Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

An Explainable LSTM Framework for Stock Trading via Behavioral Decomposition and Competition-Driven Optimization

Iván García-Magariño

igarciam@ucm.es

Complutense University of Madrid

Javier Bravo-Agapito

Complutense University of Madrid

Research Article

Keywords:

Posted Date: November 24th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11842563/v1

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

Additional Declarations: No competing interests reported.

An Explainable LSTM Framework for Stock Trading via Behavioral Decomposition and Competition-Driven Optimization

Iván García-Magariño[1,2*] and Javier Bravo-Agapito[1]

[1*]Department of Software Engineering and Artificial Intelligence, Complutense University of Madrid, c/ Prof. José García Santesmases 9, Madrid, 28040, Madrid, Spain.

[2]Instituto de Tecnología del Conocimiento, Complutense University of Madrid, Complutense University of Madrid, c/ Prof. José García Santesmases 9, Madrid, 28040, Madrid, Spain.

*Corresponding author(s). E-mail(s): igarciam@ucm.es;
Contributing authors: jabrav01@ucm.es;

Abstract

Forecasting short-term stock price movements remains challenging due to market non-stationarity, volatility, and the complex aggregation of heterogeneous investor behaviors. Although Long Short-Term Memory (LSTM) networks have demonstrated strong performance in financial time-series prediction, most existing approaches operate as black boxes and provide limited insight into short-term directional changes, constraining trust and real-world deployment. This paper introduces an explainable LSTM-based trading framework based on subrogate agent-based simulation model that reframes prediction as the estimation of collective investor behavior rather than direct price forecasting. The proposed method decomposes historical price and volume series into behavioral components representing distinct investor roles. Separate LSTM models are trained to forecast the future proportions of each behavioral role. These predictions are aggregated into price-direction signals using behavioral-to-price rules whose parameters are optimized via a genetic algorithm. The genetic algorithm evolves decomposition weights and thresholds by evaluating candidate rule sets in repeated trading competitions, directly optimizing profitability and risk-adjusted performance rather than purely statistical error. This competition-driven optimization yields behaviorally consistent decompositions aligned with realistic market outcomes. The framework is implemented and evaluated within the Dashboard for Intelligent Investment Decisions with eXplainable artificial intelligence (DI2XAI), a multi-agent backtesting simulator using real Yahoo Finance data, enabling leakage-free and reproducible trading experiments across diverse market regimes. Extensive evaluations over multiple stocks, sectors, and historical periods show that the proposed approach consistently outperforms standard trading baselines with the LSTM, convolutional neural network with LSTM (CNN-LSTM), and CNN with bidirectional LSTM (CNN-BiLSTM), random forest and extreme-gradient boosting (XGBoost), achieving higher profitability. Our approach outperformed LSTM, CNN-LSTM, CNN-BiLSTM and random forest with statistically significant improvements ($p<0.001$), and outperformed XGBoost with an improvement of 12.5% in the final capital result.

Keywords: Long-short term memory networks, time-series decomposition, stock investment, trading strategies, explainable artificial intelligence

1 Introduction

The prediction of stock price trends remains one of the most challenging tasks in financial modeling, owing to the inherent characteristics of financial time series data. Stock markets exhibit profound non-stationarity, nonlinearity, and volatility, influenced by a myriad of unpredictable factors including macroeconomic indicators, geopolitical events, investor sentiment, and sudden market shocks [1]. Traditional econometric models, such as Autoregressive Integrated Moving Average (ARIMA) and Generalized Autoregressive Conditional Heteroskedasticity (GARCH), have been instrumental in capturing linear dependencies and volatility clustering but often falter in handling the complex, high-dimensional interactions that drive price movements [2]. This unpredictability is particularly acute in short- to medium-term forecasting, where even marginal improvements in directional accuracy can yield substantial economic benefits for traders and investors.

In recent years, Deep Learning (DL) techniques have revolutionized financial forecasting by leveraging their capacity to uncover latent patterns in sequential data without assumptions of linearity [3]. Among DL architectures, Long Short-Term Memory (LSTM) networks, a variant of Recurrent Neural Networks (RNNs), have emerged as particularly adept at modeling long-range dependencies in time series, mitigating issues like vanishing gradients that plague standard RNNs [4]. LSTMs have been extensively applied to stock price prediction, often outperforming benchmarks in tasks ranging from point forecasting to trend classification [5, 6]. Hybrid extensions, such as LSTM and Gated Recurrent Unit (GRU) ensembles or attention-augmented variants, further enhance robustness by integrating multi-scale features and external signals like sentiment from textual data [7, 8].

Despite these advances, key challenges persist: most models prioritize raw price prediction, yielding limited interpretability into the behavioral drivers of market dynamics, and are rarely validated in interactive, multi-agent environments that simulate real-world trading interactions [9, 10]. This work addresses these gaps through a novel LSTM-based framework that reframes prediction as the estimation of investor behavioral ratios, fostering explainability while maintaining predictive fidelity.

The primary research questions guiding this study are: (1) Can estimation of the decomposition proportions of distinct investor behavioral types from historical price series improve trading strategies either in profitability or reduction of volatility? (2) How does integrating such behavioral modeling into a serious game-based trading simulator enhance practical deployment with novel explainability?

Recent progress in explainable artificial intelligence (XAI) has underscored the importance of interpretability in financial modeling, where opaque "black-box" predictions often hinder trust and practical adoption by analysts and investors. In stock forecasting, this challenge is amplified by the market's behavioral complexity, where collective dynamics emerge from the interactions of diverse investor types rather than from price signals alone. Consequently, models capable of bridging the gap between prediction accuracy and behavioral transparency are essential for both academic understanding and real-world decision support. One of the main branches of explainability and interpretability is the use of agent-based subrogate simulation models [11] for explaining possible emergent collective behaviors from individual decisions.

Motivated by these considerations, this work reframes stock trend prediction as a behavioral estimation problem. Instead of directly forecasting prices, the proposed framework seeks to model the proportional influence of different investor archetypes—such as individual buyers, institutional investors, and automated traders—whose aggregate actions drive market movements. This approach not only enhances explainability by linking predictions to investor behavior but also offers a novel pathway for integrating human-like reasoning into algorithmic trading systems.

Our contributions are threefold: First, we propose an innovative transformation of price series into behavioral profiles (e.g., long-term buyers, automated stop-loss sellers, institutional manipulators), predicted via ensemble LSTMs, offering a domain-specific lens for market explainability absent in prior literature [7]. Second, we embed this model within the Dashboard for Intelligent Investment with Explainable AI (DI2XAI) [12], a backtesting simulator that leverages Yahoo Finance data for multi-agent validation, enabling scalable testing against human and automated traders. Third, we employ a genetic algorithm (GA) to derive adaptive rules for behavioral-to-price conversion, prioritizing directional precision and magnitude alignment, which demonstrates superior performance in simulated trading scenarios compared to vanilla LSTM baselines.

The remainder of this paper is organized as follows. Section 2 reviews the existing literature on deep learning techniques for stock market forecasting, with emphasis on LSTM-based and hybrid approaches. Section 3 details the proposed LSTM framework for behavioral decomposition and trading simulation. Section 4 presents the experimental setup, evaluation metrics, and comparative results against baseline models. Section 5 discusses the implications, interpretability, and potential applications of the findings. Finally, Section 6 concludes the study and outlines future research directions.

2 Related works

2.1 Different variants of RNNs for stock price prediction

The application of DL techniques to financial time series forecasting has garnered significant attention over the past decade, driven by the inherent non-stationarity and nonlinearity of stock market data. Traditional statistical models, such as ARIMA and GARCH, have traditionally been used for volatility modeling and price prediction [1]. However, these methods often fail to capture long-term dependencies and complex patterns in high-dimensional financial datasets. In contrast, DL architectures, particularly RNNs as LSTM networks, have emerged as powerful alternatives due to their ability to model sequential data effectively [1, 4]. In fact, their use for predicting stock trends has been a prevalent and evolving area within financial modeling due to the need for accurate forecasts in complex and volatile markets. LSTMs have demonstrated promising results in capturing sequential dependencies in financial time series data [1]. Furthermore, Kumar et al. [4] highlighted the potential of LSTM architectures, which have outperformed traditional models, such as Moving Average (MA) and XGBoost, in predicting stock prices, while comparative analyses highlight their superiority over RNNs and GRUs in managing temporal irregularities [6, 13]. Thus, these works relied the prediction just on DL generic time-series prediction solutions, but none of these works focused on analyzing the possible underlying investor behavioral patterns for explaining the predictions as the foundations of technical analysis for gaining trust of investor users, as the current work does.

Early advancements in LSTM-based stock prediction focused on integrating technical indicators and basic feature engineering to enhance predictive accuracy. For instance, Agrawal et al. [14] proposed an Evolutionary Deep Learning Approach (EDLA) based on stacked LSTM layers that utilize technical indicators such as moving averages and Relative Strength Index (RSI) for stock trend forecasting. They evaluated their approach using trading data from three Indian financial organizations over a two-year period. EDLA generates buy or sell decisions for investors. The authors trained the EDLA-LSTM model for 100 to 500 epochs and compared its accuracy against two baseline machine learning (ML) classifiers, such as logistic regression and Support Vector Machines (SVM). Their results demonstrated that EDLA outperformed these baseline ML approaches. Similarly, Istiake Sunny et al. [5] conducted a comparative study of LSTM and bidirectional LSTM (BiLSTM) models, highlighting the effectiveness of these models in enhancing prediction accuracy through proper hyperparameter tuning. They collected Google's stock price data from Yahoo Finance from 2004 to 2019 to compare the models, training both for 10 to 250 epochs. Their findings showed that the BiLSTM model outperformed the standard LSTM model when trained for more than 50 epochs. In a related vein, Li et al. [15] proposed a hybrid model integrating a Convolutional Neural Network (CNN) and LSTM with an attention mechanism. They evaluated it on two real-world stock datasets from 2017 to 2019, comparing its performance against Support Vector Regression (SVR), LSTM, and GRU models. The results indicated that their model outperformed the baseline models, particularly when a high number of parameters is required. These works developed models to predict stock prices or price directions, with LSTM serving as the core component. Typically, such models are fed with raw stock prices or technical indicators.

In a comparative analysis, Zhang and Pinsky [16] evaluated LSTM performance across S&P-500 and NASDAQ-100 indices using three distinct trading periods: overnight (close-to-open), daytime (open-to-close) and 24-h (close-to-close) cycles. They assessed the models from two perspectives: machine learning and trading, focusing on predictive accuracy as well as the profitability and practical utility of associated trading strategies. Opening and closing stock prices for these indices were collected from Yahoo Finance covering the period from 1999 to 2024. Preprocessing involved computing returns for each period and binarizing them into positive (+) and non-positive (-)

labels. The LSTM models were trained on these binary inputs using 10 epochs, lookback window of 10, and two hidden layers with 50 units each. Separate LSTMs were developed for each period, with training durations of 1 year, 6 months, and 3 months. The authors compared a short-term LSTM-based strategy against buy-and-hold strategy (B&H). Their results showed that the LSTM required one year of training data to outperform B&H for the S&P-500, whereas a shorter 3-month period for the NASDAQ-100 was sufficient.

Other efforts of researchers to improve LSTM efficiency included using dimension reduction methods, clustering, or a genetic algorithm (GA). For example, Lee et al. [17] combined dimension reduction methods such as Principal Component Analysis (PCA), Sparse PCA, Kernel PCA, Sliced Inverse Regression (SIR), Sparse SIR, and Kernel SIR with LSTM to achieve better prediction accuracy in predicting stock market indexes (S&P500, STOXX Europe 600, and Korea Stock Index - KOSPI). To demonstrate the efficiency of their approach, they applied it to three stock market indexes, S&P500, STOXX Europe 600, and KOSPI, covering the period from 2015 to 2021. They demonstrated superior performance of their approach compared to traditional linear time series methods. In the same vein, Li et al. [9] proposed a novel similarity measure for clustering integrated with LSTM, RNN, and GRU models to predict the stock prices. They evaluated its performance using a dataset that included full historical daily prices and volume information for stocks in the NYSE, NASDAQ, and NYSE MKT from 2005 to 2017. They demonstrated that this clustering approach improved the forecasting accuracy of the three deep learning models. Baek [18] proposed a hybrid model of CNN and LSTM integrating a GA to find a model suitable for predicting the stock market's next-day closing price. They collected data on the KOSPI index through Yahoo Finance, selecting daily stock price data from 2014 to 2022. They demonstrated that their hybrid model performed better than the single models of CNN and LSTM. Emerging architectures, such as extended LSTMs (xLSTMs), show promise for long-term forecasting, while nowcasting frameworks integrate DL for real-time applications. For example, Fan et al. [19] proposed a prediction model based on xLSTM to predict stock prices for short and long-term periods. They employed historical stock price data sourced from Yahoo Finance for Apple, Johnson & Johnson, and Nike companies from 2014 to 2013. Their results demonstrated that their model outperformed the traditional LSTM model in short-term and long-term stock price predictions. In a recent study, Fan and Zhang [10] compared several deep learning models such as LSTM, GRU, CNN-LSTM, and the Patch Time Series Transformer (PatchTST) for stock price prediction in forecasting and nowcasting scenarios. They demonstrated that these models performed better in nowcasting than in forecasting. Moreover, their findings revealed that the LSTM model achieved the best performance, surpassing the PatchTST and the hybrid CNN-LSTM models.

Ding and Qin [20] designed a multi-value associated neural network based on LSTM to predict multiple prices of a stock on the next day simultaneously such as the opening price, the lowest price, and the highest price. They evaluated their model using three data sets of historical data of Chinese stocks in different Chinese index, such as the Shanghai Index, PetroChina, and ZTE. Their results showed that the average accuracy of their proposed model is better than standalone LSTM and Associated Net models. Additionally, Zou et al. [21] introduced a Deep Reinforcement Learning (DRL) stock trading system based on a Cascaded Long Short-Term Memory–Proximal Policy Optimization model (CLSTM-PPO) to address the challenges of noisy financial data. The model used two LSTM stages: one for extracting time-series features from daily stock data and another for guiding the reinforcement learning agent. It was tested on major indices—the Dow Jones Industrial Average (DJI), Shanghai Stock Exchange 50 (SSE50), S&P Bombay Stock Exchange Sensex Index (SENSEX), and Financial Times Stock Exchange 100 (FTSE100)—and outperformed baseline strategies and models such as Multilayer Perceptron (MLP), standalone LSTM, and Light Gradient Boosting Machine (LGBM), improving key performance metrics by 5% to 52%, demonstrating its effectiveness for automated trading.

Nevertheless, unlike our proposed approach, all these approaches do not decompose stock price time series into different investors. In addition, these works only tested their approaches with accuracy measures, but our work tests our approach in a controlled simulated investor environment.

2.2 Hybrid ML approaches with LSTM for stock predictions

Other advancements have focused on hybrid and optimized LSTM variants to address limitations in LSTM models, such as overfitting and hyperparameter sensitivity. Yang et al. [22] proposed a

hybrid model for predicting the direction of price movements, combining CNN for feature extraction with LSTM for prediction. They evaluated their model using data from 11 influential international stock market indices, e.g., S&P 500, Nasdaq, NYSE, Dow Jones Industrial Average (DJIA), Russell, extracted from Yahoo Finance from 2010 to 2017. Performance was assessed via a confusion matrix and standard metrics, including precision, recall, accuracy, and F-measure. Their results showed that the model outperformed baselines such as SVM and CNN. In a similar vein, Lu et al. [23] introduced a hybrid model (CNN-BiLSTM-AM) framework that integrates CNN for feature extraction with BiLSTM and attention mechanism (AM) to predict next-day stock closing prices. They tested the model using trading data from the Shanghai Composite Index from 1991 to 2020. They trained their model for 100 epochs with the Adam optimizer. They compared its performance against baselines, including Multilayer Perceptron (MLP), CNN, RNN, LSTM, BiLSTM, CNN-LSTM, CNN-BiLSTM, and BiLSTM-AM. The results indicated that their model outperformed these baselines.

Other hybrid ensembles have combined GRU with LSTM to capture both short- and long-term dependencies [24, 25]. For instance, Farhadi et al. [25] presented a hybrid model (LSTM-GRU) to forecast stock prices by addressing both short- and long-range dependencies. They collected financial data for eight Iranian companies (e.g., Atlas, Kh-Motor, Gh-Minoo) from 2012 to 2024. The dataset included core features as open, high, low, close, and volume for each stock; during preprocessing, they calculated 42 additional features (e.g., Simple Moving Average, Moving Average, Moving Average Convergence Divergence, RSI, and BIAS). Dimensionality reduction was achieved using Random Forest (RF) to select the most relevant features, followed by Min-Max normalization. Hyperparameters were tuned, and the model was trained for 100 epochs using the Adam optimizer. The authors evaluated it against three different approaches: CNN, CNN-BiLSTM-AM, and LSTM, demonstrating that their hybrid model achieved a lower error rate than these alternatives. Similar results were reported by Karim et al. [24], who proposed a hybrid model integrating BiLSTM and GRU models for stock price prediction. They evaluated it against baselines including standalone LSTM, BiLSTM, GRU, and traditional neural networks, showing that the hybrid model achieved superior performance.

Building on such approaches, Mehtarizadeh et al. [26] proposed a hybrid model integrating the Sine-Cosine Algorithm (SCA) with LSTM, ARIMA, and GARCH models for forecasting stock closing prices. SCA was used to enhance LSTM performance, ARIMA to capture long-term trends, and GARCH to model seasonal data. This framework addresses the nonlinear, volatile, and non-stationary nature of stock market time series, not only predicting price directions but also providing a robust mechanism for noise and volatility management. The authors evaluated the model using closing stock price data from 12 companies listed on the NSE, NASDAQ, and NYSE (e.g., Oracle, Microsoft, Amazon) from 2009 to 2018. Training involved up to 500-1,200 epochs and used a neural network with 30-50 hidden units. Compared to other hybrid models, such as Hybrid Mutation Particle Swarm Optimization LSTM (AHMPSO-LSTM), Dynamic Linear Weighted Regression LSTM (DLWR-LSTM), and LSTM Neural Prophet (LSTM-NP), their model demonstrated superior predictive accuracy and robustness.

Optimization techniques have also proliferated in this domain. For example, Liu et al. [27] developed a Sparrow Search Algorithm-optimized LSTM (SSA-LSTM) for stock price prediction, employing SSA to identify optimal hyperparameters for the LSTM network. They evaluated the model with a dataset of Shanghai Composite Index stocks from 2010 to 2021. They compared it against baselines such as Back-propagation (BP), RNN, LSTM, and GRU models. This involved analyzing prediction curves, which demonstrated that SSA-LSTM achieved higher forecasting accuracy and effectiveness than the baselines.

The main gap of these approaches is the absence of explainability so that investors can trust the predictions in order to take investment decisions. On the contrary, the current approach explains the predictions based on the underlying possible behavioral patterns and shows the performance of trading strategies based on these estimations using a backtesting approach without forward knowledge, comparing it with vanilla LSTM as baseline.

2.3 Explainability and trading using stock predictions

A relevant group of works use sentiment analysis as the main source for explaining stock price movements, and in some occasions suggest investment decisions. In fact, the integration of external factors, such as sentiment analysis and sectoral correlations, represents a growing trend to enrich LSTM inputs, suggesting that the addition of external data sources can provide valuable insights and potentially enhance model performance [3, 28]. For example, Liagkouras and Metaxiotis [29] proposed a hybrid LSTM system combined with a sentiment analyzer for predicting the direction of the movement of the stock price. This system is fed with historical stock data and regulatory news announcements. In particular, they used a sentiment analysis model based on Naive Bayes to extract sentiment from the regulatory news announcements. This system provided a XOR combination of the outputs from the LSTM and sentiment analyzer, meaning that this system only generated a trading signal when the directional signals of the two components are in the same direction. They evaluated their model against other standalone models, such as Logistic Regression, LSTM, simple RNN, and SVM, using historical data from the AstraZeneca and Rio Tinto companies, covering the period from 2019 to 2021. They concluded that their model significantly outperformed the standalone models. In the same vein, Narayana et al. [30] developed a hybrid model integrating RNN architectures (LSTM, BiLSTM, GRU), Natural Language Processing (NLP) and Random Forest for sentiment analysis to forecast stock price and trading strategies. They developed two strategies to recommend selling or buying depending on a threshold of 0.5%. Thus, this model recommended to buy when the projected price is 0.5% or higher than the actual price. On contrary, sell strategy is recommended when the actual price surpassed the forecasted price by this threshold. They evaluated their model using stock data from ten companies across several industries using Yahoo Finance in a period from 2021 to 2023. Their results indicated that their model outperformed (average accuracy of 91.89%) the standalone models, such as BiLSTM, GRU, RNN, LSTM, XGBoost, and AdaBoost. Although decisions based on sentiment analysis of breaking news could potentially outperform the market as proven by these authors, that would require real-time trading for anticipating market movement. Moreover, Jin et al. [7] proposed a DL–based stock market prediction model integrating investors' emotional tendencies to enhance forecasting accuracy. It incorporated investor sentiment as an additional input, applied empirical mode decomposition (EMD) to handle multi-scale price fluctuations, and employed an attention-enhanced LSTM to capture critical temporal dependencies. Experimental results showed improved accuracy and reduced time delay. However, these approaches remained focused on direct price prediction, offering limited interpretability into the behavioral mechanisms behind market changes. On the contrary, our approach suggests an alternative augmented technical analysis for price movement prediction based on explanations of explicit amounts of different types of investor behaviors.

Explainable deep learning variants further incorporate textual analysis for interpretable forecasts [8]. They used SHapley Additive exPlanations (SHAP) for explaining the output of machine learning models. Saraswat and Kumar [31] advanced multimodal forecasting by developing a deep auto-optimized framework that embeds sectoral behavior, capturing industry-wide dynamics, into LSTM layers, which proved particularly effective for cross-sector predictions. This aligns with our previous work about DI2XAI [32], which also used SHAP as basis for explaining different trading/investment strategies based on different types of ML. However, our current approach provides a novel way of explanation based on prediction of time-series ratios of investment behavioral patterns.

Reinforcement learning (RL) extensions have further bridged prediction and actionable trading strategies. Huang et al. [33] outlined a multi-scale deep RL approach intertwined with LSTM-based price movement prediction, enabling adaptive trading policies that respond to varying temporal resolutions. Meanwhile, classification-oriented frameworks, like the robust ML pipeline by Alghamdi and Alqithami [34], employed LSTM for binary market state classification (bullish/bearish), focusing on risk-adjusted returns rather than point estimates. Other reinforcement extensions, like CNN-augmented LSTMs [35], further bridge prediction with adaptive strategies [36]. In contrast, our proposed method, implemented within DI2XAI, introduces a novel perspective by using LSTM to estimate ratios of investors across behavioral types rather than prices, enhancing explainability while preserving predictive accuracy and providing a more interpretable and domain-specific understanding of the collective dynamics driving stock market evolution.

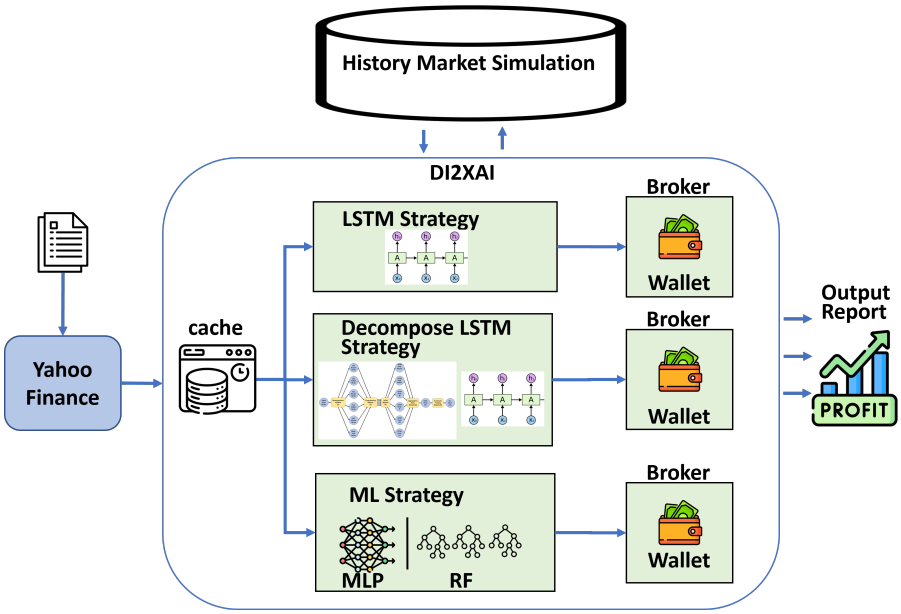

Fig. 1: Overview of the simulator including the proposed approach

3 Proposed LSTM approach for trading

The proposed LSTM-approach is implemented and integrated within the DI2XAI backtesting simulation environment. Although LSTMs have proven effective for predicting future stock prices, existing methods often perform poorly in forecasting short-term movement directions, due to the inherent difficulty of capturing short-term changes in volatile markets. However, this apparently unpredictability in price series lies in a logical explanation rooted in the collective behaviors of individual investors operating in the market. Reproducing this collective behavior does not require identifying which specific investors follow each strategy, but rather estimating the proportions of investors across behavioral categories.

Our approach introduces novelty by using LSTMs to predict these investor-type ratios rather than directly forecasting prices. This paradigm shift not only yields slightly higher accuracy compared to the most well-known LSTM-based alternative approaches, but also provides more insightful information about the explanation of the evolution of markets, getting to a new ad-hoc domain-specific way of explaining stock price series that diverges from all the alternatives in the literature.

Figure 1 illustrates the overview of DI2XAI (simulator of trading), which includes the LSTM decomposition strategy. Essentially, each strategy operates in the historical market data of stocks retrieved from Yahoo Finance, with a cache for reducing the requests to Yahoo servers, increasing the performance of the evaluation, and avoiding over passing the limit of requests to this external service. Each strategy operates with its own wallet administered by the broker inside our DI2XAI simulation tool. Each strategy has only access to the data up to the historic date of the simulation that goes forward in the simulation, so the outcomes are valid without using forward information. This architecture shows the three most relevant strategies, which are an LSTM baseline strategy, the enhanced LSTM strategy using decomposition, and a generic ML strategy that can be initialized with a Scikit-learn ML model (in this example, using a random forest). The output reports are useful for comparing the profits of different strategies in automated competitions in different historical periods.

Figure 2 illustrates the architecture of our approach. Modules are depicted as rectangles, while input and output data are represented as circles. The process commences with the decomposition module, which receives the raw input price series and generates five output series: individual buyers, individual sellers, institutional buyers (IB), institutional sellers (IS), and stop-loss (SL). Next, the preprocessing module processes each series applying a normalization. This module connects to the LSTM module, which produces the unnormalized predictions for the proportions of individual buyers and sellers, IB, IS, and SL. The aggregation rules module then receives these predicted

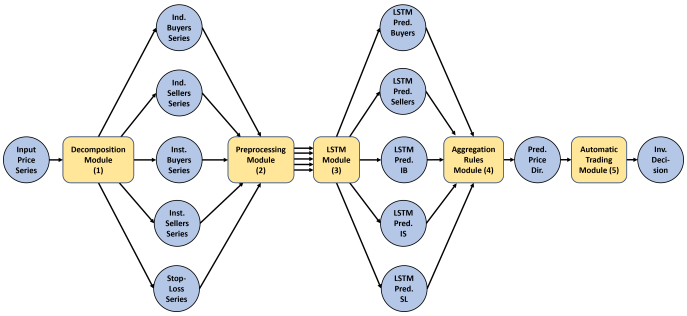

Fig. 2: Architecture for predicting series using different investors profiles

proportions and employs GA to derive the predicted price direction. Finally, the last module generates the investment decisions (buy, sell, or hold) based on the predicted price direction.

3.1 Decomposition of investor types for prediction

The decomposition of the existing information of a stock series into estimated series of the different investor types is the cornerstone of our approach, providing insight explained information of what might be occurring. In our work, this decomposition has shown to be a useful tool for enhancing the information of directional trends of markets in order to establish a more profitable trading strategy.

In essence, this estimation is based on common knowledge about usual behaviors of the different investor types reflected in the literature, like in [37]. More concretely, when the price increases slowly, we estimate more buyers than sellers. By contrast, when the price decreases slowly, we estimate more sellers than buyers. When the price drops abruptly, we estimate more stop-loss automated selling investors.

When there is a high volume in a very short time, and prices usually move across a support or resistance, we assume institutional investors are moving the price on purpose for later operating for their benefit. If breaking a support, institutional investors sell together many shares, but only a small proportion of their capital, so later they can buy at a low price. If breaking a resistance, they buy to increase the price, and they already have a high proportion already bought, so later they can sell slowly at a high price.

The input price series will be transformed into the following series: (a) requests from individual sellers, (b) requests from individual buyers, (c) institutional investors buying slowly for later benefiting, (d) institutional investors selling slowly for collecting benefits, and (e) sellers with automated stop-loss.

The steps for this estimation of the amounts of each of the roles follow: (1) initial proportion configuration of all the roles are established with equally amounts of buyers and sellers in each of the categories, and a small amount of stop-loss operations, (2) the proportions of buying actors (both individual and institutional) increase by summing a weight multiplied by the percentage increase of the price from the previous day, (3) similarly the proportions of selling actors are modified but subtracting the corresponding increase of daily prices (i.e. in this way a price decrease implies an increase of selling proportion), (4) in price drops over a certain threshold, stop-loss proportion is increased by a weight multiplied by the corresponding price decrease, (5) when a manipulation is estimated when a market movement has higher volumes than usual, proportions of institutional roles are increased multiplied by certain weights (separately for institutional seller series and institutional buyer series), and (6) the amounts of all the roles are calculated by multiplying the proportions by the volume of the days.

For the sake of reproducibility, the following equation describes the final formulas with the specific values. First, equation 1 initializes programming variables representing ratios as percentages (i.e. in "[0, 100]" interval) for respectively amount of money operated by buyers (b), sellers (s), IB (b_i), IS (s_i) and SL actions (s_L), in the given series, with the corresponding constants named B, S, B_i, S_i, S_L for initialization.

$$b = B; s = S; b_i = B_i; s_i = S_i; s_L = S_L \tag{1}$$

Then, the buyers, sellers, IB and IS are updated considering the increase of price (referred as i) and certain weights for estimating the relation of price with these roles, respectively named w_b, w_s, w_{bi}, w_{si} with equation 2 and different weights.

$$b = b + w_b * i * b; s = s + w_s * i * s; b_i = b_i + w_{bi} * i * b_i; s_i = s_i + w_{si} * i * s_i; \tag{2}$$

Equation 3 calculates stop-loss actions firstly considering price negative movements multiplied by a weight for SL named w_{sL}. When the increase is positive, SL is reduced proportionally with a minimum of zero (using max function for such purpose, and when the increase is negative, SL increases proportionally. Additionally, these estimations are multiplied by the weight for drops named w_d, when detecting massive sells due to probably chained stop-loss actions (estimated when absolute value of the decrease is greater than the threshold of drops named t_d). Negative movements are obtained by the price increase (already referred to as i) with a negative sign (i.e. $-i$). For identifying substantial drops, the corresponding threshold t_d is multiplied by the volatility measured as the daily close price standard deviation (SD) over the last month.

$$s_L = \begin{cases} w_d * max(s_L - w_{sL} * i, 0), & \text{if } i < (-t_d * SD). \\ max(s_L - w_{sL} * i, 0), & \text{otherwise.} \end{cases} \tag{3}$$

If the current volume (v_c) is higher than a threshold (t_v) multiplied by average volume (v_a) in the last month, then institutional roles are increased with a certain weight representing the manipulation(w_M) multiplied by the price increase (i), only incrementing the role between IB or IS depending on the direction of the increase as determined by equations 4 and 5.

$$b_i = \begin{cases} b_i + b_i * w_M * i, & \text{if } v_c > t_v * v_a \text{ and } i > 0 \\ b_i, & \text{otherwise} \end{cases} \tag{4}$$

$$s_i = \begin{cases} s_i + s_i * w_M * i, & \text{if } v_c > t_v * v_a \text{ and } i < 0 \\ s_i, & \text{otherwise} \end{cases} \tag{5}$$

The ratios of money amounts operated as requests by each role are calculated by normalizing the percentages of each behavioral type with equation 6, obtaining the normalized ratios of buyers (b'), sellers (s'), IB (b'_i), IS (s'_i), and SL (s'_L).

$$\begin{aligned} b' = \frac{b}{b + s + b_i + s_i + s_L}; s' = \frac{s}{b + s + b_i + s_i + s_L}; b'_i = \frac{b_i}{b + s + b_i + s_i + s_L}; \\ s'_i = \frac{s_i}{b + s + b_i + s_i + s_L}; s'_L = \frac{s_L}{b + s + b_i + s_i + s_L}; \end{aligned} \tag{6}$$

Then actual estimated volumes of operation requests for respectively buyers (b_v), sellers (s_v), IB (b_{iv}), IS (s_{iv}) and SL (s_{Lv}) are obtained by multiplying the normalized ratios of each role (b', s', b'_i, s'_i, s'_L) by the current volume (v_c) with equation 7.

$$b_v = b' * v_c; s_v = s' * v_c; b_{iv} = b'_i * v_c; s_{iv} = s'_i * v_c; s_{Lv} = s'_L * v_c \tag{7}$$

Figure 3 shows an example of decomposition. This decomposition shows an estimation of the volume of operation requests over shares of each role. It uses the NVIDIA company as an example from 21st October 2024 to 21st October 2025. Notice that when the price increases, this is reflected with more buyers than sellers, and the other way around. The volume of requests is related to the number of shares, and consequently, institutional operations can have a high volume even if only one or very few institutions are manipulating the market.

Several LSTM networks individually predict each of the future series of the established roles. The predicted series are converted into movement of prices considering certain rules. A genetic algorithm (GA) obtains the best weights for estimating behavioral roles that make trading strategies most effective.

3.2 Genetic algorithm

The GA is focused on determining the best way of performing the decomposition from technical analysis into behavioral roles. In the GA, the population will represent technical analysts that explains the investor/trader behaviors behind the price series analysis. Each analyst individual

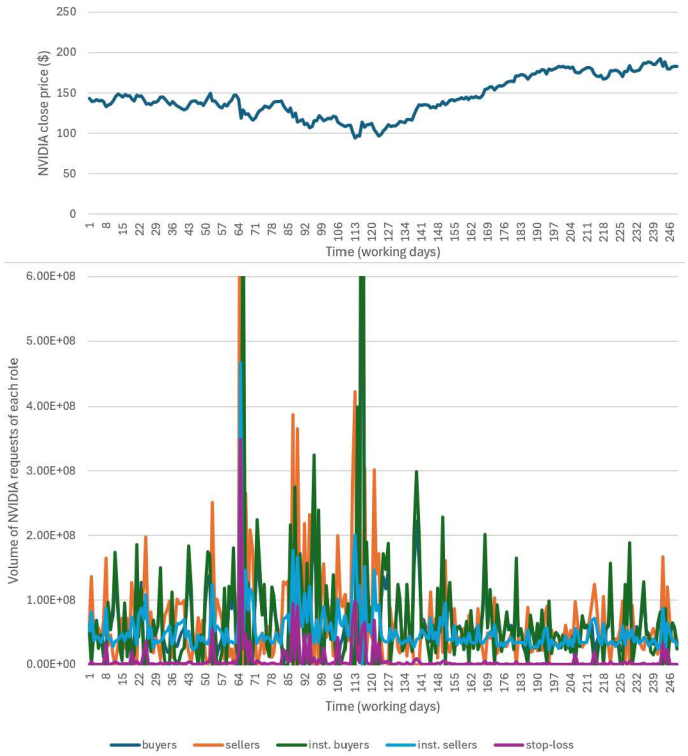

Fig. 3: Example of decomposition

Initialization					Price Increases				Price Drops		Manipulations	
B	S	B_i	S_i	S_L	w_b	w_s	w_{bi}	s_{si}	w_{sL}	w_d	t_v	w_M

Table 1: Chromosome for technical analyst individuals for the GA

will have a specific decomposition mechanism into behavioral roles defined with the weights and thresholds of table 1 defined in the previous section. Each chromosome is conformed of the specific values of these weights and thresholds for each individual, and will be used for the different phases of the GA, such as cross-over, mutation, and evaluation.

After this, the estimation of the amounts of each of the roles, including individual buyers and sellers, institutional buyers and sellers, and stop-loss operations are calculated with certain weights established with the GA. After running the GA, the obtained weights are included in the simulator to be run in a standalone way, without needing to run the costly GA again.

Figure 4 shows the process of generating the best individuals using a GA. The chromosome of each individual (depicted as an array) represents the weights applied to each individual. The cross-over is defined by randomly selecting which positions are swapped of the weights and swapping these positions among both individuals with a certain probability of swapping (p_s) using a random value (referred as r_d from the metaphor of rolling a dice). The swapping of the weights of two individuals is performed by swapping with equation 8.

$$w_i, v_i = \begin{cases} v_i, w_i, & \text{if } r_d < p_s \\ w_i, v_i, & \text{otherwise} \end{cases} \tag{8}$$

After the swapping of the cross-over, the weights (w_i) in the category of initialization are normalized (w'_i) so that the proportion is kept and the sum is one hundred with equation 9.

$$w'_i = \frac{100 \cdot w_i}{sum_{j=0}^{N} w_j} \tag{9}$$

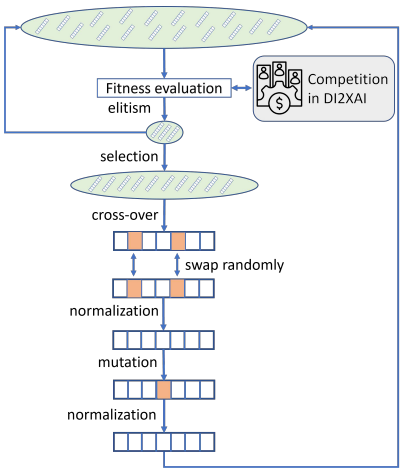

Fig. 4: Main aspects of the GA design

Parameter	Value
Probability for selecting individuals for cross-over (%) (p_c)	30%
Swapping probability (p_s) (%)	50%
Mutation probability for genes (p_m) (%)	0.72%
Number of individuals for elitism (n_e)(#)	3

Table 2: Genetic parameter values

In order to define which individuals are used for cross-over, the individuals are randomly sorted, and the pairs are formed in order.

For mutation, we used a probability of mutation (p_m), in which each weight of the chromosome was modified with a random weight (r_w) in the interval [0,1] if a random value (r_d) was below the probability value with equation 10. After the mutation, the weights are normalized again with the previously introduced equation 9.

$$w_i = \begin{cases} r_w, & \text{if } r_d < p_m \\ w_i, & \text{otherwise} \end{cases} \tag{10}$$

The individuals are evaluated by competing against each other in trading, and only the first N individuals survive. The number of individuals selected by elitism n_e, which are best ranked, are separated and involved in neither the cross-over nor the mutation operations. Table 2 shows the parameters that were used for applying the genetic algorithm.

3.3 Trading LSTM approach with decomposed roles indicators

The trading algorithm compares the behavioral amounts of investor roles outputted by the LSTMs to implicitly predict an increase or decrease in price, and then the trading algorithm buys shares in the analyzed stock using these predictions. After this, it holds the investment until reaching a certain profit margin above the expectations, with a formula that decreases the margin when more time has elapsed. It establishes a profit for a given year (or any other number of years), and when this target is reached, it sells and waits for several sessions. In abrupt LSTM predictions of decreasing trends, the algorithm switches on a flag of selling, and from this point forward it establishes selling prices according to what is reasonable given the current price and the current volatility.

The strategy starts by buying and later focuses on when to sell and buy, regarding the particular conditions and predictions. Figure 5 shows the most relevant excerpt of the "advance" method that is executed daily in the history. The "hasInvestment" variable determines whether the trading strategy has the investment on the market, or otherwise it has liquid money to invest. When the strategy has invested the capital, it should determine exit points where to buy.

```
if hasInvestment and buyers>sellers and price > self.buyPrice*(1+self.targetProfitPercentage/100):
    self.broker.sellAll()
elif not hasInvestment and (predBuyers+predInstBuyers)>(predSellers+predInstSellers+predStopLoss):
    self.broker.buyAmount(self.broker.liquid)
    self.buyPrice = self.evol.getPrice()
```

Fig. 5: Most relevant excerpt of advance method of the strategy with LSTM and decomposed roles

One of the main conditions is hitting a target, measured by calculating whether the current price is higher than the buying price plus another amount established by the target profit percentage. Another required condition is that the number of buyers is greater than the number of sellers. These are obtained from the current indicators. Notice that this equation leaves out considering manipulated market movement (institutional investors) and automated stop-losses that could also be a result of part of the manipulation. It is worth noting that it uses current information rather than predicted information, since this strategy sells when the sentiment of individual investors is positive, reflected in higher numbers of buying-shares operations than in selling-shares operations. In this way, this strategy sells when a profitable target is hit and the sentiment of investors is positive, so the selling operations are not delayed and the profits are not reduced because of waiting for some changes in future predictions. In other words, selling decisions are taken based on the current situation.

By contrast, in this strategy, buying operations are decided according to predictive future expectations. For this purpose, the strategy uses the predictions of the five different roles (i.e., buyers, sellers, institutional buyers, institutional sellers, and stop-loss operations), represented respectively in the variables "predBuyers", "predSellers", "predInstBuyers", "predInstSellers", and "predStopLoss". In order to understand this strategy, it is worth being aware that each of these variables determines the intentions of performing certain market operations, counted as the sum of the number of shares, so that the number of actual actors is not relevant in the calculus. Each of these predictions is calculated with a separate LSTM, trained separately. In case of having liquid money, this strategy decides to buy shares, if the total amount of intended buying shares (including both the predicted individual and institutional buyers) is greater than the total selling operations (including the predicted individual and institutional sellers plus the predicted automated stop-losses). If these predicted expectations are met, then the price will rise, probably resulting in profitable trading operations (requiring at least a certain profitable target percentage).

3.4 LSTM architecture

Each LSTM is designed to forecast the next timestep value of a specific behavioral ratio series (e.g., individual buyers or institutional sellers), derived from the decomposition process described in Section 3.1. This univariate forecasting setup aligns with the decomposed nature of our inputs, where each series represents an isolated behavioral component, enabling learning for distinct investor archetypes at the prediction stage.

The architecture begins with data pre-processing. Raw input series, comprising historical volumes of behavioral operations, are normalized using the MinMaxScaler from scikit-learn, transforming values to the [0, 1] range. This transformation mitigates the impact of different scales across behavioral components (e.g., high volume institutional trades versus sporadic individual actions) and stabilizes gradient flow during training, a common challenge in LSTM optimization for financial data [1]. It is important to note that normalization is fitted solely on the training set to prevent data leakage.

Subsequently, the normalized series are converted into supervised learning datasets using a sliding window approach (lookback period, L). This L governs how much data is allowed to look at when is making a prediction. For a given series of length N, we construct input-output pairs where each input sequence X_t consists of L consecutive timesteps ($L = 5$ in our experiments), and the target y_t is the subsequent timestep value. This formulation yields tensors of shape $(N - L, L, 1)$ for inputs and $(N - L, 1)$ for targets, capturing short-term temporal dependencies typical in daily stock dynamics. The choice of $L = 5$ balances computational efficiency with the need to encode recent momentum, as longer windows risk diluting signal in volatile markets while shorter ones may overlook autocorrelation [38].

The core model, illustrated in Figure 2, is an LSTM implemented using a PyTorch LSTM model. It is configured with an input size of 1 (univariate behavioral ratio), a hidden size of 50, and a variable number of recurrent layers (set to 5, as detailed below). The hidden size of 50 was empirically determined through hyperparameter tuning on validation sets, providing sufficient representational capacity to capture nonlinear patterns in behavioral ratios, such as sudden spikes in stop-loss activations, avoidance of overfitting, and utilizing train and test root mean squared error (RMSE) during cross-validation. A linear projection layer follows the LSTM output, mapping the final hidden state sequence back to a scalar prediction. The LSTM models employed in our framework are implemented using Python (version 3.11.7) and PyTorch (version 2.4.0), selected for their robust support of dynamic computation and efficient tensor operations, which facilitate handling of sequential data in financial time series [39].

The selection of the LSTM architecture over more complex contemporary models, such as Transformers or ultra-deep neural networks, is grounded in a deliberate trade-off between data efficiency, overfitting mitigation, and computational latency. Given that our decomposed behavioral series are strictly univariate and our training windows are constrained to recent historical periods (500 timesteps) to capture current market regimes, highly parameterized models like Transformers are prone to memorizing noise rather than extracting robust sequential patterns in such limited datasets. Furthermore, the DI2XAI multi-agent simulator requires real-time, low-latency inferences for thousands of simulated trading decisions. The chosen LSTM configuration provides the optimal balance, effectively capturing the short-term temporal dependencies of investor behavior without incurring the heavy computational overhead and data hunger characteristic of attention-based architectures.

To determine optimal hyperparameters, particularly the number of layers and hidden size, the data set was split into training (80%), validation (10%), and test (10%) sets. Data were sourced from the DI2XAI simulator, which generates stock closing price series starting from the first time the stock appears in market, and ending to a random date. Tickers were chosen to ensure sectoral diversification: AAPL (Apple), RACE (Ferrari), ITX.MC (Inditex), and JNJ (Johnson & Johnson, J&J). Given the sequential nature of LSTM models, each dataset comprised 500 historical data points (approximately two years) preceding the randomly selected ending date. In this study, the random date was February 18, 2021. Accordingly, the first 400 points were used for training, the subsequent 50 values for validation, and the final 50 for testing. Training was conducted using the Adam optimizer with a learning rate of 10^{-3} and RMSE as the loss function, over 100 epochs and a batch size of 8.

We evaluated training and validation RMSE for models with 1 to 5 recurrent layers, alongside a grid search for the optimal dropout rate, evaluating values in $\{0.0, 0.15, 0.30, 0.45, 0.60\}$. Figure 6 summarizes the tuning results for the AAPL and ITX.MC tickers, as a representative example, from February 2019 to 2021. This figure reveals minimal differences in validation RMSE across layer counts (less than 0.05). On average, the lowest RMSE was achieved with 5 layers and a dropout rate of 0.15, which was therefore adopted for the final model.

The final model was trained using the Adam optimizer with a learning rate of 10^{-3}, RMSE as the loss function, five recurrent layers, and 100 epochs with a batch size of 8. To mitigate overfitting, particularly important given the limited historical data (500 time steps per series), a dropout rate of 0.15 was applied within the LSTM layers. The model outputs were inverse-transformed using the fitted scaler and subsequently aggregated using GA optimized rules to reconstruct the predicted price movements, as detailed in Subsection 3.

This architecture prioritizes interpretability and low latency for integration into the DI2XAI simulator, where real-time behavioral forecasting informs multi-agent trading decisions.

3.4.1 Baseline Models and Hyperparameter Tuning

To validate the efficacy of the proposed LSTM network and the behavioral decomposition approach, a comprehensive computational competition was conducted. The standard LSTM model was benchmarked against alternative architectures, encompassing hybrid Deep Learning models and decision-tree-based Machine Learning algorithms.

To ensure a rigorous and fair comparison, all baseline models were subjected to systematic hyperparameter optimization to prevent overfitting and maximize generalization on unseen financial data. For the deep learning models (Standard LSTM, CNN-LSTM, and CNN-BiLSTM),

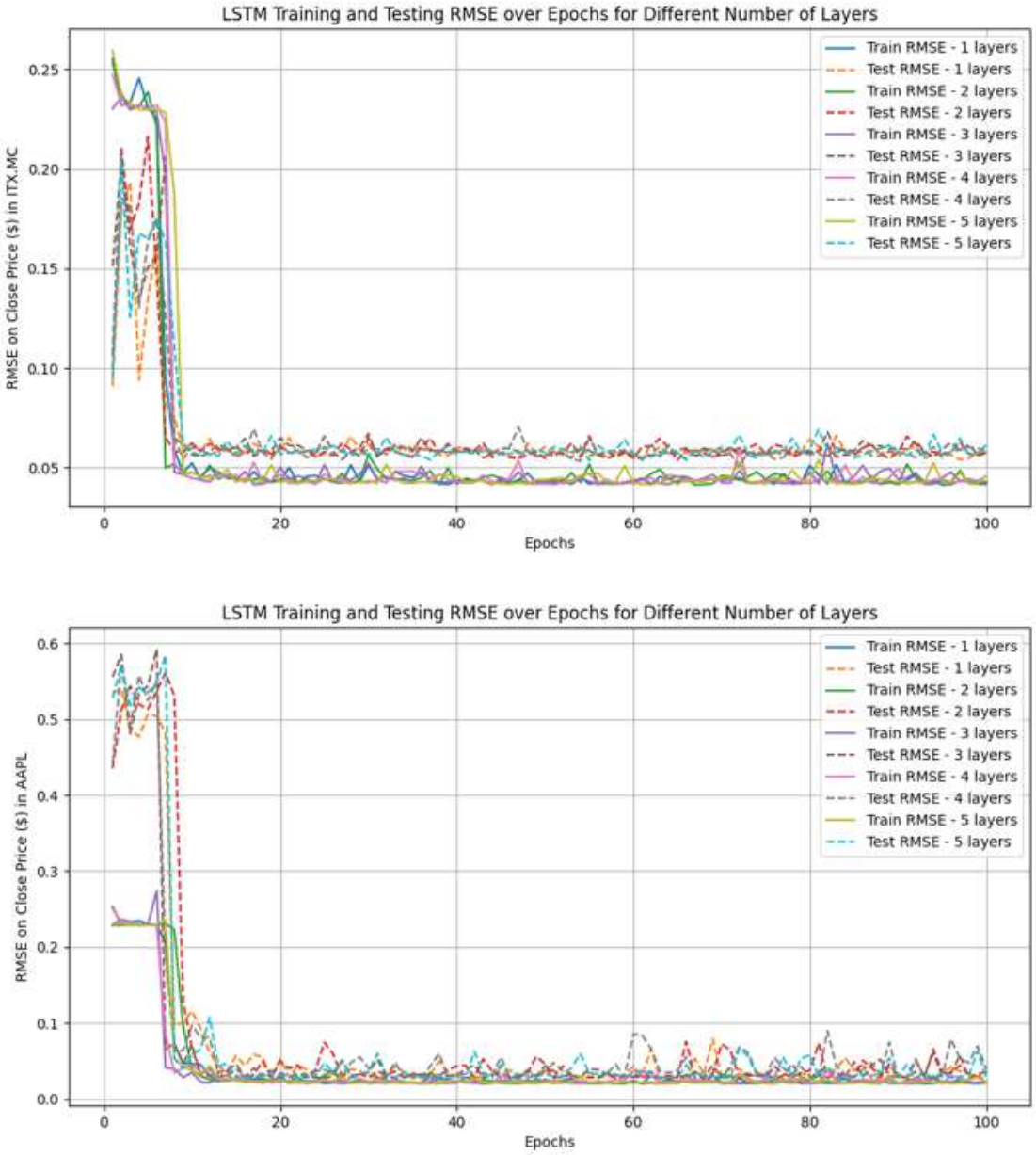

Fig. 6: Evolution of RMSE in LSTM with different layers

empirical tuning involved an 80-10-10 split (training, validation, testing) over 100 epochs. Specifically, the hybrid models (implemented via Keras) explored 1 to 2 recurrent layers, maintaining a hidden size of 32, a kernel size of 4, a lookback window of 10 periods, and recurrent dropout rates ranging up to 0.6 to penalize co-adaptation of features.

Conversely, the tree-based ensemble algorithms were optimized using a 5-fold cross-validation approach implemented via *GridSearchCV*. The search space and optimal configurations are summarized in Table 3.

The experimental setup included a scenario where all predictive models operated under a dynamic stock-picking strategy (*PickGrowth*), alongside a static index-tracking benchmark where XGBoost, Random Forest, and DecomposeLSTM traded a representative ETF (EUNL.DE). The final predictive outputs of the evaluated models determined the trading execution logic (buy/sell triggers) within the simulated environment.

Table 3: Hyperparameter Search Space and Optimal Configurations for Ensemble Models

Model	Hyperparameter	Search Space	Optimal Value
XGBoost	*n_estimators*	[10, 100, 200, 300]	100
	max_depth	[3, 5, 7, 10]	5
	learning_rate	[0.01, 0.1, 0.2, 0.3]	0.3
	reg_lambda	[0.1, 0.2, 0.3, 0.4, 0.5]	0.4
Random Forest	*n_estimators*	[10, 100, 200, 300]	100
	max_depth	[3, 5, 7, 10]	10
	min_samples_split	[1, 3, 5, 7]	5
	min_samples_leaf	[1, 5, 9, 13]	1



Fig. 7: Prediction of stock close price with an LSTM

4 Experimentation

The proposed LSTM-based approach with behavioral decomposition was implemented and evaluated within the DI2XAI backtesting simulation environment, which enables realistic multi-agent trading competitions using historical Yahoo Finance data. For the sake of reproducibility, we have made available the DI2XAI code with the used strategies in the Mendeley Dataset of DI2XAI[1]. We have included both "NpcLstm.py" (baseline LSTM used for trading) and "NpcDecomposeLstm.py" (novel strategy using investor role estimation decomposition) in the "python" folder. It is worth noting that the decomposition is included inside the file "indicators.py" as the new investor role type estimations are treated as any other indicator, so that it can benefit from reusing the software of the simulator.

While the final trading competition and Genetic Algorithm optimization were exclusively conducted on the four primary tickers (AAPL, RACE, ITX.MC, and JNJ), the initial model training and evaluation (e.g., RMSE metrics) encompassed a broader set of assets, including Amazon, Eli Lilly, NVIDIA, and Allianz. This broader evaluation phase was designed to demonstrate the base predictive model robustness and generalization capabilities across diverse market sectors.

In order to highlight the improvement of our novel interpretable LSTM-based trading strategy, firstly, we evaluate the standard LSTM strategy focused just on the time series of prices aligned with common approaches in the literature [16] that we used as a baseline, and then we compared our novel approach using LSTMs for predicting future investor role types to conform a trading strategy.

To begin with, we assessed the baseline LSTM model for direct closing price prediction. Figure 7 illustrates an example prediction for Amazon stock prices. One can observe that the testing window is working relatively well, except for an unexpected increase, while in the training segment, it works better. Thus, there is probably some overfitting, but considering the well-known unpredictability of stock series, we consider that this mismatch is within the common range in the field.

[1]García-Magariño, Iván; Bravo-Agapito, Javier (2026), "DI2XAI: Dashboard for Intelligent Investment with eXplainable AI", Mendeley Data, V1, doi: http://dx.doi.org/10.17632/thczvtw68x.1

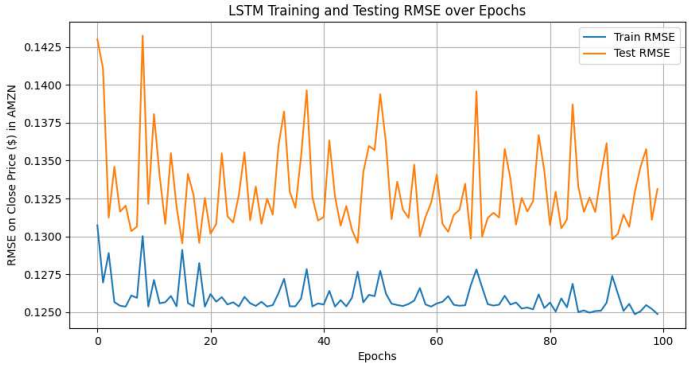

Fig. 8: Evolution of RMSE in the LSTM training

Table 4: RMSE values for the standard LSTM model across different companies (averaged over 20 executions).

Company	RMSE (train)	RMSE (test)
Amazon	0.0220	0.0438
Apple	0.0186	0.0304
Eli Lilly	0.0484	0.0294
Ferrari	0.0287	0.0312
Inditex	0.0361	0.0313
J&J	0.0500	0.0607
NVIDIA	0.0127	0.0390

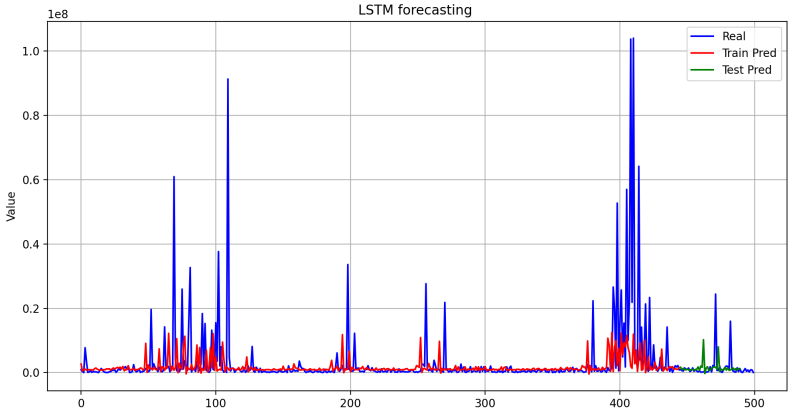

Fig. 9: Prediction of stop-loss operations with an LSTM

Figure 8 shows the evolution of RMSE in the normalized prices (using the MinMaxScaler method) during the training phase of the LSTM baseline. There is a difference between the RMSE on training and testing, with the latter being higher, confirming overfitting within a reasonable range, since the error is between 0.10 and 0.18 from 20 epochs onward.

In order to have a more representative sample, table 4 reports the average RMSE values for training and test sets across seven companies from diverse sectors in the LSTM baseline. The results indicate reasonable predictive performance, with test RMSE values generally remaining low and consistent across tickers, although some variability is observed due to differing volatility levels. It is worth mentioning that two companies even obtained lower RMSE in testing than training, so there was clearly no overfitting in these two companies (Eli Lilly and Inditex), and in another company (i.e., Ferrari), the error was almost the same in training and testing. Thus, the LSTM model baseline had no overfitting in almost half of the companies.

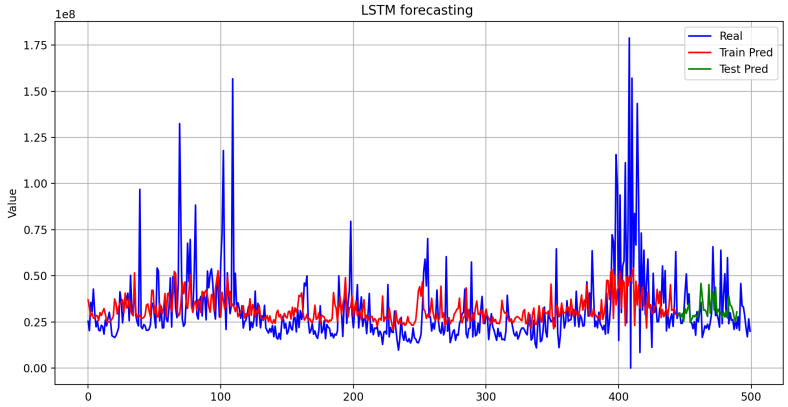

Fig. 10: Prediction of institutional-seller operations with an LSTM

Table 5: RMSE values for the LSTM decomposition model across different companies

Company	Inst. Buyers		Inst. Sellers		Stop Loss		Buyers		Sellers	
	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
Amazon	0.0651	0.0212	0.0924	0.0500	0.0831	0.0163	0.0650	0.0240	0.0903	0.0356
Apple	0.0985	0.0418	0.1026	0.0501	0.0930	0.0218	0.1008	0.0457	0.1035	0.0487
Eli Lilly	0.0884	0.0993	0.0671	0.0279	0.1064	0.0660	0.0974	0.1043	0.0762	0.0379
Ferrari	0.1167	0.0574	0.1067	0.1339	0.0948	0.0906	0.1154	0.0593	0.1050	0.1260
Inditex	0.0883	0.0721	0.0576	0.1155	0.0399	0.1512	0.1011	0.1115	0.0510	0.1234
J&J	0.0946	0.0541	0.0980	0.0827	0.0585	0.0261	0.0954	0.0640	0.0975	0.0753
NVIDIA	0.0736	0.0485	0.0765	0.1130	0.0681	0.1274	0.0730	0.0523	0.0746	0.1247
Mean	0.0893	0.0563	0.0858	0.0819	0.0777	0.0713	0.0926	0.0659	0.0854	0.0817

Next, we tested the performance for evaluating certain indicators related to the novel decomposition in estimated investor roles, in order to evaluate the proposed novel approach available for designing trading strategies. Figure 9 exemplifies predictions for stop-loss operations with LSTM compared to the estimated values obtained from the rule-based approach using GA to obtain the best weights in the corresponding formulas. One can observe that the timing of increases of stop-loss predictions was appropriate, although the scaling might not be completely appropriate, maybe because of the influence of any outlier or the fitness used in the algorithm, in which the importance was the comparison of operation requests rather than the absolute values. The timing was appropriate in both the training sequence and test sequence, revealing that the proposed LSTM model for this type of investor operations had no overfitting.

To show further insights, Figure 10 shows the prediction of selling operations by institutional investors' role compared to the rule-based estimated ones. One can observe that the timing again was properly predicted in both training and test sequences, although the scaling was not completely similar. Hence, the corresponding LSTM model neither had overfitting for this type of investor role.

Nevertheless, in both cases, since the main purpose was to determine the appropriate buying or selling momentum, this information was equally useful. Table 5 presents the RMSE values for training and test sets across the same seven companies for each of the investor roles. The errors remained comparable between training and test sets. On some occasions, test errors were slightly higher than training errors, while on others were the other way around. Thus, there was no substantial overfitting. When observing the average RMSE in the test phase (the most relevant phase for being trained with different values from the ones predicted) for the seven companies of each investor role, the lowest RMSE was obtained in buyers and institutional buyers, while the highest RMSE for the test phase was obtained in sellers and institutional sellers. Thus, buying investor roles were more predictable than selling investor roles in our framework according to our experiments. The stop-loss investor role had an intermediate predictability according to the intermediate

average RMSE. On the contrary, in the training phase, the results were more homogeneous, probably because the convergence criterion coincided since the same type of LSTM model was used in all the series of investor role types.

4.1 Baseline Model and Comparisions

To evaluate the trading efficacy of the proposed framework against both passive market indexing and state-of-the-art predictive architectures, a multi-agent competition comprising $N = 100$ simulations was conducted within the DI2XAI environment. All strategies were initialized with a starting capital of 100,000 monetary units and evaluated under identical historical conditions. To contrast active algorithmic stock selection (*PickGrowth*) against passive market exposure, a tuned Random Forest model operating on a representative global equity index tracker (*ETF EUNL.DE*) was established as the baseline model. The evaluated models included:

- Deep Learning Models: Hybrid CNN-LSTM and CNN-BiLSTM architectures (utilizing a kernel size of 4 and a 32-neuron hidden layer, with a lookback of 10), alongside our DecomposeLSTM proposed model.
- Ensemble Models: XGBoost (*n_estimators*=100, *max_depth*=5, *learning_rate*=0.3, *reg_lambda*=0.4) and Random Forest (*n_estimators*=100, *max_depth*=10, *min_samples_split*=5).
- Market Benchmark: A representative ETF (EUNL.DE) traded using Random Forest to contrast algorithmic stock picking against a static index approach.

To ensure a rigorous and fair comparison, all baseline models were systematically optimized using their respective hyperparameter tuning protocols. For the Keras-based hybrid architectures (CNN-LSTM and CNN-BiLSTM), hyperparameter tuning explored the number of recurrent layers (ranging from 1 to 2) and kernel sizes, leading to an optimal configuration of 1 recurrent layer, a kernel size of 4, a 32-neuron hidden size, and a dropout of 0.15 to prevent overfitting on the expanded 10-day lookback input sequences.

For the ensemble Machine Learning models, an exhaustive *GridSearchCV* framework was implemented employing a 5-fold cross-validation (*cv=5*). The XGBoost tuning process explored a broad parameter space: number of estimators from 10 to 300, maximum tree depth from 3 to 10, learning rate from 0.01 to 0.3, and L2 regularization (*reg_lambda*) from 0.1 to 0.5. The grid search identified the best-performing configuration as *n_estimators*=100, *max_depth*=5, *learning_rate*=0.3, and *reg_lambda*=0.4. A parallel optimization approach for the Random Forest algorithm determined its optimal hyperparameters at *n_estimators*=100, *max_depth*=10, *min_samples_leaf*=1, and *min_samples_split*=5.

As summarized in Table 6, *DecomposeLSTM* achieved the highest average final capital (266,572), more than doubling the return of the passive market baseline (125,778) and outperforming all active Deep Learning and ensemble models.

Table 6: Performance comparison of predictive models based on average final capital ($N = 100$ simulations, initial capital: 100,000).

Model	Selection Strategy	Average Final Capital
Baseline (Random Forest)	ETF (EUNL.DE)	125,778
DecomposeLSTM	PickGrowth	266,572
XGBoost	PickGrowth	236,995
CNN-LSTM	PickGrowth	218,268
CNN-BiLSTM	PickGrowth	212,331
Standard LSTM	PickGrowth	109,920

To formally assess whether these performance differences are statistically significant, a two-stage non-parametric hypothesis testing protocol was adopted. First, Shapiro-Wilk normality tests confirmed that the final capital distributions of the evaluated strategies deviate significantly from normality ($p \leq 0.007$), exhibiting the skewness and heavy tails characteristic of algorithmic trading returns. Consequently, a global Friedman test for related samples was conducted across all six models, revealing highly significant differences among their performance distributions ($p < 0.001$).

Following the significant Friedman test, five pairwise Wilcoxon signed-rank tests were performed comparing *DecomposeLSTM (PickGrowth)* against each competing model, applying the Holm-Bonferroni correction to control the family-wise error rate across multiple comparisons. Table 7 reports the standardized Z-statistics, unadjusted p-values, and Holm-Bonferroni adjusted p-values (p_{adj}).

Table 7: Pairwise Wilcoxon signed-rank tests comparing *DecomposeLSTM (PickGrowth)* against baseline and competing models ($N = 100$), with Holm-Bonferroni adjustment.

Competing Model (Strategy)	**Z-statistic**	**Raw p-value**	**Adjusted p_{adj}**	**Sig. ($\alpha = 0.05$)**
Baseline Random Forest (ETF EUNL.DE)	-7.241	< 0.001	< 0.001	Yes
XGBoost (PickGrowth)	-0.464	0.643	0.643	No
CNN-LSTM (PickGrowth)	-3.648	< 0.001	< 0.001	Yes
CNN-BiLSTM (PickGrowth)	-4.102	< 0.001	< 0.001	Yes
Standard LSTM (PickGrowth)	-7.654	< 0.001	< 0.001	Yes

The post-hoc analysis confirms that *DecomposeLSTM* achieves statistically significant improvements ($p_{adj} < 0.001$) over the baseline model as well as all evaluated neural architectures (*Standard LSTM*, *CNN-LSTM*, and *CNN-BiLSTM*). When compared against *XGBoost (PickGrowth)*, the Wilcoxon test indicates no statistically significant difference in median rank performance ($Z = -0.464$, $p_{adj} = 0.643$), demonstrating that both approaches exhibit comparable consistency across simulations. However, *DecomposeLSTM* achieves a 12.5% higher average final capital (266,572 vs. 236,995) while providing intrinsic behavioral explainability through investor-role decomposition, which is an interpretability advantage absent in black-box tree ensembles.

5 Discussion

This work presents an approach of decomposing a time series representing a collective behavior in the estimation of several time series for understanding possible individual behaviors, as the first step of a new XAI approach. The user can see the estimation of individual behaviors as an explanation of the collective behavior. Moreover, our approach does not only improve explainability, but it also represents a new way of predicting time series. This idea of representing collective behaviors by aggregating individual behaviors has been exploited in the multi-agent systems (MAS) field. More concretely, Qin et al. [40] proposed an evolutionary collective behavior decomposition for analyzing time series with data mining. They even applied their framework to financial time series using real-world stock prices. Even, we also simulated different roles in the real-estate market [41]. The novelty of our current approach is the application of LSTMs, specific neural networks for time series, in the decomposition of time series, as a step for the explainable time series prediction in the context of trading in stock markets. The proposed behavioral decomposition offers a new interpretative dimension in financial time series modeling. By using price movement prediction as the estimation of collective behavioral ratios, the framework bridges quantitative ML with qualitative behavioral finance. Unlike traditional black-box LSTM models that focus solely on price outputs, this method yields intermediate representations that correspond to interpretable investor archetypes. Such decomposition allows estimating market sentiment shifts and institutional actions, enabling domain experts to know which behavioral group may be driving particular price changes. This interpretability contributes to both model validation and practical decision support, as behavioral dynamics often precede visible market corrections or reversals (i.e., any changes in the direction of price trend). Our approach aligns with the findings of the recent survey of XAI in financial time series [42], which highlighted the importance of developing interpretable models (designing components that are easy to describe by their nature), compared to just explanatory methods over black-box forecasting approaches.

The results obtained from the $N = 100$ simulated trading competitions provide strong empirical validation for the proposed framework. Notably, the *DecomposeLSTM* model achieved an average

final capital that more than doubled the passive market benchmark (ETF EUNL.DE). Furthermore, the non-parametric statistical analysis confirmed its superiority over standard and hybrid deep learning architectures, such as CNN-LSTM and CNN-BiLSTM. This suggests that injecting domain-specific behavioral structure into the learning process is far more effective for financial forecasting than simply increasing the architectural complexity of neural networks on raw price data.

An interesting finding from the experimental evaluation is the performance comparison between the proposed approach and the tree-based ensemble models, particularly XGBoost. While *DecomposeLSTM* yielded a higher absolute average final capital (266,572 compared to 236,995 for XGBoost), the Wilcoxon signed-rank test indicated no statistically significant difference in their median rank performance. This highlights the well-documented robustness of gradient boosting algorithms in financial tabular data. However, the critical advantage of our new behavioral LSTM framework lies in its intrinsic explainability. Where XGBoost operates as an opaque black box, our approach provides transparent, intermediate estimations of investor roles, empowering users to understand the market dynamics driving algorithmic trading decisions without sacrificing profitability.

There are closing price manipulation detection models, but the usage of manipulation is controversial. The market manipulation was proven to have an impact on investor behavioral biases, especially in companies with lower investor maturity, higher arbitrage costs and in the optimism of investors [43]. Thus, using manipulation detection should be considered with caution, since some investors cannot handle this information properly, provoking lottery preference or limited attention. The current work provides a platform that estimates this manipulation detection by providing estimated amounts of the roles implied, providing higher rates of institutional investors when manipulation is detected. Hence, different ML models and trading strategies can be applied to use this information properly.

This framework provides information on all individual investors, institutional investors, and automated operations (e.g., stop-losses). Thus, it provides freedom so user programmers can decide what information to use in their strategies. This aligns with other frameworks, such as [44], which provided different indicators and time-series prediction, so users can pick the appropriate indicator(s) in their strategies.

It is worth mentioning that the most relevant theories behind technical analysis of stock trends [45], supports and resistances are ideated based on the collective thinking as the sum of all the ways of thinking of all the individuals operating on stocks. For instance, the fact that several times the price turned around the same price may create a psychological effect that there is a support, which will probably support the price because several people will buy at that price, because this generated a collective idea. Since both LSTM approaches use proportional difference series for training and prediction, they may miss the importance of certain significant support and resistance prices. In the future, we may consider to develop a collective psychological investor ideas representation about supports and resistances that can be considered in the trading strategies, although this is left outside the current work to be able to assess the improvement of just decomposing price time series into investment role series as the fundamental improvement in LSTM-based trading strategies.

From a market microstructure perspective, the proposed approach aligns with the theoretical assumption that observed price dynamics are emergent outcomes of the aggregated interactions among heterogeneous agents. The explicit modeling of investor types (i.e., individuals, institutions, and automated stop-loss agents) introduces a structured way to approximate these micro-level interactions. This behavioral layering may also provide a path to capturing liquidity shocks or herding effects (i.e., investors collectively imitating the actions of others), which are traditionally difficult to quantify. Therefore, the framework can be viewed as an intermediary step between high-frequency agent-based modeling and purely statistical forecasting, enriching the explanatory capacity of ML models in finance.

Another important observation from the experiments is the model's ability to generalize across multiple temporal windows, maintaining performance stability even when exposed to different market periods and volatility regimes. This robustness suggests that the behavioral decomposition captures invariant structural relationships within the data, rather than overfitting to local price patterns. Nevertheless, further experimentation is warranted to assess the stability of these

behavioral patterns under extreme conditions such as financial crises, speculative bubbles, or periods of exogenous shocks. Such analyses could determine whether the behavioral LSTM retains explanatory validity during non-stationary or regime-switching environments.

The behavioral LSTM framework also presents strong complementarities with reinforcement learning (RL) and hybrid optimization approaches. Incorporating behavioral forecasts as contextual inputs in RL-based trading systems could enhance adaptive decision-making, allowing agents to adjust their policies based on predicted shifts in investor sentiment. Similarly, combining this decomposition with hybrid statistical–deep learning methods, e.g., Autoregressive Integrated Moving Average (ARIMA) LSTM [46] or Generalized Autoregressive Conditional Heteroskedasticity (GARCH) transformers [47] might yield a multi-scale view of volatility and trend persistence. Such combinations could bridge short-term tactical prediction with long-term strategic positioning, expanding the scope of the model from descriptive explainability to prescriptive intelligence.

Beyond research and trading applications, the integration of explainable behavioral modeling into the DI2XAI simulator has broader implications for financial education and regulation. By visualizing the inferred behavioral components that drive market outcomes, students and practitioners can gain an intuitive understanding of how collective psychology shapes price formation. From a regulatory standpoint, transparent AI models capable of distinguishing between organic trading activity and potential manipulation may support early warning systems for abnormal market behavior. Thus, the proposed framework contributes not only to algorithmic profitability but also to the democratization and responsible governance of AI-driven finance.

A key assumption in many ML techniques, including LSTM models, is that the input time series exhibits stationarity—meaning its statistical properties (e.g., mean, variance, and autocorrelation) remain constant over time. Stationary series enable models to learn meaningful patterns more effectively, as they avoid spurious trends or drifting behaviors that can lead to poor generalization and overfitting. In financial time series, stock prices are typically non-stationary due to long-term trends, volatility clustering, and structural breaks, which violate this assumption and complicate accurate forecasting. To address this, researchers often transform price series into stationary forms, such as percentage changes, which tend to exhibit weaker trends and more stable variance. In our study, the use of behavioral ratios derived from the DI2XAI simulator inherently promotes stationarity by focusing on normalized investor-driven metrics rather than absolute prices. This transformation likely also contributed to the robust performance of our LSTM model. Future work could test stationarity in the decomposition (e.g., using the Augmented Dickey-Fuller test) and explore additional differencing or detrending techniques to further enhance model reliability in real-world volatile markets.

6 Conclusion and future work

This paper proposed an explainable LSTM-based trading framework that models stock price dynamics through the decomposition of collective investor behavior. Instead of directly forecasting prices, this approach transforms historical price and volume series into interpretable investor role components (buyers, sellers, institutional buyers, institutional sellers, and automated stop-loss operations), which are predicted using LSTM networks. This decomposition provides intrinsic interpretability by linking model outputs through agent-based simulation surrogate models to plausible behavioral drivers of market movements. Experimental results obtained within the DI2XAI multi-agent backtesting environment show that the proposed approach consistently outperformed statistically significantly LSTM, CNN-LSTM, CNN-BiLSTM and random forest baseline, and improved XGBoost in profitability. These gains are attributed both to the behavioral abstraction and to the GA-based optimization of price-to-behavioral rules, which aligns the decomposition with trading objectives through competition-driven fitness evaluation.

The proposed decomposition-based approach not only enhances predictive performance but also bridges the gap between deep learning and behavioral finance by providing interpretable insights into market dynamics. Our findings highlight that modeling aggregated investor behaviors offers a new paradigm for explainable financial forecasting, allowing analysts and automated agents to understand not only what is predicted, but why. The statistically significant improvement in profitability demonstrates that integrating behavioral decomposition into LSTM-based architectures can yield both theoretical and practical benefits for algorithmic trading.

Our approach successfully bridges quantitative machine learning with qualitative behavioral finance. This methodology transforms traditional black-box predictions into interpretable, intermediate behavioral representations, offering a new dimension of transparency in algorithmic trading. Beyond its empirical results, this framework contributes to the growing demand for transparency and interpretability in financial AI systems particularly within regulatory and educational contexts, using agent-based subrogate simulation models. The successful integration within the DI2XAI platform underlines the feasibility of deploying explainable deep learning models in interactive environments, paving the way for human-in-the-loop financial simulations. The behaviorally informed prediction structure can serve as a foundation for broader applications beyond stock markets, such as real-estate forecasting, cryptocurrency analysis, or macroeconomic indicator prediction. By interpreting price movements as emergent phenomena of collective decision-making, the proposed method reinforces the conceptual link between technical analysis principles and modern AI modeling. This allows financial experts to understand the underlying market dynamics and investor sentiment driving the algorithmic decisions, without sacrificing profitability.

Future research can further expand and refine the proposed approach in several directions. For now, the GA has been designed and tested with a high level of guidance (precise human-designed equations) and with a relatively low number of iterations run manually. However, we plan to design a GA approach starting with simpler equation piece units, and further number of iterations with automatic run over hours, recovering from unexpected server errors to be able to obtain other more complex formulas to be compared with the presented ones. In this way, we will observe whether the results can be further improved with this approach. Moreover, the integration of multimodal and external data sources, such as financial news sentiment, social media indicators, or macroeconomic variables, could enrich the behavioral decomposition process by capturing exogenous factors that influence investor decision-making. This multimodal integration may also help identify latent correlations between behavioral ratios and market sentiment dynamics, improving model robustness across varying volatility regimes. Furthermore, we plan to explore advanced explainability mechanisms to translate the model's behavioral predictions into human-interpretable narratives or visual explanations. Incorporating XAI layers capable of quantifying the contribution of each behavioral component to final trading outcomes would significantly enhance transparency and user trust, aligning the framework with regulatory and ethical demands in financial AI. Another line of work could focus on computational efficiency and optimization. Although the current model leverages a GA for rule tuning, alternative metaheuristic techniques such as particle swarm optimization or ant colony optimization could be investigated to reduce computational overhead. Lightweight or quantized variants of LSTM architectures may facilitate deployment in real-time or resource-constrained environments, such as embedded trading agents or educational platforms.

Another relevant extension involves multi-agent and real-time validation. Embedding the behavioral LSTM approach within dynamic, interactive trading ecosystems could enable the study of emergent market phenomena driven by adaptive agents. Integrating reinforcement learning components that adjust strategies based on behavioral signals would provide a deeper understanding of how collective dynamics evolve under competition and cooperation, thereby strengthening the realism and applicability of the model. In addition, the framework could be generalized to other asset classes, including cryptocurrencies, commodities, and derivatives, to evaluate its adaptability across distinct market structures and volatility patterns. Such comparative analyses would offer valuable insights into the universality of behavioral decomposition as a predictive and explanatory mechanism.

Acknowledgements. This work was supported by the CATALYST project (PID2025-168174OB-I00), funded by the Spanish State Research Agency (MICIU/AEI/10.13039/501100011033). This work also thanks the "Contract about Anticipation and defense against cyberattacks to Spanish institutions generating explanations", (2022-2023), Programa Investigo, Comunidad de Madrid, with reference CT36/22-08-UCM-INV, for their support in our research in XAI. This work was also partially funded by the Spanish Ministry of Education through the project with reference PID2023-149341OB-I00.

References

[1] Sezer OB, Gudelek MU, Ozbayoglu AM. Financial time series forecasting with deep learning : A systematic literature review: 2005-2019. Applied Soft Computing. 2020 May;90. https://doi.org/10.1016/j.asoc.2020.106181.

[2] Xiao D, Su J. Research on Stock Price Time Series Prediction Based on Deep Learning and Autoregressive Integrated Moving Average. Scientific Programming. 2022 MAR 31;2022. https://doi.org/10.1155/2022/4758698.

[3] Shahi TB, Shrestha A, Neupane A, Guo W. Stock Price Forecasting with Deep Learning: A Comparative Study. Mathematics. 2020;8(9). https://doi.org/10.3390/math8091441.

[4] Kumar R, Kumar P, Kumar Y. Analysis of Financial Time Series Forecasting using Deep Learning Model. In: 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence); 2021. p. 877–881.

[5] Istiake Sunny MA, Maswood MMS, Alharbi AG. Deep Learning-Based Stock Price Prediction Using LSTM and Bi-Directional LSTM Model. In: 2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES); 2020. p. 87–92.

[6] Chavhan S, Raj P, Raj P, Dutta AK, Rodrigues JJPC. Deep Learning Approaches for Stock Price Prediction: A Comparative Study of LSTM, RNN, and GRU Models. In: Solic P, Nizetic S, Rodrigues J, Perkovic T, Catarinucci L, Patrono L, et al., editors. 2024 9TH International Conference on Smart and Sustainable Technologies, SpliTech 2024. IEEE; IEEE Commun Soc; IEEE Council RFID; 2024. p. 01–06. 9th International Conference on Smart and Sustainable Technologies (SpliTech), Split, Croatia, Jun 25-28, 2024.

[7] Jin Z, Yang Y, Liu Y. Stock closing price prediction based on sentiment analysis and LSTM. Neural Computing & Applications. 2020 Jul;32(13):9713–9729. https://doi.org/10.1007/s00521-019-04504-2.

[8] Abdullah M, Sulong Z, Chowdhury MAF. Explainable deep learning model for stock price forecasting using textual analysis. Expert Systems with Applications. 2024 Sep 1;249(C). https://doi.org/10.1016/j.eswa.2024.123740.

[9] Li M, Zhu Y, Shen Y, Angelova M. Clustering-enhanced stock price prediction using deep learning. World Wide Web-Internet and Web Information Systems. 2023 Jan;26(1):207–232. https://doi.org/10.1007/s11280-021-01003-0.

[10] Fan C, Zhang X. Stock price nowcasting and forecasting with deep learning. Journal of Intelligent Information Systems. 2025 Apr;63(2):639–656. https://doi.org/10.1007/s10844-024-00908-2.

[11] Palar PS, Saves P, Robani MD, Verstaevel N, Garouani M, Aligon J, et al. Interpretable and Explainable Surrogate Modeling for Simulations: A State-of-the-Art Survey and Perspectives on Explainable AI for Decision-Making. Archives of Computational Methods in Engineering. 2026;33:12511–12565. https://doi.org/10.1007/s11831-026-10600-z.

[12] García-Magariño I, Bravo-Agapito J. DI2XAI: Dashboard for intelligent investment with practice over historic data using explainable artificial intelligence. Available at SSRN 5667638. 2025;https://doi.org/10.2139/ssrn.5667638.

[13] Kothari A, Kulkarni A, Kohade T, Pawar C. Stock Market Prediction Using LSTM. In: So-In C, Joshi A, Senjyu T, editors. Smart Trends in Computing and Communications, Vol. 3, SmartCom 2024. vol. 947 of Lecture Notes in Networks and Systems; 2024. p. 143–164. 8th Smart Trends in Computing and Communications (SmartCom), Pune, India, Jan 12-13, 2024.

[14] Agrawal M, Shukla PK, Nair R, Nayyar A, Masud M. Stock Prediction Based on Technical Indicators Using Deep Learning Model. CMC-Computers Materials & Continua. 2022;70(1):287–304. https://doi.org/10.32604/cmc.2022.014637.

[15] Li C, Zhang X, Qaosar M, Ahmed S, Alam KMR, Morimoto Y. Multi-factor Based Stock Price Prediction Using Hybrid Neural Networks with Attention Mechanism. In: IEEE 17th Int. Conf. on Dependable, Autonom and Secure Comp. / IEEE 17th Int. Conf. on Pervas Intelligence and Comp. / IEEE 5th Int. Conf. on Cloud and Big Data Comp. / IEEE 4th Cyber Science and Technology Congress (DASC/PICOM/CBDCOM/CYBERSCITECH). IEEE; IEEE Comp Soc; 2019. p. 961–966.

[16] Zhang X, Pinsky E. S&P-500 vs. Nasdaq-100 price movement prediction with LSTM for different daily periods. Machine Learning with Applications. 2025 Mar;19:100617. https://doi.org/10.1016/j.mlwa.2024.100617.

[17] Lee Ar, Ahn JY, Choi JE, Kim K. DR-LSTM: Dimension reduction based deep learning approach to predict stock price. Communications for Statistical Applications and Methods. 2024 Mar;31(2):213–234. https://doi.org/10.29220/CSAM.2024.31.2.213.

[18] Baek H. A CNN-LSTM Stock Prediction Model Based on Genetic Algorithm Optimization. Asia-Pacific Financial Markets. 2024 Jun;31(2):205–220. https://doi.org/10.1007/s10690-023-09412-z.

[19] Fan X, Tao C, Zhao J. Advanced Stock Price Prediction with xLSTM-Based Models: Improving Long-Term Forecasting. In: 2024 11th International Conference on Soft Computing & Machine Intelligence (ISCMI); 2024. p. 117–123.

[20] Ding G, Qin L. Study on the prediction of stock price based on the associated network model of LSTM. International Journal of Machine Learning and Cybernetics. 2020 Jun;11(6):1307–1317. https://doi.org/10.1007/s13042-019-01041-1.

[21] Zou J, Lou J, Wang B, Liu S. A novel deep reinforcement learning based automated stock trading system using cascaded LSTM networks. Expert Systems with Applications. 2024;242:122801.

[22] Yang C, Zhai J, Tao G. Deep Learning for Price Movement Prediction Using Convolutional Neural Network and Long Short-Term Memory. Mathematical Problems in Engineering. 2020;2020:1–13.

[23] Lu W, Li J, Wang J, Qin L. A CNN-BiLSTM-AM method for stock price prediction. Neural Computing & Applications. 2021 May;33(10, SI):4741–4753. https://doi.org/10.1007/s00521-020-05532-z.

[24] Karim ME, Foysal M, Das S. Stock Price Prediction Using Bi-LSTM and GRU-Based Hybrid Deep Learning Approach. In: Proceedings of Third Doctoral Symposium on Computational Intelligence. Singapore: Springer Nature Singapore; 2023. p. 701–711.

[25] Farhadi A, Zamanifar A, Alipour A, Taheri A, Asadolahi M. A Hybrid LSTM-GRU Model for Stock Price Prediction. IEEE Access. 2025;13:117594–117618. https://doi.org/10.1109/ACCESS.2025.3586558.

[26] Mehtarizadeh H, Mansouri N, Zade BMH, Hosseini MM. Stock price prediction with SCA-LSTM network and Statistical model ARIMA-GARCH. Journal of Supercomputing. 2025 Jan;81(2). https://doi.org/10.1007/s11227-024-06775-6.

[27] Liu F, Qin P, You J, Fu Y. Sparrow Search Algorithm-Optimized Long Short-Term Memory Model for Stock Trend Prediction. Computational Intelligence and Neuroscience. 2022 Aug 12;2022. https://doi.org/10.1155/2022/3680419.

[28] Yadav K, Yadav M, Saini S. Stock values predictions using deep learning based hybrid models. CAAI Transactions on Intelligence Technology. 2022 Mar;7(1):107–116. https://doi.org/10.1049/cit2.12052.

[29] Liagkouras K, Metaxiotis K. A Hybrid Long Short-Term Memory with a Sentiment Analysis System for Stock Market Forecasting. Electronics. 2025 Jul 8;14(14). https://doi.org/10.3390/electronics14142753.

[30] Narayana ML, Kartha AJ, Mandal AK, Roshini P, Suresh A, Jose AC. Ensemble time series models for stock price prediction and portfolio optimization with sentiment analysis. Journal of Intelligent Information Systems. 2025 Aug;63(4):1079–1103. https://doi.org/10.1007/s10844-025-00928-6.

[31] Saraswat R, Kumar A. Stock Price Forecasting With Integration of Sectoral Behavior: A Deep Auto-Optimized Multimodal Framework. Journal of Forecasting. 2025 Aug;44(5):1767–1784. https://doi.org/10.1002/for.3265.

[32] García-Magariño I, Bravo-Agapito J. Supporting and explaining stock investment decisions through Shapley Additive Explanations on LSTM and MLP-based agent models with user experience on one-decade backtesting. Applied Intelligence. 2026;56:388.

[33] Huang Y, Cui K, Lu X. Stock Trading Strategy Based on Multi-Scale Deep Reinforcement Learning and Price Movement Prediction. International Journal of Information Technology & Decision Making. 2025 Jul 29;https://doi.org/10.1142/S0219622025500737.

[34] Alghamdi H, Alqithami S. A robust machine learning framework for stock market classification. Expert Systems with Applications. 2025 Dec 15;294. https://doi.org/10.1016/j.eswa.2025.128573.

[35] Ishwarappa, Anuradha J. Big data based stock trend prediction using deep CNN with reinforcement-LSTM model. International Journal of System Assurance Engineering and Management. 2021 Mar 3;https://doi.org/10.1007/s13198-021-01074-2.

[36] Teixeira DM, Barbosa RS. Stock Price Prediction in the Financial Market Using Machine Learning Models. Computation. 2024;13(1):3.

[37] Ülkü N. Investor types' trading behavior in stock markets. Encyclopedia of Monetary Policy, Financial Markets and Banking. 2025;3:455–461.

[38] Hyndman RJ, Athanasopoulos G. Forecasting: Principles and Practice. 3rd ed. Melbourne, Australia: OTexts; 2021. Available from: https://otexts.com/fpp2/.

[39] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. Curran Associates, Inc.; 2019. p. 8026–8037.

[40] Qin Z, Wan T, Dong Y, Du Y. Evolutionary collective behavior decomposition model for time series data mining. Applied Soft Computing. 2015;26:368–377.

[41] García-Magariño I, Medrano C, Delgado J. Estimation of missing prices in real-estate market agent-based simulations with machine learning and dimensionality reduction methods. Neural Computing and Applications. 2020;32(7):2665–2682.

[42] Arsenault PD, Wang S, Patenaude JM. A survey of explainable artificial intelligence (XAI) in financial time series forecasting. ACM Computing Surveys. 2025;57(10):1–37.

[43] Liu J, Zhang J, Chen Z. The effect of stock market manipulation on investor behavioral bias. Journal of Behavioral and Experimental Finance. 2025;p. 101090.

[44] Varela A, Siam MK, Al Maruf A, Gu H, Cheng JQ, Aung Z. An analytical framework for real-time gold trading using sentiment and time-series forecasting. Decision Analytics Journal. 2025;p. 100633.

[45] Edwards RD, Magee J, Bassetti WC. Technical analysis of stock trends. CRC press; 2018.

[46] Kashif K, Ślepaczuk R. LSTM-ARIMA as a hybrid approach in algorithmic investment strategies. Knowledge-Based Systems. 2025;p. 113563.

[47] Mutinda JK, Langat AK. Stock price prediction using combined GARCH-AI models. Scientific African. 2024;26:e02374.