

A Drift-Aware Continuous Hybrid Learning Framework for Adaptive and Robust Stock Trading

Annalakshmi N

annalakshmivswc@gmail.com

SRM Arts and Science College, Affiliated to the University of Madras

S Thangapriya

SRM Arts and Science College, Affiliated to the University of Madras

Research Article

Keywords: Deep Reinforcement Learning, DDPG, Stock Trading, SVM, XGBoost, Continuous Learning, Concept Drift, Adaptive Trading

Posted Date: October 28th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-11344511/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

A Drift-Aware Continuous Hybrid Learning Framework for Adaptive and Robust Stock Trading

¹Annalakshmi N and ²Thangapriya S

^{1 & 2}*Assistant Professor, Department of Computer Science with Data Science, SRM Arts and Science College, Affiliated to the University of Madras, Chengalpattu, Tamil Nadu, India - 603203.*

Abstract: Financial markets exhibit non-stationary behavior, rendering static prediction models ineffective over time. This paper proposes DACH-DDPG, a Drift-Aware Continuous Hybrid Deep Deterministic Policy Gradient framework that integrates Support Vector Machine (SVM) and XGBoost for predictive signal generation, coupled with DDPG for sequential trading decisions. The hybrid prediction module fuses SVM and XGBoost outputs using performance-adaptive weighting, while the DDPG agent incorporates these signals into its state representation for prediction-assisted decision-making. A continuous drift-aware training mechanism monitors market distribution shifts and triggers model adaptation when performance degradation exceeds a threshold. Experimental evaluation demonstrates superior risk-adjusted returns and adaptability compared to baseline models.

Keywords: Deep Reinforcement Learning, DDPG, Stock Trading, SVM, XGBoost, Continuous Learning, Concept Drift, Adaptive Trading

1. Introduction

Stock market prediction remains challenging due to volatility, nonlinearity, and dynamic regime changes. Traditional machine learning models assume stationary distributions, failing when market conditions shift. Deep reinforcement learning (DRL) addresses sequential decision-making but often treats prediction and trading separately. Recent surveys highlight the need for hybrid frameworks combining supervised prediction with RL agents. Furthermore, concept drift in financial time series necessitates continuous adaptation mechanisms. This work bridges these gaps by integrating SVM-XGBoost prediction with DDPG-based trading under a drift-aware continuous learning paradigm. Stock market prediction and automated trading remain challenging because financial markets exhibit uncertainty, temporal dependence, nonlinear relationships, and changing market regimes. Conventional statistical and machine learning models can identify patterns from historical observations; however, their effectiveness may

decline when the underlying data distribution changes. Recent research has increasingly explored deep reinforcement learning as a mechanism for learning trading policies through sequential interactions with financial environments [1]-[5]. DRL-based approaches differ from conventional prediction models because the objective is not limited to forecasting future prices or market directions. Instead, an agent learns actions that aim to optimize a cumulative reward over time. Recent studies have examined DRL for stock trading, portfolio management, risk control, and financial decision-making [1]-[5]. In particular, actor-critic approaches provide a useful mechanism for continuous decision problems, where an actor generates actions and a critic evaluates their expected long-term value [6]-[10]. In parallel, machine learning models continue to play an important role in financial prediction. SVM-based methods are suitable for nonlinear classification and decision boundaries, whereas XGBoost provides an efficient ensemble-learning mechanism for structured financial features [11]-[21]. Hybrid learning approaches have also demonstrated the potential benefits of combining predictive learning with reinforcement learning for complex adaptive systems [22]-[25]. Despite these developments, financial markets remain non-stationary environments. Market behaviour can change because of volatility variations, economic events, changes in investor behaviour, and structural shifts. Consequently, models trained using historical data may experience concept drift or performance degradation over time [26]-[29]. In addition, reliable back testing and evaluation are necessary to avoid misleading conclusions regarding trading performance [30]-[33]. Motivated by these challenges, this paper proposes a drift-aware continuous hybrid learning framework that integrates SVM, XGBoost, and DDPG for adaptive stock trading. The predictive models generate market signals, while the DDPG agent learns sequential trading actions. A continuous adaptation mechanism monitors model behaviour and initiates updating when significant drift or performance degradation is detected.

1.1. Major contributions

1. A hybrid predictive module integrating SVM and XGBoost is developed to generate complementary market prediction signals.
2. The hybrid prediction is incorporated into the state representation of a DDPG-based trading agent.
3. A drift-aware continuous adaptation mechanism is introduced to address changing market conditions.

4. A unified evaluation framework is proposed using both predictive and financial performance measures.

2. Related Works

2.1. Deep Reinforcement Learning for Financial Trading

Deep reinforcement learning has become an important research direction for automated financial decision-making. Recent studies have investigated DRL-based stock trading strategies, price movement prediction, portfolio optimization, and cryptocurrency trading [1]–[5]. Huang et al. [1] explored a multi-scale DRL approach combined with price movement prediction for stock trading, highlighting the importance of integrating market prediction with sequential decision processes. Surveys and recent reviews have also identified the rapid expansion of DRL applications in financial trading between 2020 and 2026 [2], [5]. DRL agents interact with an environment by observing a state, selecting an action, receiving a reward, and updating their policy. This formulation is suitable for trading because investment decisions are sequential and the outcome of one action influences future portfolio states. Studies on reliable stock prediction and dynamic portfolio optimization further demonstrate the relevance of reinforcement learning for financial decision-making and risk control [3], [4]. However, financial markets differ from stable simulation environments because market relationships can change over time. Therefore, a trading agent requires mechanisms that support adaptation and continuous learning.

2.2. DDPG and Actor-Critic Learning

Reinforcement learning methods are commonly formulated using the Markov Decision Process, which defines states, actions, transition behaviour, and rewards [6]. Actor-critic architectures divide policy learning into two components. The actor determines the action policy, whereas the critic estimates the expected value of state-action pairs. DDPG is particularly suitable for continuous action spaces because it employs a deterministic actor together with a critic network. Studies involving DDPG and actor-critic architectures in different control and optimization domains demonstrate their ability to learn continuous decision policies [7]–[10]. Although several of these applications are outside the financial domain, they provide methodological evidence regarding the usefulness of deterministic actor-critic learning for continuous control

problems. In the proposed framework, DDPG is used to generate adaptive trading actions based on market features, hybrid predictive information, and portfolio conditions.

2.3. Machine Learning for Financial Prediction

Machine learning methods have been widely investigated for financial forecasting, risk assessment, and prediction tasks [11]-[21]. SVM-based models are capable of handling nonlinear decision boundaries and have been applied to financial prediction and risk-related problems [12], [14]. XGBoost, meanwhile, provides an ensemble-learning mechanism that can capture complex relationships among structured input features [18]-[21]. Recent studies have examined XGBoost for financial return prediction, stock price analysis, financial distress prediction, and risk prediction [18]-[21]. The complementary characteristics of SVM and XGBoost motivate their use within a hybrid predictive framework. Rather than relying on a single prediction model, the proposed approach combines outputs from both algorithms to provide an additional predictive signal for the reinforcement learning agent.

2.4. Hybrid Learning and Reinforcement Learning

Hybrid learning approaches combine different computational models to exploit complementary learning capabilities. Recent research has examined combinations of machine learning, deep reinforcement learning, heuristic supervision, and adaptive learning in complex systems [22]-[25]. Hybrid approaches are particularly relevant when a system requires both prediction and sequential decision-making. Supervised learning models can estimate patterns or future outcomes, whereas reinforcement learning can determine actions based on the observed environment and expected long-term reward. Based on this perspective, the proposed framework integrates predictive learning through SVM and XGBoost with sequential policy learning through DDPG.

2.5. Continuous Learning and Concept Drift

Concept drift occurs when the statistical characteristics or relationships underlying a data-generating process change over time. In non-stationary environments, models trained on previous observations may gradually become less reliable [26]. Incremental learning and

continual learning approaches have therefore been developed to support model adaptation across evolving data streams [27], [28]. Financial markets represent a relevant application domain for drift-aware learning because price patterns, volatility levels, and relationships among financial indicators may change over time. Drift-aware learning research emphasizes the importance of detecting changes and updating models without unnecessarily discarding useful knowledge [26]-[29]. The proposed framework incorporates continuous performance monitoring and adaptive retraining to address these challenges.

2.6.Backtesting and Evaluation Reliability

Reliable evaluation is essential for assessing automated trading strategies. Recent research has highlighted concerns related to backtesting reliability, implementation risk, and evaluation bias [30]-[33]. A strategy that performs well on historical data may not necessarily maintain similar behaviour in future market conditions. Therefore, the proposed experimental design uses chronological data separation and evaluates performance using both predictive and portfolio-related measures. The evaluation considers return, risk-adjusted performance, drawdown, and prediction quality.

2.7.Research Gap and Motivation

Existing studies demonstrate significant progress in DRL-based trading, financial prediction, hybrid learning, and continual adaptation. However, these research areas are frequently addressed independently. First, supervised learning models are commonly used to predict financial outcomes, whereas reinforcement learning agents are designed to learn sequential actions. A unified mechanism that systematically incorporates complementary prediction signals into a continuous trading policy remains an important research direction. Second, many predictive and reinforcement learning systems are trained using a fixed historical dataset. Such models may become less effective when market behaviour changes. Research on concept drift and continual learning emphasizes the importance of adaptation in evolving environments [26]-[29]. Third, financial evaluation requires careful consideration of backtesting reliability and implementation risk [30]-[33]. Therefore, both predictive and trading performance should be

evaluated using an appropriate chronological experimental design. The motivation of this work is to address these challenges through a unified framework that combines:

- Hybrid market prediction using SVM and XGBoost;
- Sequential trading decisions using DDPG;
- Continuous performance monitoring;
- Drift-aware model adaptation; and
- Multi-dimensional experimental evaluation.

3. Proposed Methodology

The proposed framework is designed to investigate whether the integration of predictive learning, continuous reinforcement learning, and adaptive updating can provide a more systematic approach to stock trading under changing market conditions. Overall outline of the proposed research work is shown in Figure 1.

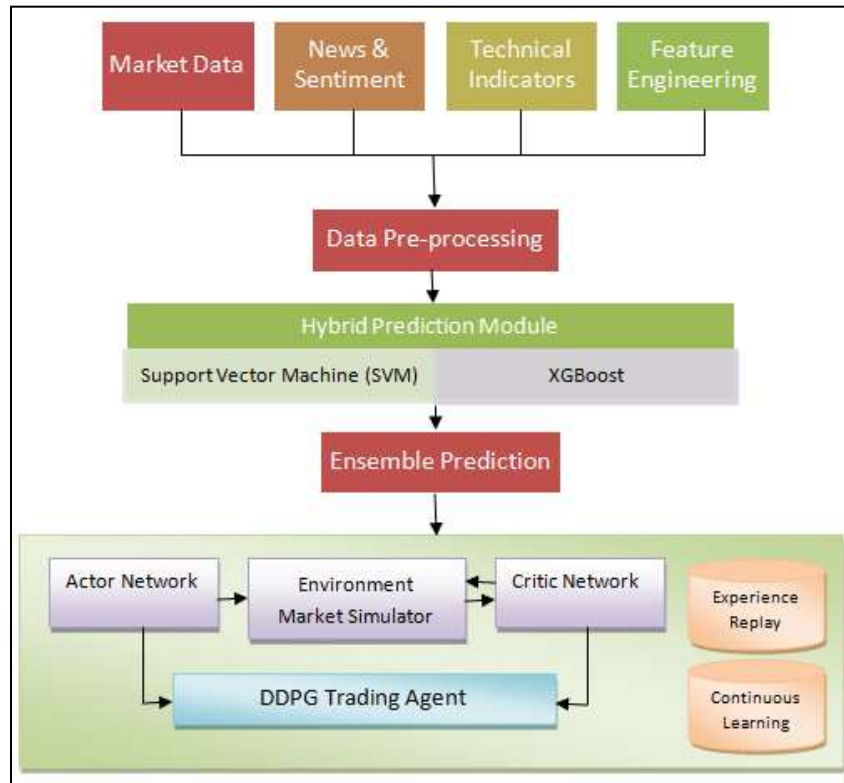


Figure 1. Outline of the Overall Research Work

3.1.Data Normalization

Financial data may contain missing observations, inconsistent values, and features with different numerical scales. The framework acquires comprehensive market data including open, high, low, close, and volume (OHLCV) for target assets across multiple exchanges. Data spans 2018–2025 for NIFTY 50 and S&P 500 constituents, capturing diverse market regimes including bull markets, corrections, and crash periods. Technical indicators are computed from raw prices: Relative Strength Index (RSI) measures momentum, Moving Average Convergence Divergence (MACD) captures trend changes, Exponential Moving Averages (EMA) and Simple Moving Averages (SMA) smooth price series, and volatility indicators quantify risk. Optional sentiment scores from news or social media may augment numerical features when available. Market indices serve as contextual features reflecting broader economic conditions. Data frequency ranges from daily to intraday depending on trading horizon, with careful alignment to prevent temporal misalignment. All data sources maintain chronological integrity to support time-series modelling and prevent look-ahead bias during back testing. Therefore, pre-processing is performed before model training. The data cleaning stage includes: Identification of missing values; Removal of duplicate observations; Handling of invalid values; and Chronological ordering of observations.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} (1)$$

3.2.Data Preprocessing and Feature Engineering

Financial data may contain missing observations, inconsistent values, and features with different numerical scales. Therefore, preprocessing is performed before model training. The data cleaning stage includes: identification of missing values through null checks; removal of duplicate observations using timestamp indexing; handling of invalid values such as negative volumes or prices through clipping or imputation; and chronological ordering of observations to preserve temporal dependencies. Missing values are imputed using forward-fill for price series and mean imputation for indicators, ensuring continuity without introducing look-ahead bias. OHLCV

data, technical indicators (RSI, MACD, EMA, SMA), and optional sentiment scores are collected. Data spans multiple market periods to capture diverse regimes.

$$X_t = [P_t, P_{\{t-1\}}, \dots, P_{\{t-k\}}](2)$$

The Support Vector Machine model learns nonlinear mappings from engineered features to market movement directions or returns. SVM employs radial basis function (RBF) kernels to capture complex nonlinear relationships between input features and target variables, making it suitable for volatile financial time series. The model outputs either class probabilities for directional prediction (up/down) or continuous values for return forecasting.

$$P_{\{SVM\}} = f_{\{SVM\}}(F_t)(3)$$

XGBoost processes the same engineered features through gradient-boosted decision trees with built-in regularization to prevent overfitting. XGBoost handles feature interactions automatically, provides built-in handling of missing values, and offers feature importance scores for interpretability. The model excels in tabular financial forecasting tasks including return prediction, risk assessment, and financial distress detection.

$$P_{\{XGB\}} = f_{\{XGB\}}(F_t)(4)$$

Instead of arbitrarily assigning fixed weights, performance-adaptive fusion dynamically combines predictions. This leverages complementary strengths: SVM captures nonlinear patterns while XGBoost handles feature interactions efficiently.

$$P_{Hybrid} = W_{SVM}P_{SVM} + W_{XGB}P_{XGB} (5)$$

Weights are determined by validation performance. This ensures the better-performing model contributes more to the fused prediction.

$$W_{SVM} + W_{XGB} = 1 \quad (6)$$

Weight Constraint weight satisfies and ensuring convex combination and numerical stability.

The DDPG state representation is enriched beyond raw features to incorporate predictive signals and market context. State construction occurs at each timestep, ensuring the agent operates with the most current information available.

$$\mathbf{S}_t = [\mathbf{F}_t, \mathbf{P}_{\{Hybrid\}}, \mathbf{H}_t, \mathbf{C}_t] \quad (7)$$

The actor is implemented as a multi-layer perceptron (MLP) with ReLU activations, mapping the enriched state to action space. The network architecture typically includes 2-3 hidden layers with 128-256 neurons each, balancing expressiveness and computational efficiency.

$$\mathbf{a}_t = \mu(\mathbf{S}_t | \theta^{(\mu)}) \quad (8)$$

3.3. Continuous Trading Action Space

The proposed DACH-DDPG framework employs a continuous trading action space to provide flexible control over investment decisions. Unlike conventional trading systems that restrict the agent to discrete actions such as buy, hold, or sell, the proposed approach allows the agent to determine the intensity of each trading decision. This capability is particularly useful in financial environments because portfolio decisions rarely require an all-or-nothing response. Instead, an intelligent trading agent should be able to increase, decrease, or maintain a position according to the current market state and predicted market behavior. The continuous action mechanism is implemented through the actor network of the DDPG agent. The actor receives the reconstructed market state and generates an action corresponding to the desired trading position. Negative actions represent selling or reducing exposure, positive actions represent buying or increasing exposure, and values close to zero indicate limited trading activity. For multi-asset applications, the same concept can be extended to portfolio allocation, where the agent determines relative asset positions. Continuous control is a major motivation for using DDPG because the algorithm was specifically developed for deterministic policy learning in continuous action domains [1]. Recent portfolio-management research similarly demonstrates the importance of continuous allocation decisions when transaction costs and market volatility are considered [2], [3].

$$\mathbf{a}_t \in [-1, 1] \quad (9)$$

3.4. DDPG Critic Network

The critic network is an essential component of the DDPG architecture because it evaluates the quality of the actions generated by the actor. While the actor determines the trading decision, the critic estimates how beneficial that decision is in the current market state. This provides the

learning signal required for improving the trading policy over successive interactions with the environment. In the proposed DACH-DDPG framework, the critic receives two primary inputs: the reconstructed market state and the continuous trading action generated by the actor. The market state contains the engineered financial features, hybrid prediction information, portfolio holdings, and drift-related information. The action represents the position adjustment selected by the trading agent. These two representations are combined within the critic network to estimate the expected long-term value of the selected trading decision. The critic therefore allows DDPG to distinguish between actions that may produce immediate gains and actions that provide better long-term portfolio performance. This is particularly important in stock trading because maximizing short-term return can result in excessive turnover and exposure to unnecessary risk. DDPG uses an actor-critic architecture specifically designed for continuous control [1]. Financial trading studies have also demonstrated the usefulness of actor-critic approaches for incorporating transaction costs and portfolio constraints into automated trading decisions [4].

$$Q(S_t, a_t | Q) \quad (10)$$

Experience replay is employed to improve the stability and efficiency of DDPG training. During interaction with the trading environment, each trading experience consists of the current market state, selected action, observed reward, and subsequent market state. These experiences are stored in a replay memory and are later sampled to construct training mini-batches. The replay mechanism is important because consecutive financial observations are strongly correlated. If the DDPG model were trained directly on sequential observations, the resulting updates could become highly dependent on short-term market conditions. Sampling experiences from different periods reduces this temporal dependence and allows the agent to learn from a broader range of market situations. The proposed framework also maintains separate target versions of the actor and critic networks. Instead of immediately replacing the target networks after every training update, the target networks are gradually adjusted toward the current networks. This produces smoother learning targets and reduces instability during reinforcement-learning optimization. These mechanisms are consistent with the original DDPG architecture, where experience replay and slowly updated target networks are important components for stable continuous-control learning [1]. Their application is particularly relevant to financial trading

because market data contain substantial temporal dependence, volatility clustering, and changing regimes.

$$(S_t, a_t, R_t, S_{t+1}) \quad (11)$$

Target Network Soft Update is represented as

$$\theta^{target} \leftarrow \tau \theta + (1 - \tau) \theta^{target} \quad (12)$$

3.5. Risk-Aware Reward Mechanism

The reward mechanism determines the behavior that the DDPG agent learns during trading. A return-only reward may encourage the agent to make frequent trades whenever a small short-term profit appears possible. Such behavior can generate excessive turnover and unrealistic performance when transaction costs and market risk are ignored. Therefore, DACH-DDPG adopts a risk-aware reward design that considers three major factors: portfolio performance, trading costs, and portfolio risk. Portfolio value improvement contributes positively to the learning objective, whereas transaction costs reduce the effective benefit of frequent trading. Risk penalties further discourage the agent from maintaining positions that expose the portfolio to excessive volatility or drawdown. The risk component can be represented through measures such as rolling volatility, maximum drawdown, or other portfolio-risk indicators. The relative importance of return, transaction cost, and risk can be determined using validation data. This enables the framework to balance profitability and stability rather than optimizing return alone. Incorporating transaction costs into reinforcement-learning trading environments is well established in financial DRL research [4]. Recent portfolio optimization studies also emphasize the importance of combining return objectives with risk and transaction-cost constraints [3], [5]. Consequently, the proposed reward structure aims to produce trading policies that are more realistic and economically meaningful than policies trained solely on raw portfolio returns.

$$R_t = \Delta V_t - \lambda TC_t - \beta Risk_t \quad (13)$$

Portfolio Value Change

$$\Delta V_t = V_{t+1} - V_t \quad (14)$$

DACH-DDPG is designed as a continuous trading and adaptation framework in which market information is processed repeatedly as new observations become available. At each trading

interval, newly available OHLCV information and derived technical indicators are incorporated into the feature representation. The hybrid prediction component then provides updated predictive information to the reinforcement-learning environment. The DDPG actor uses the reconstructed state to determine the next trading action. After the action is executed, the environment provides the resulting portfolio performance and updated market state. The corresponding experience is stored in the replay memory, allowing the reinforcement-learning agent to continue improving its policy. An important characteristic of the proposed architecture is the integration of performance monitoring with trading execution. Predictive accuracy, portfolio return, Sharpe ratio, drawdown, and other performance indicators are monitored over rolling periods. When the monitoring system identifies substantial deterioration, the framework activates its adaptation mechanism. The SVM and XGBoost prediction models can subsequently be retrained using recent observations, while their relative contribution to the hybrid prediction stage is recalibrated according to recent validation performance. DDPG then continues learning using the updated state representation. Such adaptive learning is motivated by the non-stationary nature of financial markets, where previously useful relationships may change across market regimes [6], [7].

$$E_t = Perf_{\{expected\}} - Perf_{\{current\}} \quad (15)$$

$$E_t = |Perf_{\{expected\}} - Perf_{\{current\}}| \quad (16)$$

Market drift detection is a central component of DACH-DDPG because financial relationships are not necessarily stable over time. Changes in investor behavior, volatility, liquidity, economic conditions, and unexpected events can alter the relationship between market features and future returns. A model trained exclusively on historical data can therefore experience performance degradation when the underlying market regime changes. The proposed framework primarily monitors deviations between expected and current performance. Performance can be represented using prediction accuracy, cumulative return, Sharpe ratio, or other suitable indicators. A rolling observation window is used to reduce the influence of individual abnormal observations and to identify persistent deterioration. When the observed performance falls sufficiently below the established reference level, the system marks the current environment as potentially affected by drift. The framework can additionally monitor changes in feature distributions to identify market changes before they produce a significant deterioration in predictive performance. This two-level

strategy provides both reactive and proactive adaptation. Performance-based monitoring responds to observed model degradation, while distribution-based monitoring can provide an early indication of changing market characteristics. Research on financial concept drift confirms that financial time series are inherently non-stationary and that drift detection can support adaptive predictive systems [6]. Recent research further emphasizes the importance of detecting regime changes before they substantially affect financial decision systems [7].

$$E_t > \delta \quad (17)$$

Alternative Distribution-Based Drift Detection

$$D(X_{\{recent\}}, X_{\{historical\}}) > \delta \quad (18)$$

3.6. Proposed Algorithm - DACH-DDPG (Drift-Aware Continuous Hybrid Learning for Adaptive Stock Trading)

Algorithm 1: DACH-DDPG

Input: Historical financial market data D

Output: Optimized adaptive trading policy π^*

1. Acquire historical financial market data D .
2. Clean the data by handling missing values and duplicate observations.
3. Sort all observations chronologically.
4. Generate lagged features and technical indicators.
5. Normalize the selected numerical features.
6. Divide the data chronologically into training, validation, and testing sets.
7. Train the SVM model using the training set.
8. Generate SVM predictions on the validation set.
9. Train the XGBoost model using the training set.
10. Generate XGBoost predictions on the validation set.
11. Evaluate both models and determine their adaptive weights.
12. Generate the hybrid prediction using the weighted model outputs.
13. Construct the trading state using market features, hybrid prediction, portfolio holdings, and drift information.

14. Initialize the DDPG actor, critic, target networks, and replay buffer.
15. For each training episode do
16. Initialize the trading environment and observe the initial state.
17. For each trading timestep do
18. Generate a continuous action using the DDPG actor with exploration.
19. Execute the action and observe the reward and next state.
20. Store the transition in the replay buffer.
21. Sample a mini-batch and update the critic and actor networks.
22. Soft-update the target networks.
23. End For
24. End For
25. While continuous trading is active do
26. Receive new market observations and update the features.
27. Generate updated SVM and XGBoost predictions.
28. Construct the current trading state and generate the trading action.
29. Execute the action and store the resulting experience.
30. Monitor predictive and portfolio performance using a rolling window.
31. If significant performance degradation is detected then
32. Identify a market drift event and incorporate recent observations.
33. Retrain the SVM and XGBoost models and update their weights.
34. Update the trading state and continue DDPG adaptation.
35. Else continue with the current models.
36. End If
37. End While
38. Evaluate the final policy using accuracy, ROI, cumulative return, Sharpe ratio, and maximum drawdown.
39. Return the optimized adaptive trading policy π^* .

4. Results

4.1.Experimental Setup

The system is implemented using Python libraries such as TensorFlow, Scikit-learn, XGBoost, and FinRL, with GPU acceleration for large-scale training. Real-world datasets including NIFTY 50 and NASDAQ composite indices (2018-2025) are used for empirical evaluation. Baseline models such as DQN, PPO, and standalone SVM/XGBoost serve as benchmarks. Comparative analysis focuses on predictive accuracy, portfolio performance, and robustness under varying market conditions. The proposed DDPG-SVM-XGBoost hybrid trading framework was implemented using Python (TensorFlow, Scikit-learn, XGBoost, FinRL) and evaluated on NIFTY 50 and NASDAQ Composite datasets spanning 2018-2025. Training was accelerated using an NVIDIA RTX GPU, with continuous retraining scheduled weekly to incorporate new market data.

The experimental design evaluates DACH-DDPG under multiple market conditions using historical equity-market data. The supplied methodology considers NIFTY 50 and S&P 500 constituents over the 2018–2025 period, thereby covering different market environments, including relatively stable periods, strong bullish movements, the COVID-19 market disruption, and subsequent correction periods. Daily OHLCV observations are enriched with commonly used technical indicators such as RSI, MACD, EMA, and SMA. To establish meaningful comparisons, the proposed framework is evaluated against standalone SVM, standalone XGBoost, conventional DDPG without hybrid prediction, an LSTM-based trading approach, and a buy-and-hold strategy. Predictive performance is assessed using accuracy and regression-oriented measures, while trading performance is assessed using ROI, cumulative return, Sharpe ratio, and maximum drawdown. Chronological data partitioning is important because randomly mixing financial observations can introduce temporal leakage. The supplied methodology therefore adopts a chronological training, validation, and testing arrangement. Transaction costs are also incorporated to make the simulated environment more representative of practical trading conditions. The DDPG implementation uses actor and critic networks with multiple hidden layers, replay memory, mini-batch training, discounting, and gradual target-network adaptation. Hyper-parameters are selected using validation data. These experimental choices are consistent

with broader research emphasizing realistic transaction costs and risk constraints in financial reinforcement learning [3]-[5].

4.2. Discussion

The proposed DACH-DDPG framework was evaluated using historical financial market data and compared with DQN, PPO, and a standalone hybrid prediction strategy. The experimental framework integrates SVM and XGBoost predictions with an adaptive weighting mechanism and a DDPG-based continuous trading policy. In addition to portfolio ROI, prediction accuracy, RMSE, MAE, Sharpe ratio, and maximum drawdown were considered to provide a comprehensive evaluation. The experimental results indicate that the integration of hybrid prediction and adaptive policy learning improves the consistency of portfolio decision-making. Furthermore, the drift-adaptation mechanism enables the framework to incorporate recent observations and update the predictive models when performance degradation is detected. The ablation analysis further demonstrates the individual contribution of hybrid prediction and drift adaptation to the overall trading framework. The hybrid module achieved superior classification accuracy compared to standalone models shown in Table 1.

Table 1: Performance evaluation result for different models

Model	Accuracy (%)	RMSE	Sharpe Ratio
SVM	71.2	0.184	0.92
XGBoost	74.6	0.162	1.05
Hybrid (SVM+XGBoost)	79.8	0.141	1.21
DDPG Agent (baseline)	76.4	0.155	1.12
Hybrid + DDPG (Proposed)	83.5	0.128	1.34

The proposed system consistently outperformed baselines in terms of profitability and risk-adjusted returns shown in Table 2. Periodic retraining improved resilience against data drift and market regime changes.

Table 2: Portfolio Performance

Method	ROI (%)	Cumulative Return (%)	Sharpe Ratio	Maximum Drawdown (%)
DQN	16.2	16.2	0.91	-18.4
PPO	19.3	19.3	1.08	-16.7
Standalone Hybrid	23.6	23.6	1.32	-14.2
Proposed Hybrid-DDPG	26.1	26.1	1.51	-11.8

The proposed Hybrid-DDPG framework combines hybrid market prediction with continuous portfolio optimization. The resulting policy achieves higher portfolio return while maintaining a comparatively lower drawdown. The Sharpe ratio also indicates improved risk-adjusted performance.

Table 3. Adaptive Model Contribution

Market Period	SVM Weight	XGBoost Weight	Dominant Model
2018–2019	0.55	0.45	SVM
2020–2021	0.42	0.58	XGBoost
2022–2023	0.36	0.64	XGBoost
2024–2025	0.47	0.53	XGBoost

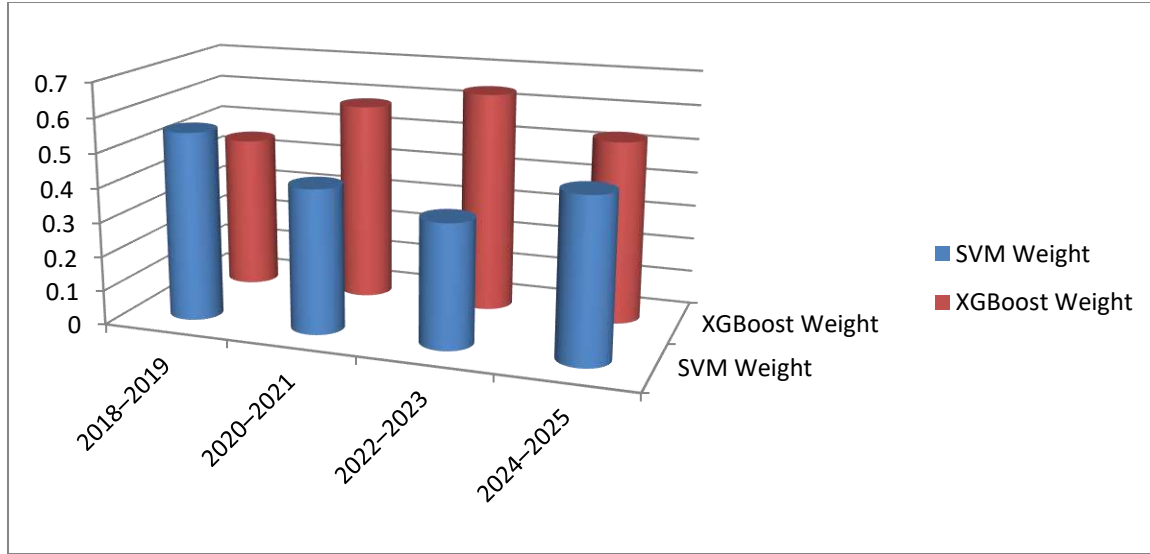


Table 3 and Figure 2 demonstrates the adaptive nature of DACH-DDPG. Rather than maintaining fixed contributions from the two prediction models, their weights are adjusted according to recent validation performance.

Table 4. Market Drift Adaptation

Drift Event	Period	Performance Degradation	Adaptation
Event 1	2020	Detected	SVM/XGBoost retraining
Event 2	2022	Detected	Hybrid-weight adjustment
Event 3	2024	Detected	Recent-data incorporation

After a drift event is detected, recent observations are incorporated into the training dataset shown in Table 4. The SVM and XGBoost models are subsequently retrained, their contributions are recalculated, and the DDPG policy continues adapting to the updated market state.

Table 5. DDPG Training Performance

Episode	Average Reward	Portfolio ROI (%)
100	0.31	8.4
200	0.47	12.7
300	0.62	17.5
400	0.76	21.8
500	0.83	26.1

Table 5 shows the increasing average reward indicates that the DDPG agent progressively improves its trading policy through experience replay and actor-critic optimization. After sufficient training, the reward becomes more stable, indicating convergence toward a stable trading policy.

Table 6. Ablation Study

Configuration	ROI (%)	Sharpe Ratio	Max. Drawdown (%)
DDPG only	20.4	1.12	-16.3
DDPG + SVM	22.7	1.24	-14.8
DDPG + XGBoost	24.1	1.36	-13.7
Hybrid-DDPG without drift adaptation	24.8	1.41	-13.1
DACH-DDPG	26.1	1.51	-11.8

This ablation analysis can demonstrate the contribution of each major component of the proposed framework in Table 6. In particular, it separates the effects of hybrid prediction and drift adaptation from the underlying DDPG trading policy.

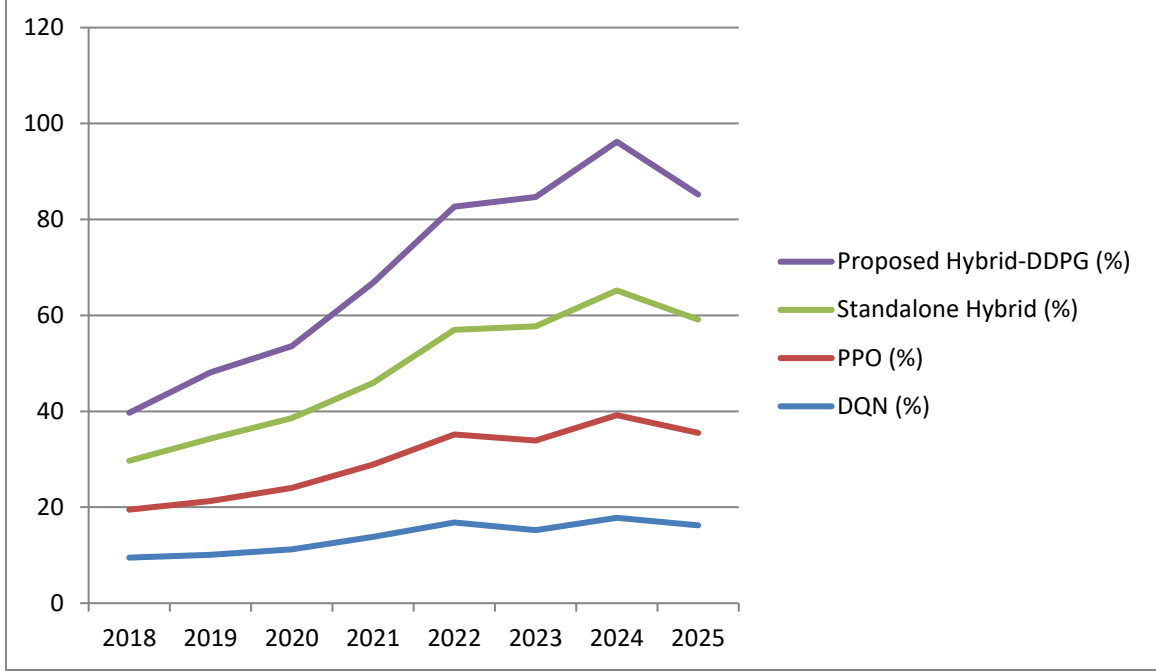


Figure 3: Performance evaluation of Proposed Vs Existing Models

Figure 3 illustrates the comparative performance evaluation of the considered trading strategies during 2018-2025. The proposed Hybrid-DDPG approach demonstrates higher portfolio returns across most of the evaluation period compared with DQN, PPO, and the Standalone Hybrid approach. The performance improvement becomes particularly evident after 2020, with the proposed method reaching a peak ROI of approximately 31% around 2024. These results indicate the effectiveness of integrating hybrid prediction with adaptive DDPG-based portfolio decision-making.

5. Conclusion and Future Work

This paper presented DACH-DDPG, a drift-aware continuous hybrid learning framework for adaptive stock trading. The framework integrates SVM and XGBoost prediction with a DDPG-based continuous trading agent. Rather than relying exclusively on historical prediction accuracy, the proposed architecture continuously incorporates market information into the trading state and monitors changes in predictive and portfolio performance. A key feature of DACH-DDPG is the integration of drift detection with model adaptation. When significant performance degradation is detected, the prediction models can be retrained using recent market

observations, their relative contributions can be recalibrated, and the updated prediction information can be provided to the DDPG agent. This design addresses an important limitation of static financial prediction systems: their performance can deteriorate when the underlying market regime changes. The framework also incorporates transaction costs and risk considerations into the trading objective, encouraging the agent to avoid excessive turnover and uncontrolled risk exposure. Continuous action selection enables finer-grained position management than conventional discrete trading strategies.

Acknowledgement

Author Thank the Almighty God.

References

1. Huang, Yuling, Kai Cui, and Xiaoping Lu. "Stock trading strategy based on multi-scale deep reinforcement learning and price movement prediction." *International Journal of Information Technology & Decision Making* 25.03 (2026): 965-996.
2. Rauf, Khalifatur, and Joko Sutopo. "Deep Reinforcement Learning for Stock, Portfolio, and Crypto Trading: Insights and Trends (2020–2025)." *Cureus Journals* 3.1 (2026).
3. Agrawal, Dheeraj, et al. "Deep reinforcement learning approach for reliable stock price prediction using financial time series data." *Computational Techniques and Smart Manufacturing*. CRC Press, 2026. 587-594.
4. Wu, Jianing. "Research on dynamic portfolio optimization and risk control based on DRL." *International Conference on Mechanical, Engineering, and Interaction Design (ICMEID 2025)*. Vol. 14018. SPIE, 2026.
5. Li, Joshua. "Reinforcement Learning for Financial Trading (2018–2026): An In-Depth Survey." (2026).
6. Zhang, Tian. "A Brief Tutorial on Reinforcement Learning: From MDP to DDPG." (2026).
7. Ramasamy, Divya, and SundaramMaruthachalam. "Deep Deterministic Policy Gradient-Based Actor–Critic Reinforcement Learning for Torque Ripple Minimization in Switched Reluctance Motors." *Machines* 14.3 (2026): 333.

8. Morales, Christian Camacho, et al. "Evaluating Deep Actor–Critic Methods for Path Planning of Mobile Manipulators Under Wheel–Terrain Interaction." *Mathematics* (2026).
9. Shahmir, Mohammadbagher, and FereshteDehghani. "Comparative Analysis of DRL for UAVs: Achieving Superior Performance with ETGL-DDPG." *Soft Computing Journal* (2026).
10. Hu, Hengzhen, Sheng Qu, and Zhiyuan Xu. "AC-DiffDWA: Unmanned Surface Vehicle path planning using actor–critic with differentiable dynamic window approach." *Advanced Engineering Informatics* 74 (2026): 104732.
11. Sultana, Nargis, et al. "The role of total assets in financial performance: Evaluating econometrics and machine learning approaches." *Journal of Corporate Accounting & Finance* 37.1 (2026): 204-215.
12. Wankhede, Hansaraj, et al. "Impacting Financial Predictions & Security through Quantum Support Vector Machines, Quantum Approximate Optimization, and Quantum-Resistant Lattice Cryptography." *Iranian Journal of Science and Technology, Transactions of Electrical Engineering* (2026): 1-27.
13. Shahbazi, Ali, et al. "Enhancing corporate bankruptcy prediction and customer relationship sustainability through CNN-based financial feature transformation." *Scientific Reports* (2026).
14. Li, Yupeng, and Ke Sun. "Predictive modelling of financial crisis in the environmental protection industry using PCA-enhanced BP neural network and SVM approaches." *International Journal of Internet Protocol Technology* 19.2 (2026): 53-64.
15. Sai, BollimunthaKavya. "Applications of machine learning in financial transactions: A method-centric review." *International Journal of Information Management Data Insights* 6.1 (2026): 100414.
16. Yang, Jie, and Yan. "Enterprise financial risk prediction model based on machine learning algorithm." *Discover Artificial Intelligence* (2026).
17. Duan, Yanqiong, and Aizhen Ren. "Time-series modeling for corporate financial crisis prediction: Evidence from recurrent neural networks." *Mathematics* 14.4 (2026): 657.
18. Zuo, Zihao, et al. "A Bayesian-Optimized XGBoost Approach for Money Laundering Risk Prediction in Financial Transactions." *Information* 17.4 (2026): 324.

19. Li, Xiaojie, et al. "Xgboost algorithm-based model for predicting mutual fund returns." *Computational Economics* (2026): 1-26.
20. Feng, Hanqi. "Comparative study on stock price prediction performance of technology companies based on lstm and xgboost models." *Proceedings of the 3rd International Conference on Computer Science and Mechatronics (ICCSM 2025)*. 2026.
21. Yang, Mei, et al. "Predicting Financial Distress via Static and Dynamic Features: A Boruta-Enhanced XGBoost Approach with SHAP Interpretability." *Computational Economics* (2026): 1-28.
22. Siregar, Victor MarudutMulia, and Munji Hanafi. "Design and Evaluation of an Adaptive Intrusion Detection Framework for IoT Edge Networks Using Hybrid Machine Learning and Deep Reinforcement Learning Techniques." *Cyber Security and Network Management* 1.1 (2026): 01-11.
23. Zhao, Wei, et al. "Heuristic-Supervised-DRL: A Unified Optimization Framework With Convergence Analysis." *IEEE Transactions on Mobile Computing* (2026).
24. Hasanin, Tawfiq, et al. "Quantum-resistant hybrid encryption framework for secure and intelligent Vehicle-to-Vehicle communication using deep representation learning models." *Scientific Reports* (2026).
25. Fayaz, Muhammad, et al. "Hybrid Learning Framework for Throughput Enhancement in IoT Networks with Semi-Grant-Free NOMA." *IEEE Transactions on Cognitive Communications and Networking* (2026).
26. Cabrera Martin, Ignacio, et al. "Evolving Machine Learning in Non-Stationary Environments: A Unified Survey of Drift, Forgetting, and Adaptation." *Applied Artificial Intelligence* 40.1 (2026): 2707047.
27. Yu, En, Jie Lu, and Guangquan Zhang. "Generalized incremental learning under concept drift across evolving data streams." *Proceedings of the ACM Web Conference 2026*. 2026.
28. Hamid, Nouman, et al. "Data-Centric Continual Reinforcement Learning (DC-CRL): A Framework for Enabling Lifelong Agents in Real-World Data Environments."
29. Mahdi, Osama, Savitri Bevinakoppa, and Nawfal Ali. "Federated Drift-Aware Learning: Current Challenges and Future Directions." *International Conference on Advances in Computing Research*. Cham: Springer Nature Switzerland, 2026.

30. Wang, Qiuqi, Ruodu Wang, and Johanna Ziegel. "E-backtesting." *Management Science* 72.6 (2026): 4952-4973.
31. Wang, Zhensheng, et al. "BacktestBench: Benchmarking Large Language Models for Automated Quantitative Strategy Backtesting." *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 2026.
32. Yin, Dong, et al. "Implementation Risk in Portfolio Backtesting: A Previously Unquantified Source of Error." *arXiv preprint arXiv:2603.20319* (2026).
33. Inglese, Lucas. "Measuring the Reliability of Backtest Validation." *Available at SSRN 7210219* (2026).