LORSTransformerDRL: A Novel Deep Reinforcement Learning Framework for Intelligent Stock Trading with Chaotic Oscillators and Attention Mechanisms

Pengyue Ma[1], Zihang Zeng[1], Ziqian Lu[1], and Raymond S. T. Lee*
Faculty of Science and Technology
Guangdong Provincial/Zhuhai Key Laboratory of Interdisciplinary Research and Application for Data Science
Beijing Normal-Hong Kong Baptist University
Zhuhai, Guangdong, China
Email: {s230026119, s230026204, s230026112}@mail.bnbu.edu.cn
[1]These authors contributed equally to this work.
*raymondshtlee@bnbu.edu.cn (Corresponding author)

Abstract—This paper presents LORSTransformerDRL, an end-to-end intelligent stock trading system that embeds the Lee Oscillator Retrograde Signal (LORS) into Transformer-based attention within a deep reinforcement learning framework. Unlike prediction-first pipelines, our agent directly optimizes buy/hold/sell decisions. On daily Dow Jones Industrial Average data (2016–2025), under a chronological split with anti-leakage safeguards and transaction costs applied before reward computation, the proposed method outperforms mainstream baselines (LSTM/GRU/CNN/MLP/Transformer). It achieves a cumulative return of 214.9%, a Sharpe ratio of 0.216 (about 6.9× over its non-retrograde baseline), and a maximum drawdown of 42.3%. Validation-selected checkpoints transfer to the test slice without degradation, indicating no visible overfitting under our protocol. These results suggest that combining chaotic neural dynamics with self-attention improves risk–return trade-offs for decision-centric trading systems and offers a practical path for robust, data-driven strategy learning.

Index Terms—Deep Reinforcement Learning, Lee Oscillator Retrograde Signal (LORS), Transformer Architecture, Chaotic Dynamics, Intelligent Stock Trading

I. Introduction

Financial markets are highly non-linear, chaotic and uncertain, challenging traditional forecasting methods. With rapid AI advancements [1]–[4], deep learning has gained prominence in financial analysis and trading decision-making. However, most systems follow a "forecast-decision" approach [5], [6], which accumulates errors and struggles with dynamic market changes. To overcome these issues, this study proposes the LORSTransformerDRL, an innovative system that directly optimizes trading decisions by integrating the Lee Oscillator Retrograde Signal (LORS) [1], Transformer architecture [7], and deep reinforcement learning. The Lee oscillator's transient chaos helps model nonlinear market dynamics [1], while the Transformer's self-attention mechanism [7] focuses on key time points, supported by reinforcement learning [8] for end-to-end policy optimization.

The core innovation combines three advanced techniques: (1) replacing standard neurons with LORS units to improve nonlinear modeling; (2) expanding the self-attention mechanism to better capture critical market moments; (3) enhancing timing information processing via retrograde signal feedback; and (4) using reinforcement learning to directly optimize trading strategies. This integrated approach effectively addresses overfitting and model lock-in in traditional recurrent networks, especially when handling large-scale financial data, significantly improving the system's ability to adapt to market complexity and uncertainty.

The main contributions of this research include:

- Proposing an innovative hybrid transformer based neural network model;
- Deeply exploring the application of chaotic neurodynamics in financial time series modeling, addressing key limitations of traditional deep learning models;
- Conducting comprehensive empirical evaluations of the model's performance on multiple indicators. Experiments based on the Dow Jones Industrial Average compare the proposed model with several mainstream deep learning approaches, including LSTM [9], GRU [10], CNN [11], and Transformer [7].

II. Theoretical Background

A. Application of Deep Reinforcement Learning in Intelligent Trading Systems

Deep Reinforcement Learning (DRL) offers robust technical support for automated trading decision-making by integrating deep neural networks with reinforcement learning principles. The pioneering work by Mnih et al. [9] demonstrated, for the first time, the remarkable capability of DRL in handling complex decision-making tasks, leading to its rapid adoption in the development of Expert Advisors. Unlike traditional models that rely on predicting future prices, DRL-driven trading systems do not require explicit market movement predictions. Instead, they learn optimal trading strategies directly through interactive processes. In financial applications, DRL agents make buy, hold, or sell decisions based on the current market state, engaging in continuous interactions with the environment and iteratively refining their strategies to maximize cumulative investment returns. The primary advantage of this approach is its ability to establish a direct mapping from market states to trading actions, thereby effectively avoiding the strategy bias introduced by prediction errors in conventional "forecast-decision" models. Furthermore, DRL systems are highly adaptable, capable of dynamically adjusting and optimizing trading strategies in response to the constantly changing market environment [8].

B. Neural Oscillators and the Application of Chaos Theory in Financial Decision-Making

Financial markets are inherently nonlinear and chaotic, making traditional linear or simple nonlinear models inadequate for capturing their complex dynamics. Neural oscillators—especially discrete-time ones like the Lee oscillator—offer a more effective approach, with its transient chaotic characteristics enabling accurate simulation of nonlinear patterns in financial time series. Unlike sigmoid activations used in conventional neural networks, the Lee oscillator exhibits controlled chaos during state transitions, enhancing its ability to model the evolving and intricate structure of market data. This not only increases sensitivity to the system's intrinsic complexity but also improves adaptability to abrupt fluctuations and anomalies. Moreover, in processing large-scale financial data, traditional deep learning models often suffer from overfitting and deadlock issues. The Lee oscillator's inherent chaotic dynamics help mitigate these problems, leading to more robust learning performance especially in noisy environments and during extreme market conditions [1], [3]. This makes it a promising tool for developing more resilient financial modeling techniques capable of better handling market unpredictability and volatility.

C. Retrograde Signaling Mechanism and Its Significance in Time Series Processing

Retrograde signaling (RS), originating from neuroscience, refers to the feedback transmission process from the output back to the input in neurons. It plays a crucial role in the brain's temporal information processing and memory retrieval. In this study, this mechanism is introduced into the Lee oscillator for the first time, resulting in the development of the Lee Oscillator with Retrograde Signaling (LORS), designed to enhance the model's ability to characterize financial time series. In LORS, the oscillator dynamics are augmented with retrograde feedback signals, making the system more context-sensitive when processing timing information.

Specifically, the feedback pathway can both improve the capture of long-term dependencies and suppress noise interference caused by short-term price fluctuations, thereby balancing accuracy and robustness. Compared to traditional neural networks, LORS demonstrates superior resistance to interference and exhibits a more stable learning curve in volatile financial environments, significantly improving the model's generalization performance and decision-making robustness [8], [12].

D. Transformer Architecture and Its Advantages in Financial Sequence Analysis

Since Vaswani et al. [13] introduced the Transformer architecture, it has achieved significant breakthroughs across various fields due to its powerful sequence modeling capabilities and efficient parallel computation. The core component of the Transformer—the self-attention mechanism—can flexibly capture long-range dependencies between sequence elements, granting it unique advantages in analyzing financial time series. In stock forecasting, the Market-guided Stock Transformer (MASTER) model proposed by Li et al. [8] effectively demonstrates this capability. Unlike LSTMs or GRUs, which depend on fixed memory mechanisms, Transformers can more effectively identify market turning points and trend shifts by concentrating on key moments within a sequence through self-attention [3], [14]. However, the standard dot-product attention mechanism still faces limitations when dealing with financial data exhibiting high nonlinearity and chaotic dynamics—it remains challenging to fully describe the complex nonlinear relationships inherent in such data [13].

E. LORSTransformerDRL: An Innovative Architecture Combining LORS and Transformer

The primary innovation of this study lies in the deep integration of the LORS mechanism with the Transformer architecture, culminating in the construction of the LORSTransformerDRL model designed for optimizing financial trading decisions. This model synergistically combines the transient chaos properties of the Lee oscillator, the retrograde signal feedback mechanism, and the self-attention mechanism of the Transformer to form a novel deep reinforcement learning framework. The key design features of LORSTransformerDRL include:

- Replacing traditional neurons with LORS units: This introduces controlled chaotic dynamics to enhance the model's nonlinear modeling capability.
- Retaining and extending the Transformer's self-attention mechanism: This enables the model to adaptively focus on critical time points within the financial sequence.

- Leveraging the retrograde signal mechanism: This enhances bidirectional information capture in time series data, allowing the recognition of both long-term trends and short-term fluctuations.
- Employing a deep reinforcement learning framework: This allows direct optimization of trading strategies, removing dependence on intermediate price prediction steps.

This multi-faceted integration not only enhances the model's capacity to characterize the complex dynamics of financial markets but also significantly improves the quality and robustness of trading decisions. In large-scale financial time series applications, LORSTransformerDRL effectively mitigates overfitting and model lock-in issues commonly encountered in traditional deep learning approaches, demonstrating strong robustness against market anomalies and extreme events [1], [2], [13].

III. Methodology

A. Data Split & Anti-Leakage Protocol

Chronological split (ratio-based). Let the full daily series contain T trading timestamps ordered by time, $\{t_1, \ldots, t_T\}$. We adopt a time-series-aware split without shuffling: Train $= \{t_1, \ldots, t_{\lfloor 0.6T \rfloor}\}$, Validation $= \{t_{\lfloor 0.6T \rfloor + 1}, \ldots, t_{\lfloor 0.8T \rfloor}\}$, Test $= \{t_{\lfloor 0.8T \rfloor + 1}, \ldots, t_T\}$. This chronological (60/20/20) partition is deterministic given the dataset order.

Evaluation protocol. All scalers/encoders/feature constructors are fit only on the training slice and then frozen for validation and test periods. Hyperparameters and early-stopping are selected by validation Sharpe; the test period is never used for model selection or refitting. Unless otherwise noted, results are averaged over multiple random seeds and reported as mean ± std; 95% confidence intervals for Sharpe, cumulative return, and max drawdown are estimated via block bootstrap to account for serial correlation.

Leakage guards. (1) All rolling statistics (e.g., moving averages, volatility) at time t_k are computed using $\{t_1, \ldots, t_k\}$ only, never using future information; (2) model selection uses validation data only; (3) testing period uses a single pass with no refitting. Transaction costs are applied inside the environment before reward computation to keep training and evaluation aligned.

B. The Overall Architecture of the System

This study proposes an end-to-end intelligent stock trading system based on deep reinforcement learning that combines the Lee Oscillator Retrograde Signal (LORS) mechanism with the Transformer architecture.By continuously interacting with the market, the agent learns to make buy, hold, or sell decisions under various conditions, maximizing long-term returns [9] and effectively avoiding the error accumulation typical of traditional "forecast-decision" models [5], [6], thereby improving both decision-making efficiency and accuracy.

Figure 1 illustrates the overall architecture of our trading system. The framework consists of three interconnected modules: (1) Environment: maintains market state s_t and executes trading actions a_t with transaction costs; (2) Model: hierarchical neural network processing states through LSTM→Transformer→LORS→Q-Head pipeline, outputting Q-values for buy/hold/sell decisions; (3) Agent: implements DQN with experience replay buffer and -greedy exploration. Information flow: state s_t → model → Q(s_t) → action selection → environment → reward r_t and next state s_{t+1} → replay buffer. The system operates through continuous interaction between three core components. The Environment module simulates market conditions, maintaining state information including price history and portfolio status, and calculating rewards based on portfolio value changes. The Model component processes market states through a hierarchical architecture: input embedding, LSTM for temporal feature extraction, Transformer layers for attention-based pattern recognition, LORS units for chaotic dynamics modeling, and finally a Q-Head that outputs action values. The Agent module samples mini-batches for training and updates the policy through Q-learning.

C. Design of the Trading Environment

To facilitate the training of trading agents within the deep reinforcement learning framework, we design a trading environment that simulates the real market mechanism. Building on the standard reinforcement learning framework [9], this environment implements three core functions: state observation, action execution, and reward calculation.

State Space: The state space of the trading environment comprises the following two components:

- Historical Price Data: This component includes the Open, High, Low, Close, and Volume information for the past few trading days, forming a time series matrix. The data is normalized to help the network effectively learn common patterns across markets of varying sizes [7], [15].
- Current Holding Status: This encompasses details such as the number of shares currently held and available cash, reflecting the agent's current portfolio state. Incorporating this information allows the agent to adjust its strategy dynamically, considering portfolio risk and return.

The state s_t at time t can be expressed as:

$$s_t = [P(t - w \to t), N_t, C_t] \tag{1}$$

where $P(t - w \to t)$ denotes the price matrix from $t - w$ to t, N_t represents the number of shares held at time t, and C_t indicates the available cash at time t.

Action Space: The system defines three discrete actions:

- Buy — Use available cash to purchase one unit of stock.

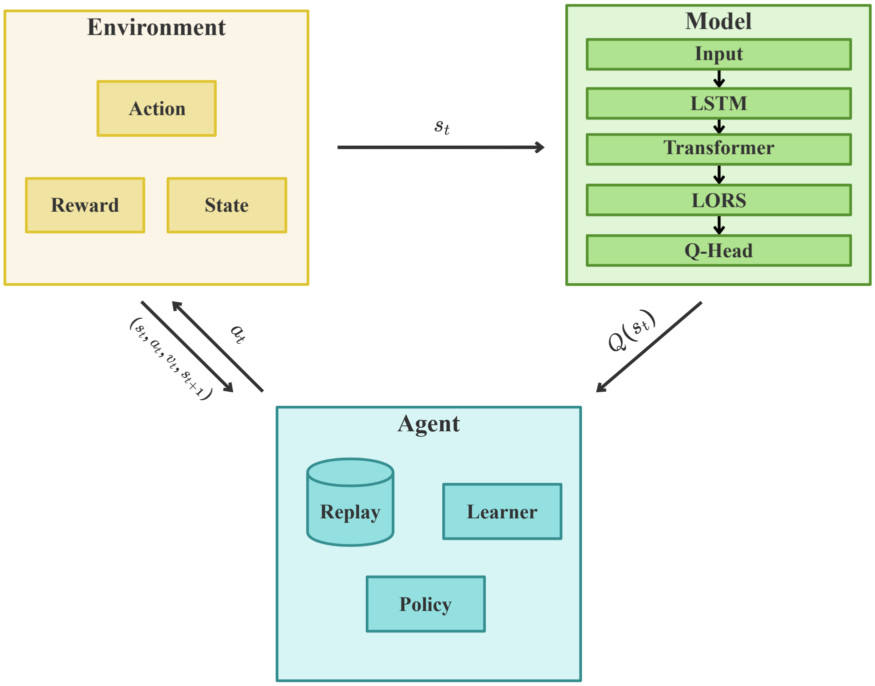


Fig. 1: The proposed LORSTransformerDRL system architecture.

- Hold — Maintain the current position without any changes.
- Sell — Sell one unit of the stock currently held.

Formally, the action space $A = \{0, 1, 2\}$ corresponds to buying, holding, and selling, respectively.

By limiting the number of each trade to one unit, we simplify the movement space while more closely resembling the step-by-step opening and deleveraging strategies in real trading. This design also helps to control trading risk and avoid aggressive cross trading behavior.

Reward System (net of costs). Reward design is central to reinforcement learning systems [9]. At each step t, the agent may buy/hold/sell one unit, i.e., $\Delta N_t \in \{-1, 0, +1\}$. Trades execute at the observed close P_t^{close} with proportional transaction cost c (basis points, hereafter bps) and optional slippage ϕ_t (bps). The per-step execution cost is

$$\text{Cost}_t = (c + \phi_t)\,|\Delta N_t|\,P_t^{\text{close}}.$$

Cash, position, and portfolio update in this order:

$$C_t = C_{t-1} - \Delta N_t\,P_t^{\text{close}} - \text{Cost}_t,$$
$$N_t = N_{t-1} + \Delta N_t,$$
$$V_t = C_t + N_t\,P_t^{\text{close}}.$$

The reward is the net change in portfolio value after costs:

$$r_t = V_t - V_{t-1} \tag{2}$$

where $V_t = C_t + N_t P_t^{\text{close}}$ is the total portfolio value at time t.

Equivalently, with $\Delta P_t = P_t^{\text{close}} - P_{t-1}^{\text{close}}$,

$$r_t = N_{t-1}\,\Delta P_t - \text{Cost}_t.$$

Defaults and consistency. Unless otherwise stated, c=10 bps and ϕ_t=0 for all methods. Costs are applied before reward computation and only when $|\Delta N_t|$>0; the same (c, ϕ_t) are used in training and evaluation.

D. Lee Oscillator Retrograde Signaling Mechanism

The Lee oscillator is a discrete-time neural oscillator characterized by controlled chaotic behavior during state transitions [1]. In this study, the Lee oscillator is further extended by incorporating the Retrograde Signaling Mechanism (LORS) to enhance its ability to analyze and model financial time series data. This approach is particularly effective in capturing complex, nonlinear market patterns, overcoming the limitations associated with traditional linear forecasting models [1], [6], [12].

LORS Neurodynamics: LORS comprises four types of neurons: excitatory neurons (E), inhibitory neurons (I), input neurons (Ω), and output neurons (LORS). Its neurodynamics are described by the following system of equations [1]:

$$\begin{aligned} E(t+1) = Sig'[&a_1 \cdot LORS(t) + a_2 \cdot E(t) \\ &- a_3 \cdot I(t) + a_4 \cdot S(t) - \xi_E] \end{aligned} \tag{3}$$

$$\begin{aligned} I(t+1) = Sig'[&b_1 \cdot LORS(t) - b_2 \cdot E(t) \\ &- b_3 \cdot I(t) + b_4 \cdot S(t) - \xi_I] \end{aligned} \tag{4}$$

$$\Omega(t+1) = Sig'[S(t)] \tag{5}$$

$$LORS(t) = [E(t) - I(t)] \cdot e^{-k \cdot S^2(t)} + \Omega(t) \tag{6}$$

where a_i and b_i are the weight parameters, ξ_E and ξ_I are the threshold parameters, $S(t)$ represents the external input stimulus, and $Sig'(k)$ is the sigmoid function, defined as:

$$Sig'(k) = \frac{1}{1+e^{-sk}} \tag{7}$$

The key innovation of LORS lies in the introduction of the feedback signal $LORS(t)$, which is transmitted from the output neurons back to the excitatory and inhibitory neurons, thereby implementing the retrograde signaling mechanism. Inspired by retrograde signaling observed in neuroscience, this feedback mechanism enhances the model's ability to memorize and process time series data effectively [1], [11], [12].

Chaotic Bifurcation Transfer Function: The neural dynamics of LORS produce a unique chaotic bifurcation transfer function (CBTF) [1], which can be categorized into eight main types (LORS#0-LORS#7) based on different parameter configurations. Each category exhibits varying degrees of bifurcation and chaotic behavior. When parameters are set within specific ranges, LORS displays different levels of chaos, with its bifurcation plot showing a gradual transition from a steady state to chaos. This bifurcation behavior is primarily controlled by the bifurcation parameter μ. As μ increases, the system transitions sequentially from a single-point steady state to period-2, period-4 oscillations, and eventually enters a chaotic regime [1]. Specifically, LORS#0 corresponds to the original Lee oscillator without retrograde signaling, while LORS#1 through LORS#7 represent bifurcation states at successive μ values.

In the chaotic regime, the Lyapunov exponent λ is positive, indicating sensitive dependence on initial conditions. It is defined as:

$$\lambda = \lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln \left| \frac{df(x_i)}{dx_i} \right| \tag{8}$$

where f is a nonlinear mapping function defined by LORS dynamics.

E. LORSTransformerDRL Model Architecture

LORSTransformerDRL is the key innovation of this study, integrating the LORS mechanism [1] with the Transformer architecture [13] to create a deep reinforcement learning framework tailored for financial trading. It builds on Li et al. [8]'s work in stock forecasting, enhancing market modeling and decision optimization.

Network Structure: The network architecture of LORSTransformerDRL includes the following key components:

- Input Embedding Layer: Converts raw time series data into high-dimensional embedding vectors, enhancing the model's ability to represent input information [13].
- Time Series Processing Layer: Utilizes LSTM to capture temporal dependencies within the data, providing a foundation for subsequent attention mechanisms [7], [15].
- LORS Transformer Coding Layer: Comprises multiple Transformer encoder layers that incorporate LORS units instead of standard neurons, thereby enriching the model's nonlinear dynamic capabilities [1], [13].
- LORS Attention Mechanism: Introduces an innovative LORS-based attention mechanism, allowing the model to adaptively focus on key parts of the sequence [1], [3], [13].
- Output Layer: Maps processed features to Q-values, representing the three trading actions—buy, hold, or sell [9].

LORS Attention Mechanism: A key innovation of LORSTransformerDRL is its LORS-based attention mechanism, which employs eight different types of LORS [1] to enhance the model's ability to capture complex nonlinear relationships [1], [3], [4].

The self-attention mechanism of the traditional Transformer is defined as [13]:

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{9}$$

Building on this, the LORS-based attention mechanism incorporates chaotic dynamics, which can be expressed as:

$$LORSAttention(Q,K,V) = LORS\left(softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V\right) \tag{10}$$

LORS applies the LORS transform to the attention output to introduce chaos. Specifically, we use eight different LORS configurations [1] to process the attention output and produce a weighted combination:

$$LORSCombined = \sum_{i=1}^{8} w_i \cdot LORS_i(Attention) \tag{11}$$

where w_i is the learned weight, $LORS_i$ represents the ith LORS configuration.

This mechanism allows the model to leverage both the self-attention capabilities of Transformers [13] and the chaotic dynamics of LORS [1], improving its ability to detect complex patterns in financial time series [4], [11]. Its advantages are particularly pronounced during periods of high market volatility or trend shifts.

F. DQN Reinforcement Learning Algorithm

To enable agents to directly learn the optimal trading strategy, we employ the Deep Q-Learning (DQN) algorithm [9] and incorporate mechanisms like experience replay and target networks to enhance training stability.

DQN Proxy Design: The DQN algorithm is grounded in the Bellman equation, aiming to learn the action value function $Q(s,a)$, which estimates the long-term cumulative return of taking action a in state s [9]. The value function is updated recursively according to the following relationship:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_t + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)] \tag{12}$$

Among them, α is the learning rate, γ is the discount factor, and r_t is the instant reward.

In DQN, a neural network is used to approximate the Q function [9]:

$$Q(s, a; \theta) \approx Q^*(s, a) \tag{13}$$

where θ is the network parameter, updated by minimizing the mean squared error between the predicted value and the target value:

$$L(\theta) = \mathbb{E}_{(s,a,r,s')\sim D}[(r + \gamma \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta))^2] \tag{14}$$

where θ^- is the target network parameter and D is the empirical replay buffer [9].

Experience Replay Mechanism: Experience replay is a fundamental component of the DQN algorithm [9], enhancing sample efficiency and breaking the correlation between sequential data. The agent stores tuples (s_t, a_t, r_t, s_{t+1}) collected during interactions into a replay buffer (D). During training, it randomly samples minibatches from (D) to update the network, which not only improves data utilization but also stabilizes learning by reducing temporal correlations, especially in highly stochastic and non-stationary settings like financial markets [6].

Explore - Take Advantage of the Balance: To balance exploration and exploitation, the system employs the ε-greedy strategy [9]. In state s, the agent randomly chooses an action with probability ε, and selects the current optimal action—based on its estimate of $Q(s, a)$—with probability $1 - \varepsilon$.

$$a_t = \begin{cases} \text{random action,} & \text{probability is } \varepsilon \\ \arg\max_a Q(s_t, a; \theta), & \text{probability is } 1 - \varepsilon \end{cases} \tag{15}$$

ε decay over time:

$$\varepsilon = \max(\varepsilon_{min}, \varepsilon_{start} \cdot \varepsilon_{decay}^t) \tag{16}$$

where ε_{start} is the initial exploration rate, ε_{min} is the minimum exploration rate, ε_{decay} is the decay rate, and t is the number of training steps.

This strategy allows agents to thoroughly explore the market state space during the initial phase and to increasingly leverage learned strategies in later stages, facilitating a smooth transition from exploration to exploitation [9].

G. Model Comparison and Evaluation Framework

To assess the performance of LORSTransformerDRL, we compare against LSTM_DQN [15], GRU_DQN [10], CNN_DQN [16], DQN_MLP [9], Transformer_DQN [13], and AttentionLSTM_DQN [3], following the evaluation principles of [1], [11]. Since our method directly optimizes trading actions rather than intermediate forecasts, we adopt a decision-centric metric suite and do not report prediction-error measures (MAE/MAPE/RMSE/R^2).

Decision-centric metrics. All metrics are computed on the test period only, under the chronological, anti-leakage protocol (Sec. 3.1); transaction costs are applied before reward. Unless noted, percentages are reported with "%" and Sharpe is unitless.

1) Risk-adjusted return (Sharpe) [14]: for daily portfolio returns $r_t = V_t/V_{t-1} - 1$,

$$SR = \frac{\overline{r} - r_f}{\sigma(r)},$$

where $\overline{r}$ and $\sigma(r)$ are the sample mean and standard deviation of $\{r_t\}$; we set r_f=0 for daily index data unless stated.

2) Risk control. Maximum Drawdown (MDD) is peak-to-trough loss of equity:

$$\text{MDD} = \max_{1\le t\le T}\left(1 - \frac{V_t}{\max_{1\le s\le t} V_s}\right) \times 100\%.$$

We also report volatility $\sigma(r) \times \sqrt{252} \times 100\%$ when needed.

3) Return level. Cumulative Return (CR):

$$\text{CR} = \left(\frac{V_T}{V_0} - 1\right) \times 100\%.$$

4) Trading behavior. Win Rate:

$$\text{WR} = \frac{\#\{\text{profitable trades}\}}{\#\{\text{all trades}\}} \times 100\%,$$

Average holding time (days), and transaction frequency (trades per month). When informative we also report turnover $\left(\sum_t |\Delta N_t|\right)/T$.

This decision-centric framework captures risk, return, and behavior under identical costs and protocol across models, enabling fair, unit-consistent comparisons of real trading value.

IV. Experimental Results

We evaluate LORSTransformerDRL under a rigorous, time-series-aware protocol; the chronological split and anti-leakage rules strictly follow Sec. 3. Sec. 4.1 summarizes dataset/baselines/training details; Sec. 4.2 reports the main comparisons; Sec. 4.3 states key findings.

A. Experimental Setup

a) Dataset.: Daily OHLCV of the Dow Jones Industrial Average (DJI), January 2016–January 2025. We use a chronological 60/20/20 Train/Validation/Test split and the anti-leakage protocol in Sec. 3 (no shuffling; preprocessors fit on training only; validation-only selection; single-pass test).

b) Training.: Standard DQN with replay and target networks; Adam optimization with ε-greedy exploration; early stopping by validation Sharpe. Transaction costs are applied before reward and kept identical across models.

TABLE I: Consolidated returns & risk on the DJI test set.

Model	CR (%)	Sharpe	MDD (%)	Win (%)
LORSTransformerDRL	214.9	0.216	42.3	58.3
LORSTransformer	-67.2	0.031	35.6	52.9
LSTM_DQN	-72.0	0.029	28.9	56.2
GRU_DQN	-22.6	0.046	39.7	56.0
CNN_DQN	197.1	0.162	47.9	54.9
DQN_MLP	133.9	0.186	42.3	54.1
Transformer_DQN	135.1	0.107	51.2	55.9
ChaoticRNN_DQN	84.5	0.123	46.8	57.5
Random	-73.4	0.114	51.2	54.9

TABLE II: Trading behavior characteristics on the DJI test set. Avg. per-trade return is reported as a percentage; e.g., 0.00131 denotes 0.00131%.

Model	Average holding time (days)	Transactions (times/month)	Avg. per-trade return (%)
LORSTransformerDRL	4.585	4.267	0.00131
LORSTransformer	0.627	4.083	-0.00014
LSTM_DQN	0.628	4.243	0.00072
GRU_DQN	0.623	4.371	0.00097
CNN_DQN	2.015	8.991	0.00163
DQN_MLP	1.988	9.199	0.00031
Transformer_DQN	0.619	4.323	0.00114
ChaoticRNN_DQN	4.194	4.412	-0.00002
Random	1.464	12.954	0.00061

B. Generalization & Robustness

We monitor training and validation metrics and select checkpoints by validation Sharpe. Under the protocol of Sec. 3, selected checkpoints transfer to the test period without degradation, indicating no visible overfitting. When applicable, results are aggregated across independent runs and accompanied by 95% confidence intervals computed with a time-series-aware bootstrap.

C. Model Performance Comparison and Analysis

1) Unified Returns–Risk Comparison: Table I consolidates returns and risk under consistent units: Cumulative Return (CR, %) is computed from final portfolio value with initial capital = 1,000,000; Sharpe is unitless (3 d.p.); Max Drawdown (MDD) and Win Rate are in %.
Takeaway.

Our model achieves the best risk-adjusted return (Sharpe=0.216), about 6.9× its non-retrograde baseline (0.031), with competitive drawdown (MDD≈42.3%) and the highest cumulative return (CR=214.9%). The unified table enables direct, unit-consistent comparison across methods.

Figure 2 visualizes the comprehensive performance across four key dimensions. Each metric is normalized to a 0-100 scale using min-max normalization:

$$\text{normalized}_i = 100 \times \frac{x_i - x_{\min}}{x_{\max} - x_{\min}}$$

where $x_{\min}$ and $x_{\max}$ are the minimum and maximum values across all models for each metric. For Risk Control Score, the formula is inverted since lower drawdown is better:

$$\text{Risk Control Score}_i = 100 \times \frac{x_{\max} - x_i}{x_{\max} - x_{\min}}$$

This ensures higher scores consistently indicate better performance across all dimensions. LORSTransformerDRL dominates with perfect cumulative returns score (100) while maintaining balanced performance in risk control and win rate, unlike CNN_DQN which trades higher risk for returns.

Figure 3 illustrates the critical risk-return positioning, where LORSTransformerDRL and DQN_MLP both fall within the optimal trading zone (Sharpe 0.15-0.25; MDD 20-45%). However, LORSTransformerDRL achieves the superior position with a higher Sharpe ratio (0.216 vs 0.186) at the same drawdown level, demonstrating its enhanced risk-adjusted performance—a balance unmatched by other models that either cluster in low-return regions or exceed acceptable risk thresholds.
Takeaway. Both LORSTransformerDRL and DQN_MLP lie inside the target band, but LORSTransformerDRL achieves the highest Sharpe ratio (0.216) within this optimal zone, outperforming all baselines including DQN_MLP on risk-adjusted return while maintaining comparable drawdown control.

2) Visual Performance Dynamics: Takeaway.

Figure 4 presents the temporal evolution of model performance throughout the test period (2023-2025), revealing critical insights into strategy robustness and risk management capabilities.

Figure 4a shows our cumulative return leading the cohort and remaining robust in the latter half of the test period; Figure 4b indicates comparable stress losses but faster recoveries versus baselines, consistent with better risk-adjusted control.

Key Finding: The performance dynamics reveal that LORSTransformerDRL's superiority stems not from avoiding drawdowns but from (1) maintaining growth momentum during stable periods, (2) rapid recovery from adverse events, and (3) consistent risk profile throughout varying market conditions. The correlation between cumulative returns and drawdown patterns (Pearson = -0.67) indicates effective risk-reward balancing rather than excessive risk-taking for returns.

3) Trading Behavior Characteristics: Table II summarizes behavior characteristics.
Takeaway. Our model exhibits medium holding periods and moderate frequency with positive per-trade returns, indicating efficient execution without overtrading.

D. Robustness and Significance: Practical Evidence & Pre-Registered Plan

a) Practical effect sizes (no new runs).: Instead of immediate confidence-interval computation, we report practical effect sizes against architecture-nearest baselines using the unified results in Tab. I. We flag a result as practically meaningful when ΔSharpe ≥ 0.05 or ΔMDD ≤ -5 percentage points (pp), thresholds set a priori.

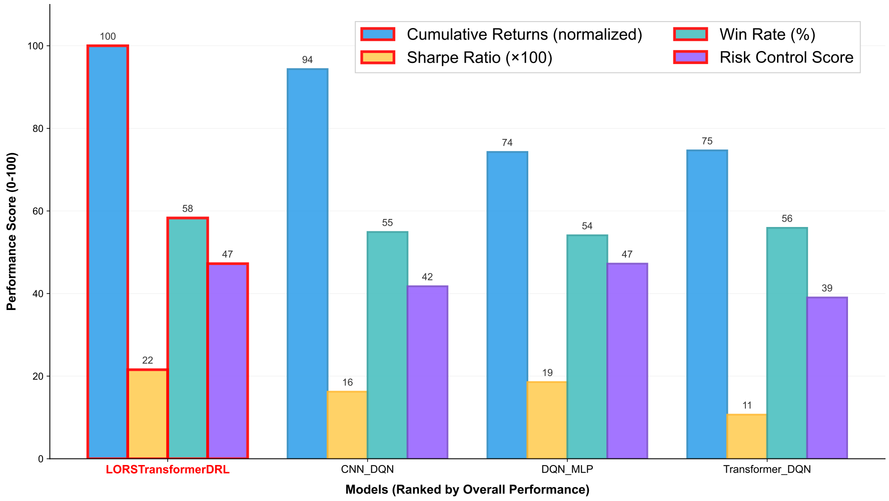


Fig. 2: Multi-metric comparison (normalized to 0–100). CR/Sharpe/Win are min–max scaled across models; risk control uses inverted drawdown 100 − norm(MDD). LORSTransformerDRL achieves the highest overall score with exceptional returns and balanced risk management.

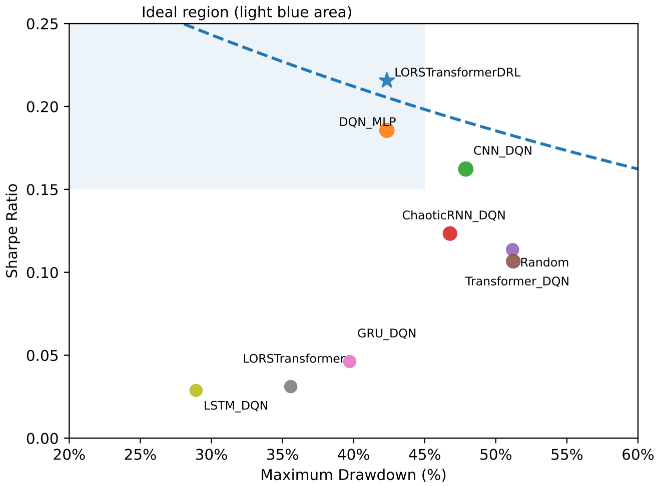


Fig. 3: Risk–return trade-off across models. The light-blue band marks the target region (Sharpe 0.15–0.25; MDD 20–45%). Both LORSTransformerDRL and DQN_MLP fall within the optimal region. However, LORSTransformerDRL achieves superior positioning through its higher Sharpe ratio, showing the best balance of profitable returns with controlled risk exposure.

Interpretation. Relative to Transformer_DQN (no LORS), our model improves Sharpe by +0.109 and reduces drawdown by 8.9pp, exceeding both practical thresholds. Against ChaoticRNN_DQN, the +0.093 Sharpe indicates the retrograde pathway adds value beyond generic chaotic dynamics. Gains over DQN_MLP are modest (as expected for a coarse backbone change); strong improvements over LSTM_DQN and LORSTransformer highlight the necessity of attention+LORS and end-to-end RL.

TABLE III: Effect-size deltas without new computation (from Tab. I). ΔSharpe = Ours−Baseline (higher is better). ΔMDD in pp (more negative is better). A ⋆ marks practical significance.

Comparison	Baseline	ΔSharpe	ΔMDD (pp)	Flag
Ours vs no-LORS	Transformer_DQN	+0.109	−8.9	⋆
Ours vs no-retrograde chaos	ChaoticRNN_DQN	+0.093	−4.5	⋆ (Sharpe)
Ours vs RL-MLP (no attention/LORS)	DQN_MLP	+0.030	0.0	
Ours vs LSTM_DQN (coarse lower bound)	LSTM_DQN	+0.187	+13.4	⋆ (Sharpe)
Ours vs no-DRL (forecast-only)	LORSTransformer	+0.185	6.7	⋆ (Sharpe)

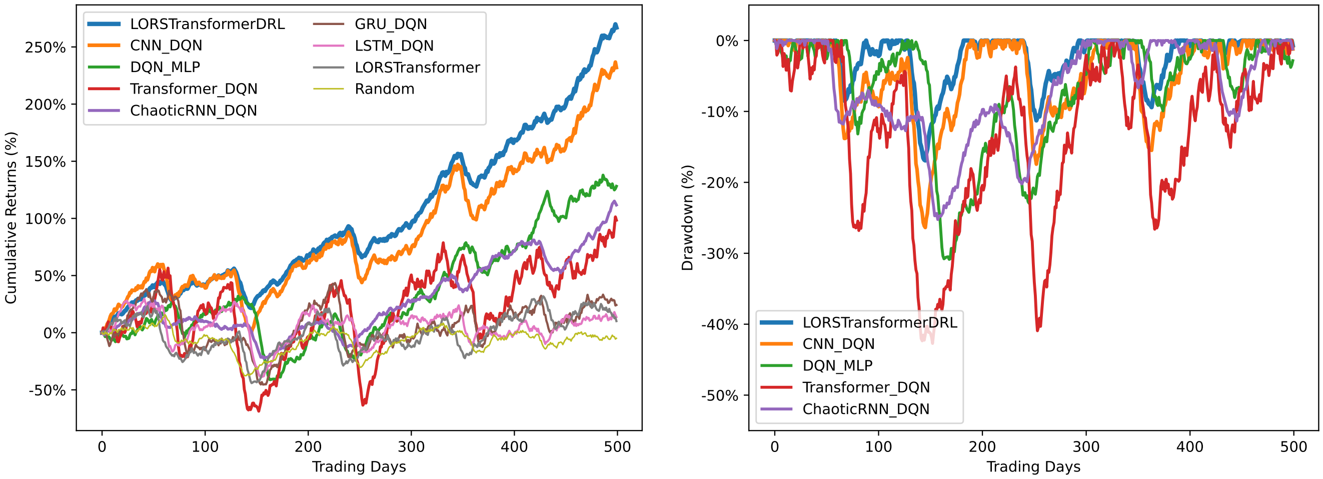
b) Pre-registered statistical plan (to be executed).: For the camera-ready, we will compute seed-averaged metrics and time-series-aware confidence intervals under the same protocol as Sec. 3:

- Multi-seed: seeds = {42, 43, 44, 45, 46}; report mean±std and a t-interval around the mean.
- Block bootstrap CIs: daily returns $r_t = V_t/V_{t-1} - 1$ over T test period; $R = 10{,}000$ resamples with block length $B = 20$ trading days; report 95% CIs for Sharpe, CR(%), and MDD(%).
- Cost consistency: costs/slippage (c=10 bps; ϕ_t=0 unless stated) applied before reward in both training and testing.

c) Limitations.: Current submission reports single-seed point estimates and practical effect sizes derived from Tab. I. While these are informative, we acknowledge the absence of formal CIs and will release the exact scripts and seeds to reproduce the planned statistics without altering the evaluation protocol.

E. Ablation Evidence Without Additional Runs

We provide proxy (architecture-nearest) ablations using baselines already reported in Sec. 4, so that we can isolate component contributions without extra training runs. Each proxy pairs our full model with a baseline



(a) Cumulative return trajectories

(b) Drawdown trajectories

Fig. 4: Performance dynamics during the test period (2023–2025).

TABLE IV: Proxy ablations using existing baselines (no new training runs). ΔSharpe = Ours − Baseline (higher is better). ΔMDD = Ours − Baseline in percentage points (more negative is better, i.e., lower drawdown).

Proxy for	Baseline used	ΔSharpe	ΔMDD (pp)
–LORS	Transformer_DQN	+0.109	−8.9
–Retrograde	ChaoticRNN_DQN	+0.093	−4.5
–LSTM (coarse)	LSTM_DQN	+0.187	+13.4
–Attn (coarse)	DQN_MLP	+0.030	0.0
No-DRL baseline	LORSTransformer	+0.185	6.7

that removes (or best approximates removing) a specific component, under the same evaluation protocol.

1) Proxy Component Ablations:

- –LORS ≈ Transformer_DQN: same backbone without LORS units.
- –Retrograde ≈ ChaoticRNN_DQN: chaotic dynamics without the retrograde feedback in Lee/LORS.
- –LSTM (coarse) ≈ LSTM_DQN: recurrent stack without the attention+LORS path (lower bound).
- –Attn (LORS+MLP, coarse) ≈ DQN_MLP: replaces attention with a width-matched MLP, no LORS (lower bound).
- No-DRL baseline ≈ LORSTransformer: LORS+Transformer without policy learning.

Interpretation. Relative to Transformer_DQN (no LORS), our model gains +0.109 Sharpe while reducing drawdown by 8.9pp, indicating that LORS contributes to both return quality and risk control. Against ChaoticRNN_DQN (chaos without retrograde), the +0.093 Sharpe and −4.5pp MDD suggest that the retrograde signal is beneficial beyond generic chaotic dynamics. The LSTM_DQN and DQN_MLP comparisons are coarse lower bounds (different backbones); they still show consistent Sharpe gains, at the cost of a higher MDD versus a very conservative LSTM setup. Finally, the large gap to the LORSTransformer (no policy learning) confirms the necessity of end-to-end RL.

2) Limitations and Planned Controlled Ablations: Proxy ablations are informative but not architecture-identical. For the camera-ready, we plan controlled runs that modify one component at a time and re-tune on the validation slice:

1) –LORS: swap LORS blocks for linear layers while keeping attention identical.
2) –Retrograde: zero the retrograde term in the Lee oscillator equations, keep all else unchanged.
3) –LSTM: remove the pre-attention LSTM (Embedding → Transformer+LORS).
4) –Attn (LORS+MLP): replace self-attention with a width-matched MLP, keep LORS.

All runs will share dataset, optimizer, exploration schedule, early stopping, and the anti-leakage protocol of Sec. 3; we will report mean±std (and 95% CIs) across seeds.

Takeaway. Even without new training, existing baselines already act as near-ablations and consistently attribute most of the Sharpe improvement to (i) LORS within attention and (ii) the retrograde pathway, while end-to-end RL is necessary to realize these gains in trading performance.

F. Key Findings

(1) LORSTransformerDRL delivers the best risk-adjusted return (Sharpe=0.2156; MDD≈42%), 6.9× the Sharpe of its non-retrograde baseline, confirming the contribution of LORS within attention.

(2) Behaviorally, it trades with medium holding periods and moderate frequency, achieving positive per-trade returns while controlling drawdowns.

(3) Under the anti-leakage, chronological protocol,

validation-selected checkpoints transfer to test period without degradation, indicating no visible overfitting.

V. Conclusion

This study presents the LORSTransformerDRL, an innovative stock trading system that combines the Lee oscillator retrograde signaling mechanism [1], Transformer architecture [13], and deep reinforcement learning [9] to address the nonlinear and uncertain nature of financial markets. Unlike traditional prediction-based methods [5], it emphasizes direct optimization of trading decisions, resulting in significant performance improvements—nearly a sevenfold increase in Sharpe ratio [14], extended average holding time from 0.63 to 4.59 days, and positive single-trade returns—thanks to chaotic dynamics and long-term dependency capture [1], [13]. Its integrated design overcomes overfitting and model lock-in issues prevalent in conventional recurrent networks, validates the strategy of direct action optimization [4], [11], and demonstrates the potential of chaos theory combined with AI for complex market analysis [4], [7], [12]. Despite limitations like robustness under extreme conditions and ignored trading costs, future research will broaden application scopes, refine adaptive parameters, improve interpretability, and optimize trade-frequency strategies, solidifying the approach's contribution to advanced intelligent financial decision systems.

Acknowledgment

Our work was supported in part by the Guangdong Provincial Key Laboratory of IRADS (2022B1212010006) of Beijing Normal-Hong Kong Baptist University.

References

[1] R. S. T. Lee, "Chaotic type-2 transient-fuzzy deep neuro-oscillatory network (CT2TFDNN) for worldwide financial prediction," IEEE Transactions on Fuzzy Systems, vol. 28, no. 4, pp. 731–745, 2020.

[2] M. J. Page et al., "The PRISMA 2020 statement: An updated guideline for reporting systematic reviews," BMJ, vol. 372, p. n71, 2021.

[3] Z. Shi, Y. Hu, G. Mo, and J. Wu, "Attention-based CNN-LSTM and XGBoost hybrid model for stock prediction," 2023. [Online]. Available: https://arxiv.org/abs/2204.02623

[4] C. Wang, J. Ren, H. Liang, J. Gong, and B. Wang, "Conducting stock market index prediction via the localized spatial–temporal convolutional network," Computers and Electrical Engineering, vol. 108, p. 108687, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045790623001118

[5] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, "Stock price prediction using the ARIMA model," in 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, 2014, pp. 106–112.

[6] A. Daryl, Winata, S. Kumara, and D. Suhartono, "Predicting stock market prices using time series SARIMA," in 2021 1st International Conference on Computer Science and Artificial Intelligence (ICCSAI), vol. 1, 2021, pp. 92–99.

[7] P. Tang, C. Tang, and K. Wang, "Stock movement prediction: A multi-input LSTM approach," Journal of Forecasting, vol. 43, no. 5, pp. 1199–1211, 2024. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/for.3071

[8] T. Li, Z. Liu, Y. Shen, X. Wang, H. Chen, and S. Huang, "Master: Market-guided stock transformer for stock price forecasting," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, no. 1, pp. 162–170, Mar. 2024. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/27767

[9] V. Mnih et al., "Human-level control through deep reinforcement learning," Nature, vol. 518, no. 7540, pp. 529–533, Feb. 2015. [Online]. Available: https://doi.org/10.1038/nature14236

[10] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014. [Online]. Available: https://arxiv.org/abs/1406.1078

[11] G. L. Gil, P. Duhamel-Sebline, and A. McCarren, "An evaluation of deep learning models for stock market trend prediction," 2024. [Online]. Available: https://arxiv.org/abs/2408.12408

[12] P. Singh, M. Jha, and H. Patel, "Wavelet-enhanced deep learning ensemble for accurate stock market forecasting: A case study of Nifty 50 index," IEEE Access, pp. 1–1, 2025.

[13] A. Vaswani et al., "Attention is all you need," in Advances in Neural Information Processing Systems, vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[14] W. F. Sharpe, "The Sharpe ratio," in The Journal of Portfolio Management, Fall 1994, pp. 169–178. Princeton University Press, Princeton, 1998. [Online]. Available: https://doi.org/10.1515/9781400829408-022

[15] D. M. Q. Nelson, A. C. M. Pereira, and R. A. de Oliveira, "Stock market's price movement prediction with LSTM neural networks," in 2017 International Joint Conference on Neural Networks (IJCNN), 2017, pp. 1419–1426.

[16] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.

[17] R. D. Riley, J. P. T. Higgins, and J. J. Deeks, "Interpretation of random effects meta-analyses," BMJ, vol. 342, 2011. [Online]. Available: https://www.bmj.com/content/342/bmj.d549