, S. (2021). Self-Evolving AI Architectures: Real-Time Autonomous Adaptation for Smarter Systems. INTERNATIONAL JOURNAL, 2(6), 458-461.

[2] Bhukya, S. N., Babu, P. R., Annapoorna, E., Reddy, A. S., Anitha, K., Mathew, J. (2025, April). AI-Generated Machine Learning Models: Towards an Era of Self-Evolving Intelligence. In 2025 International Conference on Metaverse and Current Trends in Computing (ICMCTC) (pp. 1-4). IEEE.

[3] Parashar, P. (2025). Self-Evolving AI Workflows: A Formalized Feedback Model for Autonomous Optimization. International Journal of Emerging Research in Engineering and Technology, 6(3), 34-40.

[4] Fang, J., Peng, Y., Zhang, X., Wang, Y., Yi, X., Zhang, G., ... Meng, Z. (2025). A comprehensive survey of self-evolving ai agents: A new paradigm bridging foundation models and lifelong agentic systems. arXiv preprint arXiv:2508.07407.

[5] James, N. H., Ene, D. S., Lenu, G. F. (2025). Self-Programming Artificial Intelligence: Autonomous Learning and Evolutionary Algorithms. International Journal of Computer Science and Mathematical Theory, 11(5), 77-92.

[6] Narayana, V. A., AlJawahry, H. M., Sridivya, A., Rao, K. S., Pooncholai, S., Violet, S. (2025, April). Autonomous Real-Time Model Evolution: A Self-Evolving AI Framework for Dynamic Data Streams. In 2025 International Conference on Metaverse and Current Trends in Computing (ICMCTC) (pp. 1-5). IEEE.

[7] Gao, H. A., Geng, J., Hua, W., Hu, M., Juan, X., Liu, H., ... Wang, M. (2025). A Survey of Self-Evolving Agents: What, When, How, and Where to Evolve on the Path to Artificial Super Intelligence. arXiv preprint arXiv:2507.21046.

[8] Chennamsetty, C. S. (2024). Adaptive Model Training Pipelines: Real-Time Feedback Loops for Self-Evolving Systems. International Journal of Advanced Research in Computer Science Technology (IJARCST), 7(6), 11367-11373.

[9] Singh, A. (2023). Self-evolving IoT systems through edge-based autonomous learning. International Journal of Engineering Extended Technologies Research (IJEETR), 5(6), 7547-7555.

[10] Azmat, H., Raheem, A. (2024). Automated Machine Learning (AutoML): The Path to Self-Evolving Models. Euro Vantage journals of Artificial intelligence, 1(4), 75-81.

[11] Nezami, Z., Shah, S. D. A., Hafeez, M., Djemame, K., Zaidi, S. A.

R. (2025). From connectivity to autonomy: the dawn of self-evolving communication systems. Frontiers in Communications and Networks, 6, 1606493.

[12] Sapkota, R., Roumeliotis, K. I., Pokhrel, S., Karkee, M. (2025). From self-learning to self-evolving architectures in large language models: A short survey. Authorea Preprints.

[13] Caulfield, T. R., Islam, N. N., Chitale, R. (2025). Liquid Adaptive AI: A Theoretical Framework for Continuously Self-Improving Artificial Intelligence. AI, 6(8), 186.

[14] Lohani, M. C., Venkataraman, N. L. (2024). Self-Evolving Transformer Networks: A Reinforcement Learning-Based Approach for Continual Model Adaptation in Dynamic Environments. Available at SSRN 5142381.

[15] Pandey, A., Bepari, K., Tiwari, R., Prakash, V., Singh, J., Singh, R., Singh, H. (2025). Self-Learning AI Agents for Adaptive Cyber Defense in Internet of Things Ecosystems. networks, 53(54), 56.

[16] Li, X., Wang, Z., Huang, Y., Chen, H. (2023). A survey on self-evolving autonomous driving: a perspective on data closed-loop technology. IEEE Transactions on Intelligent Vehicles, 8(11), 4613-4631.

[17] Gheibi, O., Weyns, D. (2024). Dealing with drift of adaptation spaces in learning-based self-adaptive systems using lifelong self-adaptation. ACM Transactions on Autonomous and Adaptive Systems, 19(1), 1-57.

[18] Tang, J., Qin, Y., Tang, F., Zhao, M., Kato, N. (2025). Unifying AI for Networking and Networking for AI: the Self-evolving Edge Learning. IEEE Transactions on Mobile Computing.

[19] Uzair, M. Y., Rehan, M., Chand, M. (2025). Policy-Governed Self-Evolving Architecture for Autonomous AI Agents in Enterprise Systems. INTERNATIONAL JOURNAL OF ENGINEERING DEVELOPMENT AND RESEARCH, 13(4), 510-519.

[20] Zhao, C., Zhang, R., Wang, J., Niyato, D., Sun, G., Wang, X., ... Jamalipour, A. (2025). From agentification to self-evolving agentic ai for wireless networks: Concepts, approaches, and future research directions. arXiv preprint arXiv:2510.05596.