Received: 12 October 2023 | Revised: 26 March 2024 | Accepted: 11 June 2024

DOI: 10.1111/exsy.13667

ORIGINAL ARTICLE

Expert Systems WILEY

Portfolio construction using explainable reinforcement learning

Daniel González Cortés[1] | Enrique Onieva[2] | Iker Pastor[2] | Laura Trinchera[1] | Jian Wu[1]

[1]NEOMA Business School, Mont Saint Aignan, France

[2]Faculty of Engineering, University of Deusto, Bilbao, Spain

Correspondence
Daniel González Cortés, NEOMA Business School, Rue du Maréchal Juin, Mont Saint Aignan Cedex 76825, France.
Email: daniel-alejandro.gonzalez-cortes.20@neoma-bs.com

Funding information
NEOMA Business School's AI, Data Science & Business Area of Excellence, Grant/Award Number: 416004

Abstract

While machine learning's role in financial trading has advanced considerably, algorithmic transparency and explainability challenges still exist. This research enriches prior studies focused on high-frequency financial data prediction by introducing an explainable reinforcement learning model for portfolio management. This model transcends basic asset prediction, formulating concrete, actionable trading strategies. The methodology is applied in a custom trading environment mimicking the CAC-40 index's financial conditions, allowing the model to adapt dynamically to market changes based on iterative learning from historical data. Empirical findings reveal that the model outperforms an equally weighted portfolio in out-of-sample tests. The study offers a dual contribution: it elevates algorithmic planning while significantly boosting transparency and interpretability in financial machine learning. This approach tackles the enduring 'black-box' issue and provides a holistic, transparent framework for managing investment portfolios.

KEYWORDS
algorithmic transparency, explainable reinforcement learning, finance, portfolio management

1 | INTRODUCTION

Building portfolios with multiple risk financial assets has received much attention from academics and investors; however, it is challenging due to the complex and evolving nature of financial markets. Since Markowitz's initial work on portfolio selection (Markowitz, 1952), many ideas and techniques for developing investment portfolios have emerged, mainly to address the difficulties of applying the model in practice. For example, Perold and Sharpe (1988) showed the importance of proper dynamic asset allocation to increase the value of a portfolio and better deal with market fluctuations, while Black and Litterman (1990) introduced a Bayesian technique to incorporate investors' subjective views. In the same way, other techniques have helped to expand the literature, such as the incorporation of constraints in the portfolio weights to reduce the risk in large portfolios (Jagannathan & Ma, 2003), robust optimization (Tütüncü & Koenig, 2004), naive diversification (DeMiguel et al., 2009) and by adding tail-risk measures (Harvey et al., 2010).

In recent years, the emergence of Machine Learning (ML) as a powerful and disruptive technology that enables computers to learn and predict financial events using multiple data sources (Kamruzzaman et al., 2024) has gained popularity. As a result, different approaches have been used to address the construction of portfolios to maximize the distribution of resources using ML, such as implementations built using Deep Learning (DL) (Ma et al., 2021), Reinforcement Learning (RL) (Syu et al., 2020; Zhang et al., 2020) and different optimization techniques such as particle

Abbreviation: CAC-40, 'Cotation Assistée en Continu', representing the 40 largest firms on the Euronext Paris stock exchange.

swarm optimization (Thakkar & Chaudhari, 2021), genetic algorithms (Cheong et al., 2017), hybrid techniques (Paiva et al., 2019) among others (Ge et al., 2014; Lai et al., 2020; Li et al., 2017; Perrin & Roncalli, 2020).

Despite the popularity of ML, it is still not widely accepted among financial experts for different reasons. One is the difficulty of selecting a suitable algorithm with a correct objective function that guarantees efficient and effective performance. Also, it is challenging to test a realistic environment using back-testing to verify the success of a strategy because the implicit assumption is that the future will behave like the past. In this way, algorithms will likely learn from high-dimensional datasets where the behaviour of some of the input variables will change or will not be relevant in the upcoming periods.

Additionally, one of the most significant concerns is the lack of transparency of ML algorithms, which often exhibit black-box behaviour, and it is difficult to determine which features are the most relevant during the learning process. Knowing the importance of variables can help decision-makers reform and audit their algorithms when necessary. However, this is not only a concern in the financial system but also in different areas where making decisions based on results that are not justifiable or explanatory is dangerous, which has generated a growing demand for transparent ML algorithms from different stakeholders (Barredo Arrieta et al., 2020).

This research intends to display a suitable solution to build an investment strategy using an explainable RL with an attention-layers approach to creating an investment portfolio of shares listed in the French CAC-40. The primary purpose of this research is to implement a technique that efficiently allocates financial assets through a continuous learning process showing the main variables considered by the model. The main contribution of this study is the creation of an explainable RL application to be used in forming an investment portfolio. So far, the authors of this research have not found another paper that covers explanations for RL applications in the creation of portfolios; therefore, our work provides significant insights to academics and professionals on how to deal with transparent and explainable ML models.

The study presents a novel approach to integrating RL within the context of the CAC-40. It connects to the efficient and adaptive market hypotheses by recognizing that financial markets can be unpredictable and change rapidly. Despite these challenges, the research aims to use explainable RL to find patterns in the market's behaviour. Evidence suggests that stock returns might not be completely random and can show some predictable trends (Bao et al., 2023; Lo & MacKinlay, 1988). Building on this foundation, we developed an explainable RL model that efficiently explores and creates an optimal portfolio during different periods, and it is validated using an out-of-sample testing period. Specifically, we build a model that: (i) learns an objective function to build a portfolio that maximizes the earnings, (ii) tests the performance in an out-of-sample period, and (iii) provides an explainable tool to obtain the importance of each feature in the learning process. The rest of this paper is arranged as follows: Section 2 presents a background on the construction of portfolios using ML and explainable models. Then, in Section 3, we present the methodology followed to construct the RL application where the structure of the decision process is described, outlining the structure of the agent, environment, and attention layer that will help to provide explainability. Next, in Section 4, we show details of the experimentation, followed by Section 5, where we summarize the results and discuss them. Lastly, in Section 6, we complete the paper with conclusions and a discussion about future research directions and the limitations of RL agents in creating financial portfolios.

2 | LITERATURE REVIEW

This section reviews RL applied to finance in the first subsection and continues with the review of the concept of explainable artificial intelligence in the following subsection.

2.1 | RL in finance

Novel ML techniques bring significant advantages to analysing complex phenomena by providing new approaches to data modelling, clustering, and forecasting. They have also been similarly used in finance, which has generated growing interest and literature (Goodell et al., 2021). A subtype of ML is DL, with widespread applications in the field of finance; this technique is an eminent portfolio management tool and can be separated into two groups. The first approach deals with model-based methods which predict prices but do not deal directly with trading and thus require another method to execute the trade, which may be based on a system of rules derived from human experience or meta-heuristics. Within this group, many authors (Heaton et al., 2017; Kamalov, 2020; Ozbayoglu et al., 2020) rely heavily on using Artificial Neural Networks (ANN) to predict prices. On the other hand, a model-free approach creates a trading strategy without the need for an explicit price prediction process, where an ANN can create a portfolio vector by mapping variables from the assets (Betancourt & Chen, 2021) and typically uses the RL method to train the network.

Different authors have used RL in finance to create portfolios and trading systems with widespread acceptance in the financial community (Millea, 2021; Ozbayoglu et al., 2020). Early work (Moody et al., 1998) showed the feasibility of these techniques by presenting empirical results in controlled experiments to demonstrate the effectiveness of optimization methods in creating portfolios focusing on a custom Sharpe ratio; also, in the same manner, different attempts have been applied to automated trading systems using adaptive RL (Dempster & Leemans, 2006; Dempster & Romahi, 2002).

Various DL methods allowed researchers to train ANN techniques that enhance the use of RL and the use of popular algorithms such as Policy Gradients (PG), Advantage Actor-Critic (A2C), and Deep Q-Learning (DQL) (Mnih et al., 2015, 2016; Sutton et al., 1999; Watkins & Dayan, 1992) have emerged as important advances in the discovery of investment policies in the stock and futures markets (Deng et al., 2017; Dixon et al., 2020; Moody & Saffell, 2001; Pendharkar & Cusatis, 2018) as well as in the cryptocurrency market (Sattarov et al., 2020) Jeong and Kim (2019) implemented a DQL RL model to improve financial trading decisions by adjusting the number of shares in the portfolio and implementing transfer learning to deal with insufficient data and avoid overfitting. Likewise, Zarkias et al. (2019) executed a trading strategy applying DQL to predict the Euro-Dollar exchange rates using a trailing stop strategy. Similarly, Li et al. (2019) used deep RL to design a trading strategy by using a Long Short-Term Memory (LSTM) network and extending the use of DQL to Asynchronous Advantage Actor-Critic for better adaptation to trading market conditions.

Zhang et al. (2020) designed an RL trading strategy testing A2C, DQL, and PG against traditional time-series momentum strategies using a dataset from 2011 to 2019 on 50 liquid futures contracts, showing that the RL models beat the classical models. Similarly, Yang et al. (2020) created an ensemble trading strategy using three based actor-critic algorithms. Brim (2020) utilized a model for trading pairs of cointegrated financial assets using DQN and double DQN, creating a strategy with positive returns. Additionally, Fengqian and Chao (2020) introduced k-line theory clustered learning features into the model to characterize candlesticks as a way to generalize price movements over time and then used DL to create an online control of the parameters in the environment. In a similar way, Taghian et al. (2022) trained an RL model with candlestick data to learn trading rules. Similarly, Hirchoua et al. (2021) developed an approach based on a rule-based policy for different agents trading against each other in a virtual environment.

Recently, some modifications to the RL techniques have been proposed to create financial portfolios; for example, Aboussalah and Lee (2020) incorporated portfolio constraints and continuous actions on numerous assets into the proposed trading RL framework. Wu et al. (2020) proposed a trading strategy using PG and DQL methods and recurrent ANN that outperformed other popular strategies. Further research conducted by Betancourt and Chen (2021) created a novel portfolio management solution using RL, with a dynamic number of multiple assets, attempting to learn the optimal holding in order to minimize the transaction costs. Similarly, Lim et al. (2022) presented a dynamic rebalancing of portfolios using RL and LSTM networks.

A novel approach was proposed by Théate and Ernst (2021) by training the RL agents on generated artificial trajectories from a partial set of historical data from the stock market. Another original methodology in the construction of trading RL agents in the stock market is the utilization of multi-agents to deal with collective intelligence. AbdelKawy et al. (2021) created a multi-stock model based on synchronous multi-agents to deal with an extensive historical dataset. Meanwhile, Shavandi and Khedmati (2022) developed a multi-agent RL application that is tested on a historical dataset from an important currency pair, outperforming single agents based on different return and risk performance measures in different time frames.

2.2 | Explainable artificial intelligence

Even though ML and RL techniques are expanding in different sectors and disciplines and are gaining popularity in finance due to their strong performance, stakeholders are concerned about the opacity of these methods (Langer et al., 2021). Therefore, to satisfy the need for transparent models, eXplainable Artificial Intelligence (XAI) has emerged as a flourishing area of multidisciplinary research aimed at creating artificial systems that humans can read and understand (Dikmen & Burns, 2022).

In previous publications, XAI has been used in finance with an increasing demand from banks and supervisory authorities to guarantee accountability, fairness, and transparency (Kuiper et al., 2022). In recent work, different ML applications have been developed; for example, Ohana et al. (2021) applied an ML to analyse stock market crashes, while Sachan et al. (2020) implemented a decision support system that explains the procedure that determines the approval or rejection of a loan. Likewise, XAI has been applied to explain stock market trend prediction (Mandeep et al., 2022), auditing (Zhang et al., 2022), credit scoring (El Qadi et al., 2022), money laundry detection (Kute et al., 2021) and other financial applications (Hoepner et al., 2021).

Despite recent advances in the development of explanatory models for intelligent models, RL has not yet been fully explored in this field (Krajna et al., 2022), and research has begun to grow due to the current demand and rise of RL as an efficient technique. Heuillet et al. (2021) classify the latest eXplainable RL (XRL) studies according to the main ideas of Barredo Arrieta et al. (2020), which are presented in two primary groups: transparent algorithms and post-hoc explainability. The initial group can subsequently be subdivided into three subgroups; hierarchical, representation and simultaneous learning, while the second group is segmented into interaction data and saliency maps models. In Figure 1 it is possible to observe the categorization of different XRLs.

Transparent algorithms can be explained by themselves by examining their architecture without applying an external model to understand their behaviour. By examining the sub-groups, we can see that hierarchical learning involves a high-level agent learning an interpretable representation of the environment, while a low-level agent splits the main goal into a set of different subgoals and learns which one is optimal. An efficient approach to filtering which sub-goals have been operated is the Hindsight Experience Replay developed by Andrychowicz et al. (2017), which

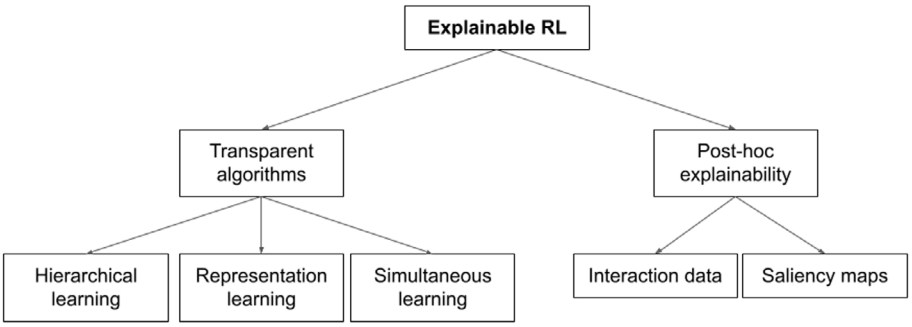

FIGURE 1 The general scheme of the explainable RL is based on the categorization made by Heuillet et al. (2021).

allows RL algorithms to learn policies from sparse rewards efficiently (Fang et al., 2019). In a different way, representation learning uses the explanation of a meaningful low-dimensional representation of the state (Lesort et al., 2018). In contrast, simultaneous learning is based on applying a method that allows the learning of the policy and the development of explanatory models via casual models (Madumal et al., 2020) and reward decomposition (Juozapaitis et al., 2019).

Additionally, attention mechanisms are gaining significance due to their ability to provide detailed understanding. Studies such as Amirshahi and Lahmiri (2023) highlight the effectiveness of hybrid models incorporating attention layers for predicting cryptocurrency prices using an ensemble of language models to analyse social media sentiment. Similarly, Zhang et al. (2023) proposed an attention-based CNN-BiLSTM hybrid model for credit risk prediction in real estate enterprises, showing high accuracy. Meanwhile, Grądzki and Wójcik (2023) has demonstrated that Transformer neural networks featuring attention mechanisms exhibit predictive prowess in intraday Forex trading, surpassing even the ResNet-LSTM benchmark models. These converging lines of evidence underscore the utility and versatility of attention mechanisms in financial modelling and prediction in the financial markets.

Another way to explain a model that is not transparent is by using post-hoc methods and creating a different algorithm for explainability, which is used after the training and testing of the main model. When investigating the sub-groups, we observe that interaction data allows the agent to extract characteristics of interest by scrutinizing the history of interactions with the environment, capturing the key features in a given task (Sequeira & Gervasio, 2020). In the same vein, saliency maps have been created to generate XRL, creating a visual interpretation using a topographically assembled map to display the importance of each pixel in a given image (Greydanus et al., 2018; Guo et al., 2021).

3 | METHODOLOGY

This work brings together the RL world of the XAI and applies it to the financial field to create an automated trading system. This section will describe our steps to create our RL model and scrutinize its explanatory properties. First, we will start by presenting the setup, explaining the decision process by introducing the concept of state, action space, and reward function, followed by the presentation of the agent and its structure and then continue with the exposition of the attention layers that will be added to the agent to obtain explainability.

3.1 | The decision process

Given that an investor seeks to maximize their profits, under the concept of modern portfolio theory we can see that the expected utility function (Arrow, 1974; Pratt, 1964) is defined as,

$$\mathbb{E}[U(W_T)] = \mathbb{E}\left[U\left(W_0 + \sum_{t=1}^{n} \delta W_t\right)\right], \tag{1}$$

where over time T the utility function U has a final wealth function W_T. This framework proposes that an investor's satisfaction or utility from final wealth is determined not solely by initial wealth W_0 but also by the cumulative changes in wealth δW_t across periods. The expectation $\mathbb{E}$ encapsulates the probabilistic outcomes of different investment decisions, factoring in the likelihood and impact of potential gains or losses over time. This approach, emphasizing the trade-off between risk and return, underlines the investor's objective to craft a portfolio that not only aims for the highest possible returns but also aligns with their risk tolerance.

In this research we assume that the investor is risk neutral and their utility function is linear since the objective is to maximize the expected cumulative returns, as shown in Equation (2).

$$\mathbb{E} = \sum_{t=1}^{T} \delta W_t. \tag{2}$$

Assuming a linear utility function for an investor, particularly under modern portfolio theory and in modern investment approaches (Zhang et al., 2020), is a theoretical simplification that facilitates the mathematical modelling of investment strategies. Using a linear function means that the investor is considered risk-neutral. This means that the investor values incremental gains in wealth equally. Risk-neutral investors are aware of the risks involved, yet they choose not to factor them into their decision-making process at the portfolio construction stage, focusing instead on maximizing expected returns irrespective of the variance in those returns.

Thus, the objective of the RL in this study is to maximize an investor's wealth by following a sequential or Markov Decision Process (MDP) in which the agents learn by interacting with an environment at time steps to achieve a goal. At any point in time t, the agent obtains a representation of the environment denoted as a state s_t in a state space $\mathcal{S}$ and then takes action a_t from the action space $\mathcal{A}$, following a policy $\pi(a_t|s_t)$.

Because the agent takes action a_t in a certain state s_t, it receives a scalar reward r_t, and according to the dynamics of the environment, it also receives the transition to the next state s_{t+1} for a state transition probability $\mathcal{P}(s_{t+1}|s_t,a_t)$ and a reward function $\mathcal{R}(s,a)$. The constant interplay of the agent with the environment yields a trajectory $\tau = [s_0,a_0,r_0,s_1,a_1,r_1,...,s_n,a_n,r_T]$. The goal of the agent is to maximize the expected return at time t as denoted in the following equation,

$$G_t = \sum_{j=t+1}^{n} \gamma^{j-t-1} r_j, \tag{3}$$

where, $\gamma \in (0,1]$ is the discount factor; in this way MDP, can be defined by a tuple $(\mathcal{S},\mathcal{A},\mathcal{P},\mathcal{R},\gamma)$. Therefore, the RL agent focuses on developing a maximization r_t, thus optimizing the function $\mathbb{E}(G)$ is equivalent to optimizing Equation (2), where a given investor maximizes the expected cumulative returns.

To create a reward function $\mathcal{R}$ that satisfies and relies on Equation (1), in this research, we calculate the returns by creating a vector q that represents the shares of assets that an investor has, where its components are symbolized as q_i where $i \in 0,1,..,n$ in a specific market with n financial assets. Each component q_i must receive a certain weight w_i, representing the proportion of this element in a portfolio. In this way, the expected returns and reward function can be represented as,

$$\mathcal{R} = \mathbb{E}[q_n] = \sum_{i=1}^{n} w_i \mathbb{E}[q_i]. \tag{4}$$

The action space $\mathcal{A}$ is the set of all possible actions and determines how the agent reacts in a given environment, and in this research, we define an action a_t at a given time t as,

$$a_t = w : w \in \mathbb{R}, 0 \le w \le 1, \tag{5}$$

where,

$$\sum_{i=1}^{n} w_i \le 1. \tag{6}$$

In this way, the agent is trained to find an optimal w_i for each q_i component. Since, in the initial stage, the agent has no prior knowledge of the optimal values of w, these are started randomly. Subsequently, these change according to the agent's training through RL. The modification of the weights generates an adjustment in the number of shares held in our model, which must be translated into a purchase or sale of assets to adjust the portfolio within our simulation environment.

Regarding the state space, $\mathcal{S}$ can be considered the set that holds all the possible variables in the agent's environment. It is impossible to know which variables will influence the movement of a certain financial asset. However, in this work, we have created a state space with features other authors have used in the literature.

3.2 | The agent

The goal of the RL agent is to find an optimal policy π to maximize the reward function $\mathcal{R}$, to achieve this, an ANN can be a function approximator that outputs a policy $\pi_\theta(a_t|s_t)$ where the parameters of the function are represented by θ.

The policy gradients method to update θ based on rewards uses gradient ascent on the expected cumulative returns of the policy represented as $J(\theta)$:

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau \left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) G_t \right]. \tag{7}$$

However, this process is not always efficient because the policy update is developed once the episode ends, so a proposed modification to solve this is the A2C method, which is the synchronous version of the Asynchronous Advantage Actor-Critic algorithm proposed in Mnih et al. (2016) to update the policy in real-time. This solution relies on two models, one is a network that plays the role of an actor creating a policy that updates through the output of a critic that estimates the value function in a given state. The A2C method maximizes the objective function by updating the policy π_θ as defined in the following equation:

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau \left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) A_{advantage}(S,A) \right]. \tag{8}$$

The improvement of A2C relies on the addition of an advantage function that is expressed as:

$$A_{advantage}(S_t,A_t) = R_t + \gamma V(S_{t+1}|w) - V(S_t|w), \tag{9}$$

where, $V(s|w)$ represents a state function with parameters w from the critic network that can be updated by minimizing the temporal difference error using gradient descent as:

$$J(\theta) = (R_t + \gamma V(S_{t+1}|w) - V(S_t|w))^2. \tag{10}$$

Since the agent must process the information contained in the state space, in this research, we have designed a multi-head LSTM neural network structure to process the information of each asset in a parallel way; by doing this, the data of an asset is processed by a network LSTM independent of that of another asset as shown in Figure 2.

The results of each network are concatenated to be then processed by another ANN that will finally give two outputs: the critic's values and the actor's actions.

The actor's output determines the agent's actions and asset weights, which is why a vector with several elements equal to the number of assets in the portfolio represents it. These elements comprise the action space expressed in Equation (5) and (6), which go directly to the environment to perform a buying or selling action.

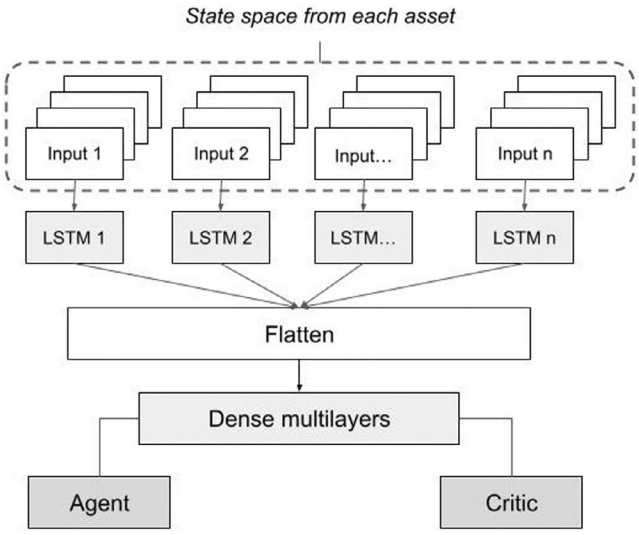

FIGURE 2 General scheme of the agent's architecture.

3.3 | The environment

After the agent has delivered a series of actions to be taken, these must be tested in an environment that allows financial conditions to be simulated. To achieve a realistic scenario, we have created an environment that enable trading shares in the market by emulating the set of shares available for transactions.

The environment starts once it receives the sets containing the current state s_t, composed of ten elements with four lagging variables. In this way, we develop $\mathcal{S}$ incorporating the different opening, high, low, and closing prices and volume at a given time t. Additionally, we include different technical indicators commonly used to predict financial markets. In this paper, we include moving average convergence/divergence (MACD), relative strength index (RSI), and 14, 21, and 100-day moving averages (MA).

The MA technical indicator can be described as:

$$\mathrm{MA}(k) = \frac{1}{N}\sum_{t=k}^{N+k-1} \mathrm{closingprice}_t, \tag{11}$$

where, N stands for the number of data points and k is the lagging period.

$$\mathrm{RSI} = 100 - 100/(1 - \mathrm{RS}_p), \tag{12}$$

where, RS_p is the relationship between the averages of up price movements with the average of down price change in period p.

For the MACD formula, it is first necessary to calculate the Exponential Moving Average (EMA) since it is the difference between the fast and a slow formula, and it is defined as:

$$\mathrm{MACD}(\mathrm{fast},\mathrm{slow}) = \mathrm{EMA}_{fast} - \mathrm{EMA}_{slow}, \tag{13}$$

$$\mathrm{EMA}_t = \alpha^* \mathrm{closingprice}_t + (1-\alpha) * \mathrm{EMA}_{t-1}, \tag{14}$$

and α stands for the degree of decrease, calculated as $\frac{2}{t+1}$, where t is the day of EMA.

Additionally, the environment needs a set of action vectors from the agent a_t; this information calculates how much capital should be positioned in each asset. The weights from this vector are divided over the initial cash amount[1] and then divided by a purchase price, with a rounded down to the next lower integer. The simulation environment employs an adaptive pricing model to refine this process further and mitigate the challenges of executing buy or sell orders within a given trading day. This model strategically selects execution prices within the day's observed price range, embracing the market's inherent volatility and liquidity constraints. By incorporating a random selection mechanism within a trading day, the simulation reflects a more realistic scenario where the exact price at which an order is filled cannot be predetermined, thus closely mimicking the unpredictability and fluctuations of market conditions. This refined approach tackles the uncertainty associated with trade execution prices but also aids in preventing overfitted strategies.

Once there is a change in the weightings, the portfolio is readjusted and starts by selling shares if the agent's weight is less than that of the portfolio, and buying is done if the weighting is increased and there is enough cash because, in this RL application, we do not consider leverage. Once the simulation is finished, the environment produces the final balance with which the performance will be evaluated.

Once the simulation has started, the agent sends an updated set of portfolio weights, which will rebalance the portfolio if different from the previous actions. This action implies the simulation of the sale or purchase of assets. If the new weight of a specific asset is greater than the old one, a purchase must be made. In each step within the simulator, we first sell all the shares that must be readjusted and then buy those necessary to balance the portfolio according to the agent's actions.

The environment is implemented utilizing the OpenAI Gym interface,[2] featuring a custom-designed simulation environment tailored explicitly for the demands of algorithmic trading research. Detailed pseudo-code outlining the operational framework of the trading environment is delineated within the structure of Algorithm 1, illustrating its comprehensive structure and functionality.

3.4 | The attention layer

Due to the great need to understand the variables involved in the decision-making process of different ML algorithms, especially neural networks, different models have emerged that help show the importance of the different features that have become popular over the years. Within the taxonomy of the literature, two large groups of XAI can be identified: transparent algorithms and post-hoc explainability (Barredo Arrieta et al., 2020). Since our paper intends to provide explainability, and as we opted for an RL which is not transparent by nature (Heuillet et al., 2021)

ALGORITHM 1 **Algorithm of the trading agent in the simulated environment**

Require: Actions from the agent
Require: State space
Ensure: Final Balance
$N \leftarrow$ number of assets
$A \leftarrow$ set of actions from the agent
Prices $\leftarrow$ set of actual prices from the assets
PW $\leftarrow$ set of weights of the assets in the portfolio
Commission Rate $\leftarrow 0.005$ ▷ Define the commission rate
Balance $= \sum_{i=1}^{N} PW_i \cdot \text{Prices}_i$
Cash = Cash − Balance
Total Commissions = 0 ▷ Initialize total commissions
while Balance > 0 **do**
  for $i = 1$ to Episodes **do**
    if Episode = 0 **then** ▷ Initial Buy of stocks
      for $j = 1$ to N **do** ▷ Iteration through the different assets
        Buy n_j shares
        Commission $= n_j \cdot \text{Prices}_j \cdot$ Commission Rate
        Cash = Cash − $(n_j \cdot \text{Prices}_j + \text{Commission})$
        Total Commissions += Commission
        Initial portfolio = $\text{PW}_{ij}{}^{*}\text{Prices}_{ij}$
        Balance = Portfolio + Cash
      end for
    else ▷ Rebalance the portfolio
      for $j = 1$ to N **do** ▷ Iteration through the different assets
        $n_j \leftarrow$ number of shares different between A and PW
        if $A_{ij} < \text{PW}_{ij}$ **then** ▷ Selling phase
          SELL n_{ij} shares
          Commission $= n_{ij} \cdot \text{Prices}_{ij} \cdot$ Commission Rate
          Cash = Cash + $(n_{ij} \cdot \text{Prices}_{ij} - \text{Commission})$
          Total Commissions += Commission
          $\text{PW}_{ij} = A_{ij}$
        else if $A_{ij} > \text{PW}_{ij}$ and Cash > 0 **then** ▷ Buying phase
          BUY n_j shares
          Commission $= n_j \cdot \text{Prices}_j \cdot$ Commission Rate
          Cash = Cash − $(n_j \cdot \text{Prices}_j + \text{Commission})$
          Total Commissions += Commission
          $\text{PW}_{ij} = A_{ij}$ ▷ Update of the weights
        else
          Continue
        end if
      end for
    end if
  end for
end while
Final Balance $= \sum_{k=1}^{N} \text{PW}_k{}^{*}\text{Prices}_k + \text{Cash} - \text{Commissions}$ ▷ This value used as Reward

and the implantation of a post-hoc algorithm is impractical due to the intricate configuration of the network, we decided to add an attention layer to the agent architecture, between the inputs and the LSTM network, as shown in Figure 3.

An attention layer is a vector added to the policy network and helps elucidate each variable's weight and memorize long information concatenations. To compute attention att_k for each variable k in different time steps, we use a softmax function:

$$\text{att}_k = \text{softmax}(\text{FW}_k x_k), \tag{15}$$

where the input feature has a learned weight FW_k and x_k represents a single variable over time. Afterward, the inputs from the state vector for each individual share are weighted by the calculated attention vector and go to the LSTM as input y_k, where:

$$y_k = a_k \bigodot x_k. \tag{16}$$

Multiple studies have used attention layers, especially in computer-assisted decision support (Barredo Arrieta et al., 2020; Kaji et al., 2019). Figure 3 shows the representation of the addition of an attention layer to the general scheme of the agent previously shown in Figure 2. By adding this layer, it is possible to obtain the attention vector att_k to determine the relevance of each feature before passing the inputs to the LSTM. The details of the architecture of the ANN, marked as Model architecture I and II in Figure 2, are shown in full detail in Appendix A in Figures A1 and A2, respectively.

3.5 | Model overview

Integrating all components, the research we are addressing involves an RL agent whose objective is to maximize expected cumulative returns by optimizing the action-selection policy within the constraints of the financial market, represented by an MDP. The optimization thus, revolves around finding a policy $\pi^*(a_t|s_t)$ that maximizes the expected cumulative return, given by:

$$\pi_\theta^* = \arg\max_{\pi_\theta} \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t \mathcal{R}(s_t, a_t)|\pi_\theta\right], \tag{17}$$

where, $\gamma \in (0,1]$ is the discount factor, emphasizing the preference for immediate rewards over future rewards, and T represents the investment horizon. This formulation starts with a state space $\mathcal{S}$ as a comprehensive set of market indicators hypothesized to impact the price and, hence, the trading decisions, making $\mathcal{S}$ a multi-dimensional space.

Then, second, the action space $\mathcal{A}$ is formulated as a continuum of portfolio allocations across n assets, where each action $a_t \in \mathcal{A}$ at time t is a vector of weights $(w_1, w_2, ..., w_n)$ subject to the constraint $\sum_{i=1}^{n} w_i \leq 1$ and $0 \leq w_i \leq 1$ for each w_i. This represents the proportion of the total portfolio value allocated to each asset.

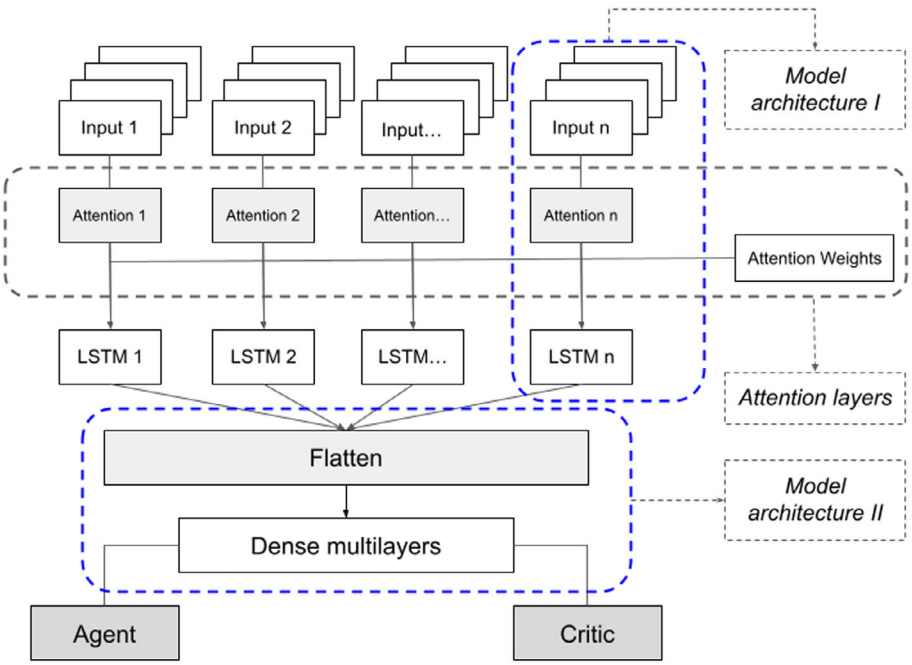

FIGURE 3 General scheme of the agent's architecture with an attention layer.

The reward function $\mathcal{R}(s_t, a_t)$ captures the immediate return from taking action a_t in state s_t and is directly related to the change in portfolio value δW_t as a result of this action. With the probabilistic nature of moving from one state to another after taking an action of $\mathcal{P}(s_{t+1}|s_t, a_t)$. Enhancing the model's explainability, attention mechanisms are integrated to identify which features within the state space $\mathcal{S}$ predominantly influence the selection of actions. This aligns with the goal of developing an automated trading system that not only maximizes returns but also ensures transparency and interpretability, meeting the growing need for XAI in finance.

To summarize, our approach utilizes RL techniques to develop a strategy focused on maximizing expected returns, as detailed in Section 3.1. This strategy is refined through the interactions between an agent, as outlined in Section 3.2, and a simulated environment, as presented in Section 3.3. Furthermore, incorporating attention mechanisms, as described in Section 3.4, enhances the model's ability to discern significant market features influencing trading decisions.

4 | EXPERIMENTATION

The research utilizes a comprehensive dataset spanning from 3 January 2005, to 31 December 2021, encompassing 4435 observations. The dataset exhibits a 100% completeness rate, with no missing values across the observation period. The data source is from a global provider of financial data called Bloomberg. The details of the dataset are shown in Table 1.

Once all the components were assembled and codified, a training process was carried out using 1000 episodes, each comprising a maximum of 200 steps, and each step was equivalent to 1 day of negotiation. The starting day of each of these episodes was chosen randomly within the training period. Since it is impossible to know the actual price at which the assets would be bought or sold in the market on a given day due to uncertain market conditions, a random value between the lowest and highest price of the respective day was taken to calculate the opening and closing price of the first and last day of the episode, for detailed configurations in this process, refer to Table 2.

Once a sample episode has been taken, it iterates through the trading days, where different states are given to the agent. Once received, the agent processes them through a neural network where two values are obtained: the score of the actor and the critic. Although these values let

TABLE 1 Summary of the data set.

Stock symbol	Mean close price	SD (price)	Min/max close price	Lowest price	Highest price
AIR	50.11	34.19	8.47/139.0	8.12	139.40
BNP	52.84	13.13	20.78/91.6	20.08	92.40
OR	143.66	84.34	46.96/429.8	46.00	433.65
TTE	44.42	6.41	21.8/63.05	21.12	63.40
EL	79.53	38.30	26.28/193.36	26.08	195.00
MC	179.36	152.75	35.32/734.7	34.34	741.60
KER	218.29	184.33	28.86/792.1	28.42	798.00
RMS	330.58	294.16	47.73/1675.5	47.03	1678.00
SAN	69.04	13.92	37.71/100.1	35.86	100.55
SU	60.25	26.96	20.15/173.18	19.42	173.78

TABLE 2 Parameter specifications for the RL model.

Hyperparameter	Value	Hyperparameter	Value
Number of LSTMs	10	Number of common layers	3
Node of LSTMs	32	Number of drop layers	3
Activation of LSTMs	Sigmoid	Nodes of common layers	250, 125, 250
Activation of common layers	LeakyReLU, Sigmoid	Values of drop layers	0.99, 0.8, 0.5
Final layer	Softmax	Optimizer	RMSprop
Learning rate	0.01	Loss function	Huber
Initial budget	1,000,000	Commission rate	0.5%
Number of inputs	10	Number of lagged variables	5
Input matrix size	10×5	Technical analysis variables	MACD, RSI
Price variables	Open, high, low close, volume	Moving averages periods	14, 21, 100

calculate the agent's performance in creating a profitable portfolio, they also allow calculating the return of a simulated investment in the sampled period, which will be used to calculate the agent's reward after the episode is finished. The episode's reward is divided by the initial investment, and then it is added to a running reward, as seen in Equation (18), which will be used for further network backpropagation.

Once the training has finished, the explanatory part of the model is examined to obtain information relevant to the importance of the variable. This begins by collecting the states of each asset, which are subsequently entered into the explanatory layer to perform a partial extraction of the model, obtaining the att_k values previously described above in Equation (15). Then, using these values from the explanatory layers, we obtain the softmax activation of a single state, disregarding the lagged variables for this analysis.

$$\text{running reward} = 0.05 \cdot \text{episode reward} + (1 - 0.05) \cdot \text{running reward}. \tag{18}$$

The environment, agent structure, simulation, and graphs were coded using the Python programming language,[3] and each neural network was coded using Keras[4] and Tensorflow.[5] The whole implementation was carried out in a Google cloud virtual machine instance and a general scheme of the training process can be seen in Figure 4.

5 | RESULTS

After training with 1000 episodes, it is possible to see the running reward progressively increasing. In order to test and prove the trading agent's efficiency, we took an out-of-sample period, where the agent is compared against a benchmark, which we defined as an equally weighted portfolio. Additionally, to compare the performance of our proposed portfolio, we added two additional architectures: one without an attention layer and another substituting an LSTM layer with a dense layer to assess their impact on the trading strategy's overall performance. Furthermore, to compare the results for our proposed portfolio further, we incorporated two variants of the Markowitz model: one static version and another that adjusts positions annually. The first model calculates the portfolio weights at the end of the training period, considering the past earnings and volatility to create a portfolio that will not change until the end of the testing set. The second Markowitz model, however, recalculates the portfolio weights at the end of each year during the out-of-sample period, using the most recent data to adjust its allocations. This dynamic approach allows the portfolio to adapt to changes in the market environment and is more realistic in reflecting on how portfolio managers often operate, continuously reassessing and adjusting their investment strategies to align with current market conditions and outlooks.

The RL agent, specifically the one we propose that utilizes LSTM and Attention mechanisms, displays the best performance as indicated by its total cumulative return of 66.70%, the lowest standard deviation of 0.0114, the shallowest maximum drawdown of −31.15%, and the highest Sharpe ratio of 0.98. which achieved a total return of 62.60%, a standard deviation of 0.0117, and a maximum drawdown of −34.72%, as shown in Table 3. Our proposed RL model not only demonstrates superior risk-adjusted returns but also shows enhanced resilience in market downturns. As shown in Figure 5, the variations of our RL model also outperform traditional and static portfolio management strategies, such as the Annual

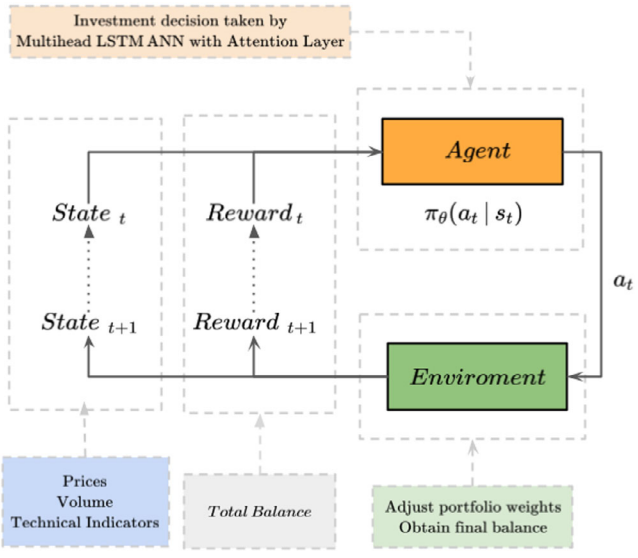

FIGURE 4 General scheme of the RL training process shown in this paper.

TABLE 3 Table with the main results from the experimentation in the out-of-sample test.

Portfolio	Total return	Standard deviation	Maximum drawdown	Sharpe ratio
RL with LSTM & attention	**0.6670**	**0.0114**	**−0.3115**	**0.98**
RL with LSTM Only	0.5908	0.0147	−0.3918	0.76
RL with dense & attention	0.6348	0.0144	−0.3848	0.77
CAC-40	0.3154	0.0129	−0.3855	0.60
Equally weighted	0.6260	0.0117	−0.3472	0.95
Annual rebalanced Markowitz	−0.0960	0.0197	−0.5950	−0.09
Static Markowitz	−0.1086	0.0147	−0.4548	−0.12

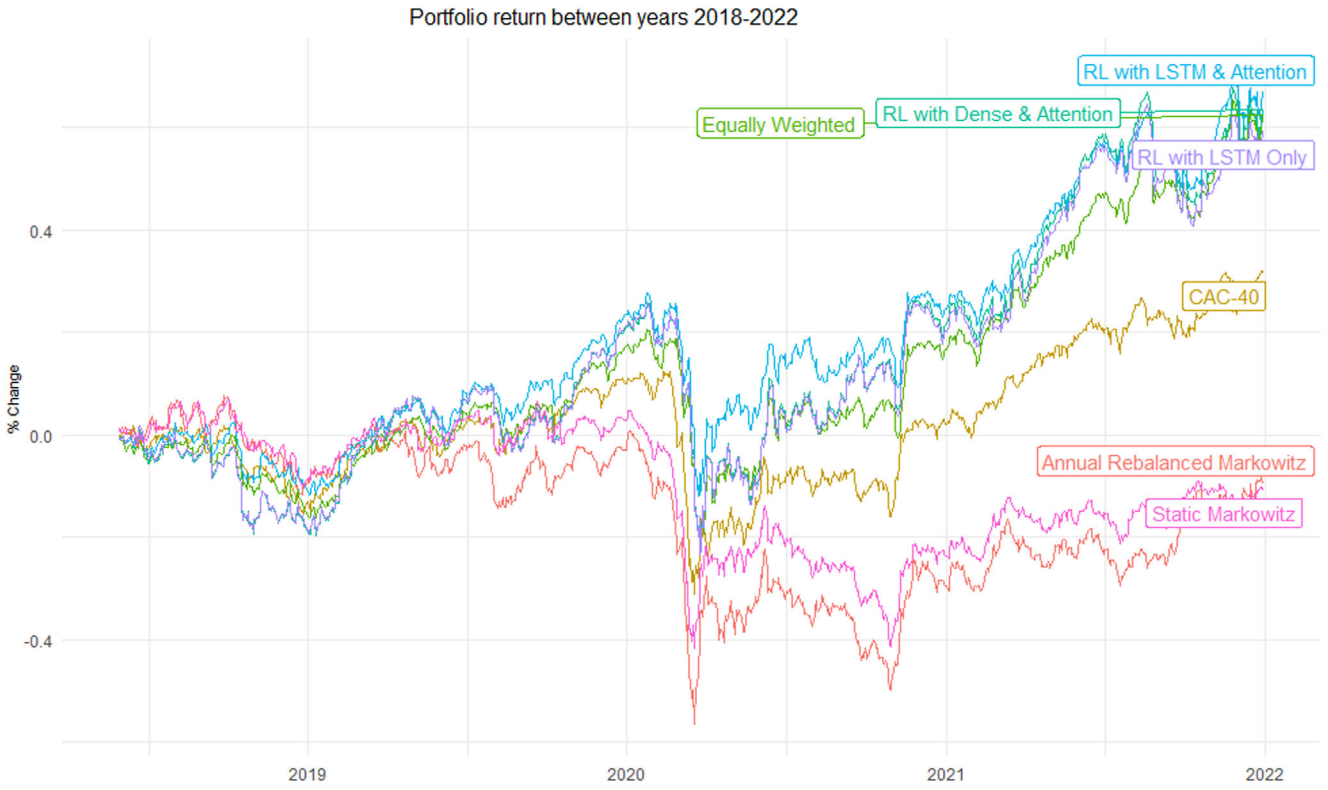

FIGURE 5 Performance of the agent in an out-of-sample period compared to the benchmark and the CAC-40 index.

Rebalanced Markowitz and Static Markowitz portfolios, which posted negative returns and higher volatility, highlighting the limitations of these methods in adapting to market fluctuations.

The LSTM Only, without the attention layer, is better than our benchmark, yet it was not as effective as the model incorporating both LSTM and Attention mechanisms. It secured a total return of 59.08%, slightly higher than the standard deviation of 0.0147, and encountered a deeper maximum drawdown of −39.18%. Meanwhile, the RL with a dense layer instead of an LSTM performed with a total return of 63.48%, with a standard deviation of 0.0144, a maximum drawdown of −38.48%, and a Sharpe ratio of 0.77.

The RL models have demonstrated superior performance compared to techniques associated with the Markowitz portfolios. The performance of the Annual Rebalanced Markowitz and Static Markowitz portfolios during the out-of-sample period reveals significant differences compared to the other strategies outlined in Table 3. The Annual Rebalanced Markowitz portfolio experienced a negative return of 9.60%, with a standard deviation of returns at 0.0197, the highest among all portfolios examined, indicating higher volatility and risk. Furthermore, its maximum drawdown reached −59.50% and a Sharpe ratio of −0.09.

Similarly, the Static Markowitz portfolio also encountered negative performance, with a total return of −10.86%, the lowest among all portfolios examined. Its standard deviation was 0.0147, suggesting a somewhat lower risk profile than the Annual Rebalanced Markowitz portfolio but still higher than most strategies analysed. The maximum drawdown for the Static Markowitz was −45.48%, and a Sharpe ratio of −0.12.

To explain the agent's most important variables, we obtain the data of our explanatory layer of attention located between each of the inputs and each of the networks of LSTM, as shown in Figure 3. In order to obtain this descriptive data, a call was made to the agent's neural network, where a partial extraction of each of the attention layers was performed using the data chosen as an out-of-sample. The main idea involves knowing which state values are the most important in the agent's decision-making process, and we can see that these vary according to the different assets and time.

The main advantage of mixing a multi-head structure with independent explanatory layers, like the one we presented in this paper, is that it allows us to extract relevant information from each LSTM independently and discover the critical variables for each asset. Additionally, it should be noted that the structure of the states given to the agent is three-dimensional because, at each step, each LSTM network is fed with ten variables together with its lagged values for four episodes, as explained in Section 4. However, in this research, for the purpose of analysing the explainable values coming from the attention vector att_k, we will only consider those values at time t, which we will call Q values. In this way, we will merely show the importance of the inputs without the lag variables; also, this facilitates the construction of flat two-dimensional schemes that are easier to analyse visually.

If we analyse the Q values, we can see that not all variables have the same importance in every asset, nor is their importance constant over time. For example, in the graphs for AIR and BNP in Figure 6, 14-day MA emerges as a significant indicator, reflecting short-term trends and momentum in pricing; therefore, it is possible to assume that the model captures that this stock is more sensitive to short-term fluctuations and uses this information to make more informed trading decisions. However, the BNP asset also appears to be affected by a broader spectrum of moving averages and other technical indicators, suggesting a more complex interplay of factors influencing its price. This complexity indicates that the RL model identifies simple trend-following strategies and integrates multiple indicators to assess the overall market context and asset-specific behaviours.

Similarly, the SAN asset considers the 21-day MA more important, followed by the high and closing values, as shown in Figure 7. This suggests a sensitivity to trends extending over 3 weeks. High and closing prices are also notable factors for SAN, indicating that the asset's price movement within a given day can substantially impact investment decisions.

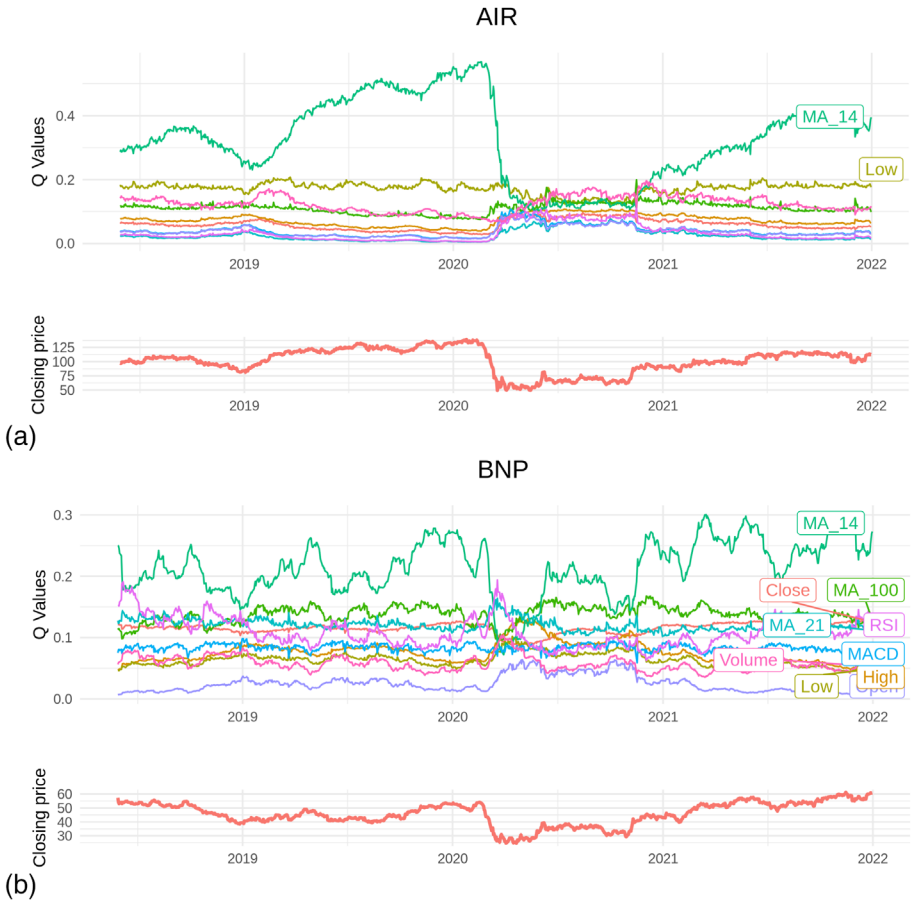

FIGURE 6 Graphical representation of Q values and closing prices in the out-of-sample period for (a) AIR, (b) BNP.

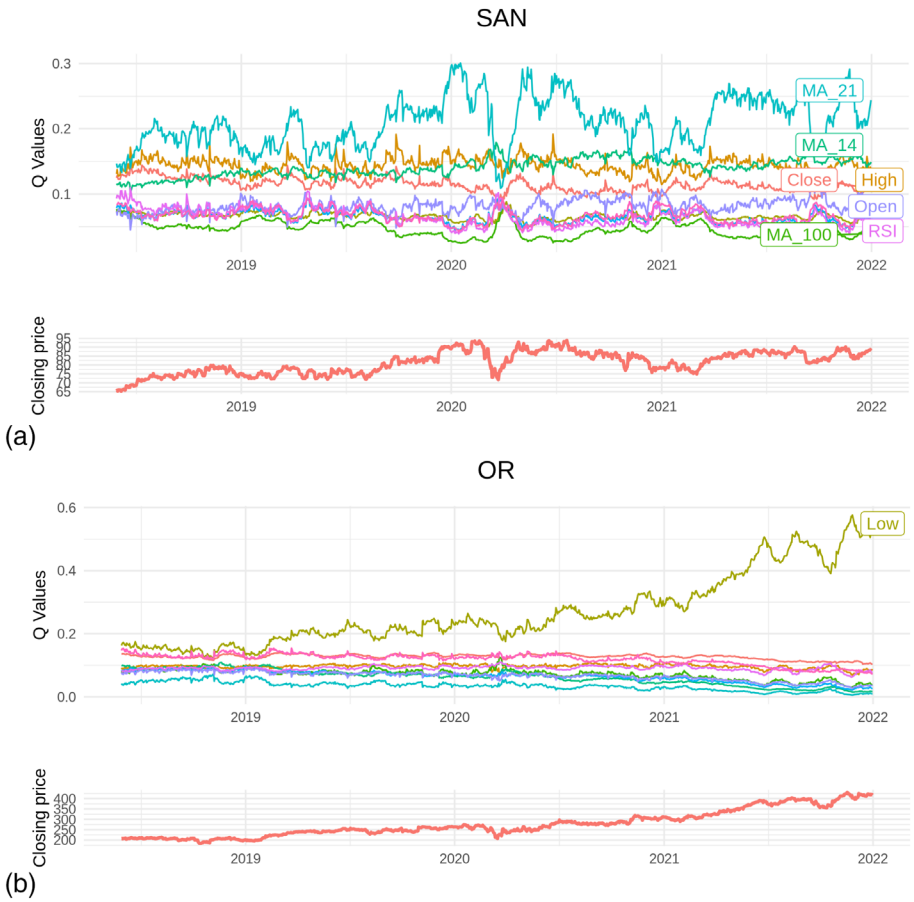

FIGURE 7 Graphical representation of Q values and closing prices in the out-of-sample period for (a) SAN, and (b) OR stock.

Also, we see that certain LSTMs in some assets consider the price and volume of the assets to be more important than the technical analysis data. For example, in Figure B1, we see SU, MC, and EL, respectively, illustrating that the model gives considerable weight to raw price and volume data. For EL, by observing the Q values from the given figure, the attention given to the closing price indicates a high value placed on the final price at which the asset settles at the end of the trading day. Following the closing price, the 21-day MA is also given significant importance, suggesting that the model values the trends and momentum established over a longer period. This may imply that the asset is influenced by medium-term trends, which could be representative of market cycles or recurring trading patterns.

In the case of MC, the results indicate an increasing significance of momentum, as measured by the RSI indicator, in the model's decision-making process. This emphasis on the MSI indicator suggests that the model is progressively utilizing it to identify overbought or oversold conditions as critical decision-making points. For SU, the prominence of volume in the Q values suggests a strong correlation between trading volume and price movement, which might indicate that volume spikes precede significant price changes. This can be a sign that the model considers market momentum and liquidity before making decisions, inferring that, in SU's case, volume is a leading indicator of market activity.

Analysis of Figure B2 further demonstrates the diversity in how the RL model assesses the value of different data types across various assets. The observed variability in data significance indicates the model's ability to recognize and assign importance to different data points uniquely for each asset. In the case of the remaining assets, indicators such as KER, RMS, and TTE exhibit distinct patterns of importance, further highlighting the model's refined grasp of the unique market dynamics associated with each asset. The KER considers the closing price the most critical variable, while RMS is the 21-period MA. This variation in indicator preference across KER and RMS underscores the RL model's capability to discern and adapt to the specific market behaviours and trends relevant to each asset. For KER, prioritizing the closing price may indicate a focus on the final market sentiment at the end of the trading day, which could be a key indicator of the asset's stability or volatility, as is for EL and SU.

On the other hand, RMS's emphasis on the 21-period MA as the most critical variable points to a strategic focus on medium-term trends. This pattern mirrors observations in SAN, alongside AIR and BNP, which also underscore the importance of medium-term trend indicators, notably identifying the 14-period MA as crucial. Finally, the TTE suggests a sophisticated combination of volume analysis, momentum tracking, and trend following, with adaptability to prioritize different indicators as their relevance shifts over time.

6 | CONCLUSIONS

The main goal of this research paper was to explore how an explainable RL application could create a profitable portfolio in the stock market. It is appropriate and challenging to determine the most predominant variables considered by an RL agent to assist investors, traders, or stakeholders in creating investment portfolios. Therefore, we have developed an explainable RL model in line with this goal that has proven to not only build a stock portfolio effectively with better performance than an equally weighted portfolio but also to show the most relevant features for each asset.

The use of RL can significantly improve the creation of investment portfolios due to its ability to continuously learn and add or remove different variables throughout its implementation, and it is not only influenced by a fixed set of variables. In addition, adding components that help explain the RL agent allows for greater confidence and auditing of certain decisions it may have to make over time. With this research, we have filled a gap in the financial literature by adding an explainable RL agent that can use DL to create portfolios and explain key features simultaneously.

Although our implementation and results propose critical and innovative insights into research and the financial markets, some limitations remain, as with other impactful research. From a practitioner's perspective, implementing an RL model to predict the stock market is challenging due to the market's complex nature. Key hurdles to implementing a trading strategy based on our model include handling vast amounts of data, the computational resources needed for real-time processing, dealing with regulatory obligations, and due diligence processes.

Additionally, the market's unpredictability, driven by factors beyond historical data, can make outcomes uncertain. Despite these difficulties, the potential benefits of optimized trading strategies make it a compelling venture for those in the field. However, we believe that the limitations that emerge from this work may benefit future research lines. First, our research only focuses on ten stocks in a specific market, such as the CAC-40; however, this exploration can be extended to different markets and regions, further expanding the capacity to create and diversify investment portfolios.

Second, our research has a limited exploration of the input variables taken by the agent and does not conduct an exhaustive review with multiple inputs that could affect the creation of an investment portfolio. The main advantage of ML and RF models is that numerous variables and a significant amount of data can be used. This flexibility also allows it to be extended to be used in Big Data applications, which utilize asset price data and other variables, such as financial market, macroeconomic, or alternative data. Finally, future research can validate our work by applying it to different markets in different periods; however, the application of attention-layered RL to explain the importance of variables in a financial RL application also offers a promising direction in future research.

Third, the implementation shown in this paper is primarily tailored for the risk-neutral investor and does not consider other types of investors, such as risk-averse investors. Therefore, future research can consider integrating various risk measures and preferences into the RL model to cater to different investor profiles by adjusting the reward structure to account for the degree of risk aversion and incorporating risk metrics.

ACKNOWLEDGEMENTS

This was supported in part by the NEOMA Business School under Grant 416004. The work of Daniel González Cortés, Laura Trinchera, and Jian Wu was supported by the Data Science for Insight and Value Creation, Research Group of the AE AI, Data Science and Business, NEOMA Business School.

FUNDING INFORMATION

This work was supported by the NEOMA Business School's AI, Data Science & Business Area of Excellence.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from Bloomberg. Restrictions apply to the availability of these data, which were used under license for this study. Data are available from the author(s) with the permission of Bloomberg.

ORCID

Daniel González Cortés https://orcid.org/0000-0002-5170-9883
Enrique Onieva https://orcid.org/0000-0001-9581-1823
Iker Pastor https://orcid.org/0000-0002-3068-6248
Laura Trinchera https://orcid.org/0000-0001-9679-0956
Jian Wu https://orcid.org/0000-0002-0855-1881

ENDNOTES

[1] A complete presentation of the variables involved in the agent and the environment phase are presented in Table 2.

[2] Brockman G, Cheung V, Pettersson L, Schneider J, Schulman J, Tang J, et al. Openai gym. 2016.

[3] G. Van Rossum and F. L. Drake Jr, Python reference manual, 1995.

[4] F. Chollet et al., Keras, https://keras.io, 2015.

[5] M. Abadi, A. Agarwal, P. Barham, et al., TensorFlow: Large-scale machine learning on heterogeneous systems, 2015.

REFERENCES

AbdelKawy, R., Abdelmoez, W. M., & Shoukry, A. (2021). A synchronous deep reinforcement learning model for automated multi-stock trading. *Progress in Artificial Intelligence*, *10*(1), 83–97.

Aboussalah, A. M., & Lee, C. G. (2020). Continuous control with stacked deep dynamic recurrent reinforcement learning for portfolio optimization. *Expert Systems with Applications*, *140*, 112891.

Amirshahi, B., & Lahmiri, S. (2023). Investigating the effectiveness of Twitter sentiment in cryptocurrency close price prediction by using deep learning. *Expert Systems*, e13428. https://doi.org/10.1111/exsy.13428

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., & Zaremba, W. (2017). Hindsight experience replay. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, & S. Vishwanathan (Eds.), *Advances in neural information processing systems* (Vol. 30). Curran Associates, Inc.

Arrow, K. J. (1974). *Essays in the theory of risk-bearing* (Vol. 121). North-Holland Amsterdam.

Bao, T., Corgnet, B., Hanaki, N., Riyanto, Y. E., & Zhu, J. (2023). Predicting the unpredictable: New experimental evidence on forecasting random walks. *Journal of Economic Dynamics and Control*, *146*, 104571.

Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, *58*, 82–115.

Betancourt, C., & Chen, W. H. (2021). Deep reinforcement learning for portfolio management of markets with a dynamic number of assets. *Expert Systems with Applications*, *164*, 114002.

Black, F., & Litterman, R. (1990). Asset allocation: Combining investor views with market equilibrium. *Goldman Sachs Fixed Income Research*, *115*, 7–18.

Brim, A. (2020). Deep reinforcement learning pairs trading with a double deep Q-network. In *2020 10th annual computing and communication workshop and conference (CCWC)* (pp. 0222–0227).

Cheong, D., Kim, Y. M., Byun, H. W., Oh, K. J., & Kim, T. Y. (2017). Using genetic algorithm to support clustering-based portfolio optimization by investor information. *Applied Soft Computing*, *61*, 593–602.

DeMiguel, V., Garlappi, L., & Uppal, R. (2009). Optimal versus naive diversification: How inefficient is the 1/N portfolio strategy? *The Review of Financial Studies*, *22*(5), 1915–1953.

Dempster, M. A. H., & Leemans, V. (2006). An automated FX trading system using adaptive reinforcement learning. *Expert Systems with Applications*, *30*(3), 543–552. Intelligent Information Systems for Financial Engineering.

Dempster, M. A. H., & Romahi, Y. S. (2002). Intraday FX trading: An evolutionary reinforcement learning approach. In *International conference on intelligent data engineering and automated learning* (pp. 347–358). Springer.

Deng, Y., Bao, F., Kong, Y., Ren, Z., & Dai, Q. (2017). Deep direct reinforcement learning for financial signal representation and trading. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(3), 653–664.

Dikmen, M., & Burns, C. (2022). The effects of domain knowledge on trust in explainable AI and task performance: A case of peer-to-peer lending. *International Journal of Human-Computer Studies*, *162*, 102792.

Dixon, M. F., Halperin, I., & Bilokon, P. (2020). *Applications of reinforcement learning* (pp. 347–418). Springer International Publishing.

El Qadi, A., Trocan, M., Díaz-Rodríguez, N., & Frossard, T. (2022). Feature contribution alignment with expert knowledge for artificial intelligence credit scoring. *Signal, Image and Video Processing*, *17*, 427–434.

Fang, M., Zhou, C., Shi, B., Gong, B., Xi, W., & Wang, T. (2019). DHER: Hindsight experience replay for dynamic goals. In *International conference on learning representations*. IEEE.

Fengqian, D., & Chao, L. (2020). An adaptive financial trading system using deep reinforcement learning with candlestick decomposing features. *IEEE Access*, *8*, 63666–63678.

Ge, B., Hipel, K. W., Fang, L., Yang, K., & Chen, Y. (2014). An interactive portfolio decision analysis approach for system-of-systems architecting using the graph model for conflict resolution. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *44*(10), 1328–1346.

Goodell, J. W., Kumar, S., Lim, W. M., & Pattnaik, D. (2021). Artificial intelligence and machine learning in finance: Identifying foundations, themes, and research clusters from bibliometric analysis. *Journal of Behavioral and Experimental Finance*, *32*, 100577.

Grądzki, P., & Wójcik, P. (2023). Is attention all you need for intraday forex trading? *Expert Systems*, *41*(2), e13317.

Greydanus, S., Koul, A., Dodge, J., & Fern, A. (2018). Visualizing and understanding atari agents. In *International conference on machine learning* (pp. 1792–1801). PMLR.

Guo, S. S., Zhang, R., Liu, B., Zhu, Y., Ballard, D., Hayhoe, M., Ballard, D., & Stone, P. (2021). Machine versus human attention in deep reinforcement learning tasks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. S. Liang, & J. W. Vaughan (Eds.), *Advances in neural information processing systems* (Vol. 34, pp. 25370–25385). Curran Associates, Inc.

Harvey, C. R., Liechty, J. C., Liechty, M. W., & Müller, P. (2010). Portfolio selection with higher moments. *Quantitative Finance*, *10*(5), 469–485.

Heaton, J. B., Polson, N. G., & Witte, J. H. (2017). Deep learning for finance: Deep portfolios. *Applied Stochastic Models in Business and Industry*, *33*(1), 3–12.

Heuillet, A., Couthouis, F., & Díaz-Rodríguez, N. (2021). Explainability in deep reinforcement learning. *Knowledge-Based Systems*, *214*, 106685.

Hirchoua, B., Ouhbi, B., & Frikh, B. (2021). Deep reinforcement learning based trading agents: Risk curiosity driven learning for financial rules-based policy. *Expert Systems with Applications*, *170*, 114553.

Hoepner, A. G., McMillan, D., Vivian, A., & Wese Simen, C. (2021). Significance, relevance and explainability in the machine learning age: An econometrics and financial data science perspective. *The European Journal of Finance*, *27*(1-2), 1–7.

Jagannathan, R., & Ma, T. (2003). Risk reduction in large portfolios: Why imposing the wrong constraints helps. *The Journal of Finance*, *58*(4), 1651–1683.

Jeong, G., & Kim, H. Y. (2019). Improving financial trading decisions using deep Q-learning: Predicting the number of shares, action strategies, and transfer learning. *Expert Systems with Applications*, *117*, 125–138.

Juozapaitis, Z., Koul, A., Fern, A., Erwig, M., & Doshi-Velez, F. (2019). Explainable reinforcement learning via reward decomposition. In *IJCAI/ECAI workshop on explainable artificial intelligence*. IEEE.

Kaji, D. A., Zech, J. R., Kim, J. S., Cho, S. K., Dangayach, N. S., Costa, A. B., & Oermann, E. K. (2019). An attention based deep learning model of clinical events in the intensive care unit. *PloS One*, *14*(2), e0211057.

Kamalov, F. (2020). Forecasting significant stock price changes using neural networks. *Neural Computing and Applications*, *32*(23), 17655–17667.
Kamruzzaman, M. M., Alruwaili, O., & Aldaghmani, D. (2024). Measuring systemic and systematic risk in the financial markets using artificial intelligence. *Expert Systems*, *41*, e12971.
Krajna, A., Kovac, M., Brcic, M., & Šarčević, A. (2022). Explainable artificial intelligence: An updated perspective. In *2022 45th jubilee international convention on information, communication and electronic technology (MIPRO)* (pp. 859–864). IEEE.
Kuiper, O., van den Berg, M., van der Burgt, J., & Leijnen, S. (2022). Exploring explainable AI in the financial sector: Perspectives of banks and supervisory authorities. In L. A. Leiva, C. Pruski, R. Markovich, A. Najjar, & C. Schommer (Eds.), *Artificial intelligence and machine learning* (pp. 105–119). Springer International Publishing.
Kute, D. V., Pradhan, B., Shukla, N., & Alamri, A. (2021). Deep learning and explainable artificial intelligence techniques applied for detecting money laundering—A critical review. *IEEE Access*, *9*, 82300–82317.
Lai, Z. R., Yang, P. Y., Fang, L., & Wu, X. (2020). Reweighted price relative tracking system for automatic portfolio optimization. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *50*(11), 4349–4361.
Langer, M., Oster, D., Speith, T., Hermanns, H., Kästner, L., Schmidt, E., Sesing, A., & Baum, K. (2021). What do we want from explainable artificial intelligence (XAI)?—A stakeholder perspective on XAI and a conceptual model guiding interdisciplinary XAI research. *Artificial Intelligence*, *296*, 103473.
Lesort, T., Díaz-Rodríguez, N., Goudou, J. F., & Filliat, D. (2018). State representation learning for control: An overview. *Neural Networks*, *108*, 379–392.
Li, J., Liu, G., Yan, C., & Jiang, C. (2017). Robust learning to rank based on portfolio theory and AMOSA algorithm. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *47*(6), 1007–1018.
Li, Y., Zheng, W., & Zheng, Z. (2019). Deep robust reinforcement learning for practical algorithmic trading. *IEEE Access*, *7*, 108014–108022.
Lim, Q. Y. E., Cao, Q., & Quek, C. (2022). Dynamic portfolio rebalancing through reinforcement learning. *Neural Computing and Applications*, *34*(9), 7125–7139.
Lo, A. W., & MacKinlay, A. C. (1988). Stock market prices do not follow random walks: Evidence from a simple specification test. *The Review of Financial Studies*, *1*(1), 41–66.
Ma, Y., Han, R., & Wang, W. (2021). Portfolio optimization with return prediction using deep learning and machine learning. *Expert Systems with Applications*, *165*, 113973.
Madumal, P., Miller, T., Sonenberg, L., & Vetere, F. (2020). Explainable reinforcement learning through a causal lens. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 34, pp. 2493–2500). IEEE.
Mandeep, A. A., Bhatia, A., Malhi, A., Kaler, P., & Pannu, H. S. (2022). Machine learning based explainable financial forecasting. In *2022 4th international conference on computer communication and the internet (ICCCI)* (pp. 34–38). IEEE.
Markowitz, H. (1952). Portfolio selection. *The Journal of Finance*, *7*(1), 77–91.
Millea, A. (2021). Deep reinforcement learning for trading—A critical survey. *Data*, *6*(11), 119.
Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., & Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In M. F. Balcan & K. Q. Weinberger (Eds.), *Proceedings of the 33rd international conference on machine learning, vol. 48 of proceedings of machine learning research* (pp. 1928–1937). PMLR.
Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.
Moody, J., & Saffell, M. (2001). Learning to trade via direct reinforcement. *IEEE Transactions on Neural Networks*, *12*(4), 875–889.
Moody, J., Wu, L., Liao, Y., & Saffell, M. (1998). Performance functions and reinforcement learning for trading systems and portfolios. *Journal of Forecasting*, *17*(5-6), 441–470.
Ohana, J. J., Ohana, S., Benhamou, E., Saltiel, D., & Guez, B. (2021). Explainable AI (XAI) models applied to the multi-agent environment of financial markets. In *Explainable and transparent AI and multi-agent systems: Third international workshop, EXTRAAMAS 2021, virtual event, May 3–7, 2021. Revised selected papers* (pp. 189–207). Springer-Verlag.
Ozbayoglu, A. M., Gudelek, M. U., & Sezer, O. B. (2020). Deep learning for financial applications: A survey. *Applied Soft Computing*, *93*, 106384.
Paiva, F. D., Cardoso, R. T. N., Hanaoka, G. P., & Duarte, W. M. (2019). Decision-making for financial trading: A fusion approach of machine learning and portfolio selection. *Expert Systems with Applications*, *115*, 635–655.
Pendharkar, P. C., & Cusatis, P. (2018). Trading financial indices with reinforcement learning agents. *Expert Systems with Applications*, *103*, 1–13.
Perold, A. F., & Sharpe, W. F. (1988). Dynamic strategies for asset allocation. *Financial Analysts Journal*, *44*(1), 16–27.
Perrin, S., & Roncalli, T. (2020). Machine learning optimization algorithms & portfolio allocation. In *Machine learning for asset management: New developments and financial applications* (pp. 261–328). MDPI
Pratt, J. W. (1964). Risk aversion in the small and in the large. *Econometrica*, *32*(1/2), 122–136.
Sachan, S., Yang, J. B., Xu, D. L., Benavides, D. E., & Li, Y. (2020). An explainable AI decision-support-system to automate loan underwriting. *Expert Systems with Applications*, *144*, 113100.
Sattarov, O., Muminov, A., Lee, C. W., Kang, H. K., Oh, R., Ahn, J., Oh, H. J., & Jeon, H. S. (2020). Recommending cryptocurrency trading points with deep reinforcement learning approach. *Applied Sciences*, *10*(4), 1506.
Sequeira, P., & Gervasio, M. (2020). Interestingness elements for explainable reinforcement learning: Understanding agents' capabilities and limitations. *Artificial Intelligence*, *288*, 103367.
Shavandi, A., & Khedmati, M. (2022). A multi-agent deep reinforcement learning framework for algorithmic trading in financial markets. *Expert Systems with Applications*, *208*, 118124.
Sutton, R. S., McAllester, D., Singh, S., & Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, & K. Müller (Eds.), *Advances in neural information processing systems* (Vol. 12). MIT Press.
Syu, J. H., Wu, M. E., & Ho, J. M. (2020). Portfolio management system with reinforcement learning. In *2020 IEEE international conference on systems, man, and cybernetics (SMC)* (pp. 4146–4151). IEEE.
Taghian, M., Asadi, A., & Safabakhsh, R. (2022). Learning financial asset-specific trading rules via deep reinforcement learning. *Expert Systems with Applications*, *195*, 116523.

Thakkar, A., & Chaudhari, K. (2021). A comprehensive survey on portfolio optimization, stock price and trend prediction using particle swarm optimization. *Archives of Computational Methods in Engineering*, *28*(4), 2133–2164.

Théate, T., & Ernst, D. (2021). An application of deep reinforcement learning to algorithmic trading. *Expert Systems with Applications*, *173*, 114632.

Tütüncü, R. H., & Koenig, M. (2004). Robust asset allocation. *Annals of Operations Research*, *132*(1), 157–187.

Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. *Machine Learning*, *8*(3), 279–292.

Wu, X., Chen, H., Wang, J., Troiano, L., Loia, V., & Fujita, H. (2020). Adaptive stock trading strategies with deep reinforcement learning methods. *Information Sciences*, *538*, 142–158.

Yang, H., Liu, X. Y., Zhong, S., & Walid, A. (2020). Deep reinforcement learning for automated stock trading: An Ensemble strategy. In *Proceedings of the first ACM international conference on AI in finance ICAIF'20*. Association for Computing Machinery.

Zarkias, K. S., Passalis, N., Tsantekidis, A., & Tefas, A. (2019). Deep reinforcement learning for financial trading using price trailing. In *ICASSP 2019 – 2019 IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 3067–3071). IEEE.

Zhang, C., Cho, S., & Vasarhelyi, M. (2022). Explainable artificial intelligence (XAI) in auditing. *International Journal of Accounting Information Systems*, *46*, 100572. 2021 Research Symposium on Information Integrity & Information Systems Assurance.

Zhang, X., Ma, Y., & Wang, M. (2023). An attention-based logistic-CNN-BiLSTM hybrid neural network for credit risk prediction of listed real estate enterprises. *Expert Systems*, *41*(2), e13299.

Zhang, Z., Zohren, S., & Roberts, S. (2020). Deep reinforcement learning for trading. *The Journal of Financial Data Science*, *2*(2), 25–40.

AUTHOR BIOGRAPHIES

Daniel González Cortés is a Research Fellow at NEOMA Business School and a Ph.D. candidate at NEOMA and the University of Deusto. His research fuses Finance with Machine Learning and Artificial Intelligence, specializing in solving financial problems with Deep Learning, Reinforcement learning, and Explainable models. His interests are interdisciplinary research in Computer Science, Information Systems, Finance, and Management.

Enrique Onieva received his B.E. degree in computer science engineering, his M.E. degree in soft computing and intelligent systems, and his Ph.D. in computer science from the University of Granada, Spain, in 2006, 2008, and 2011, respectively. From 2007 to 2012, he has been with the AUTOPIA Program at the Centre of Automation and Robotics, Consejo Superior de Investigaciones Científicas, Madrid, Spain. In 2012, he joined the Models of Decision and Optimization group at the University of Granada. Since 2013, he has been a professor in artificial intelligence at the University of Deusto and a Researcher of Intelligent Transportation Systems applications within the Deusto Smart Mobility Research Unit. He has participated in more than 40 research projects. Among them are CYBERCARS-2 (FP6), ICSI (FP7), and PostLowCit (CEF-Transport). Research responsible for the Artificial Intelligence Work Package of Project TIMON (H2020) and Project Coordinator of the LOGISTAR Project (H2020). He has authored more than 100 scientific articles. From them, more than 40 are published in journals of the highest level. His research interest is based on applying Artificial Intelligence to Intelligent Transportation Systems, including fuzzy-logic-based decisions, evolutionary optimization, and machine learning.

Iker Pastor received a degree in computer engineering in 2007, a master's degree in information security in 2010, and a Ph.D. degree (cum laude) in computer science in 2013. He participated in the Program in Big Data and Business Intelligence, in 2016. He is with Deusto University and focuses its scientific interests on the areas of big data analytics, opinion mining, and computer vision. He is the author of several scientific articles reviewed by peers in conferences and indexed journals. He has participated in the gestation, scientific development, and technical development of numerous competitive projects and contracts with companies, the latter with several successful cases of knowledge transfer actions. In addition, he is a member of the Scientific Committee of several congresses, such as CISIS, SOCO, and ICEUTE. He is a Reviewer of many journals, included in the JCR as the magazine of Engineering and Industry–DYNA.

Laura Trinchera is an Associate Professor of Statistics, NEOMA Business School. Holds a Master's degree in Economic and Business Sciences (2004) and a Ph.D. in Statistics (2008) from the University of Naples Federico II, Italy. Her research focuses on Data Sciences and Statistical Learning methods, with a focus on Structural Equation Modeling, PLS Methods, and classification algorithms. Her research has been published in internationally recognized journals such as Structural Equation Modeling: A Multidisciplinary Journal, Journal of Production Economics, Journal of Organizational Behavior, Recherche et Applications en Marketing, International Journal of Information Management and Management Decision. Also contributed to the Handbook of Partial Least Squares: Concepts, Methods and Applications. Visiting researcher at the University of California Santa Barbara, the University of Michigan at Ann Arbor, the University of Hamburg, the Charles University of Prague, and HEC Paris. Guest lecturer at ESSEC Business School, Sciences Po Paris, and Sorbonne University in Abu Dhabi.

Jian Wu holds a Ph.D. from the University of Paris Dauphine and has vast experience teaching Investments, Financial Risk Management, Sustainable Finance, Economic Environment, and its impact on Financial Markets in initial training and executive education courses. Additionally,

she served over a decade as head of the Finance and Economics Department at NEOMA Business School. She has published her research work in journals such as Finance, International Review of Financial Analysis, Economic Bulletin, and Journal of Business Ethics. Her research interests are Financial Engineering, Corporate Governance, Banking Regulation, and Corporate Social Responsibility.

How to cite this article: Cortés, D. G., Onieva, E., Pastor, I., Trinchera, L., & Wu, J. (2024). Portfolio construction using explainable reinforcement learning. *Expert Systems*, *41*(11), e13667. https://doi.org/10.1111/exsy.13667

APPENDIX A

This appendix shows the detailed structures for model architecture I and II, shown in Figure 3. These images are part of the graphical representation of the ANN model created by in Python by the Keras library.

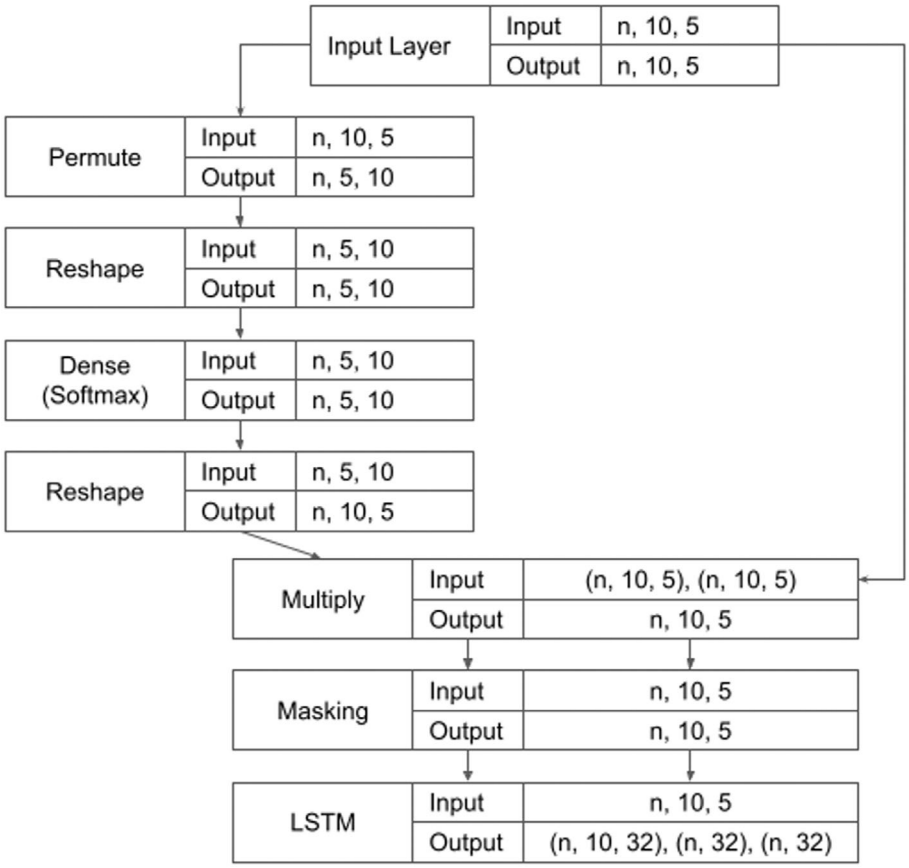

FIGURE A1 General scheme of the agent's architecture with an attention layer.

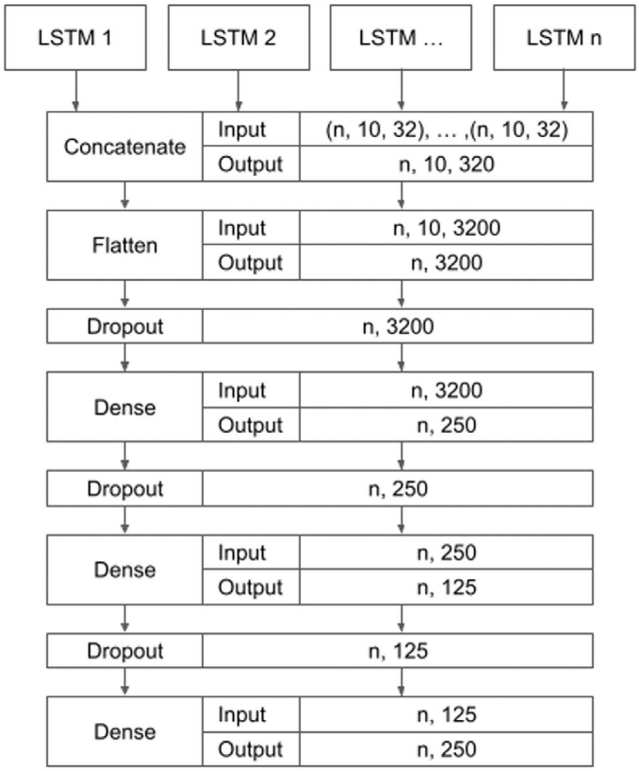

FIGURE A2 General scheme of the agent's architecture with an attention layer.

APPENDIX B

This appendix shows the graphs that display Q values and closing prices in the out-of-sample period that are not shown in the previous Section 5. The graphs are for the assets EL, MC, SU, KER, RMS, and TTE.

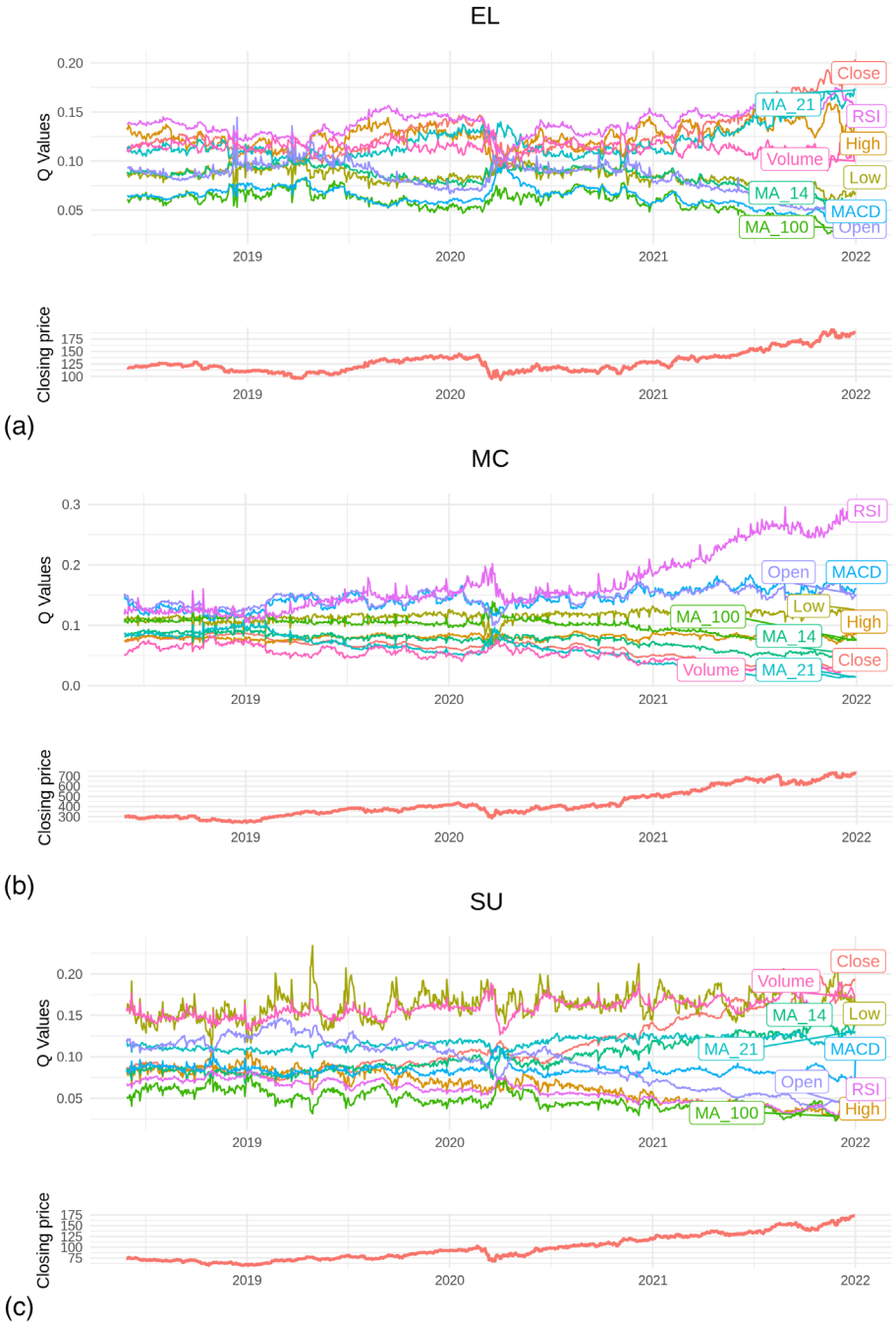

FIGURE B1 Graphical representation of Q values and closing prices in the out-of-sample period for (a) EL (b) MC, and (c) SU stock.

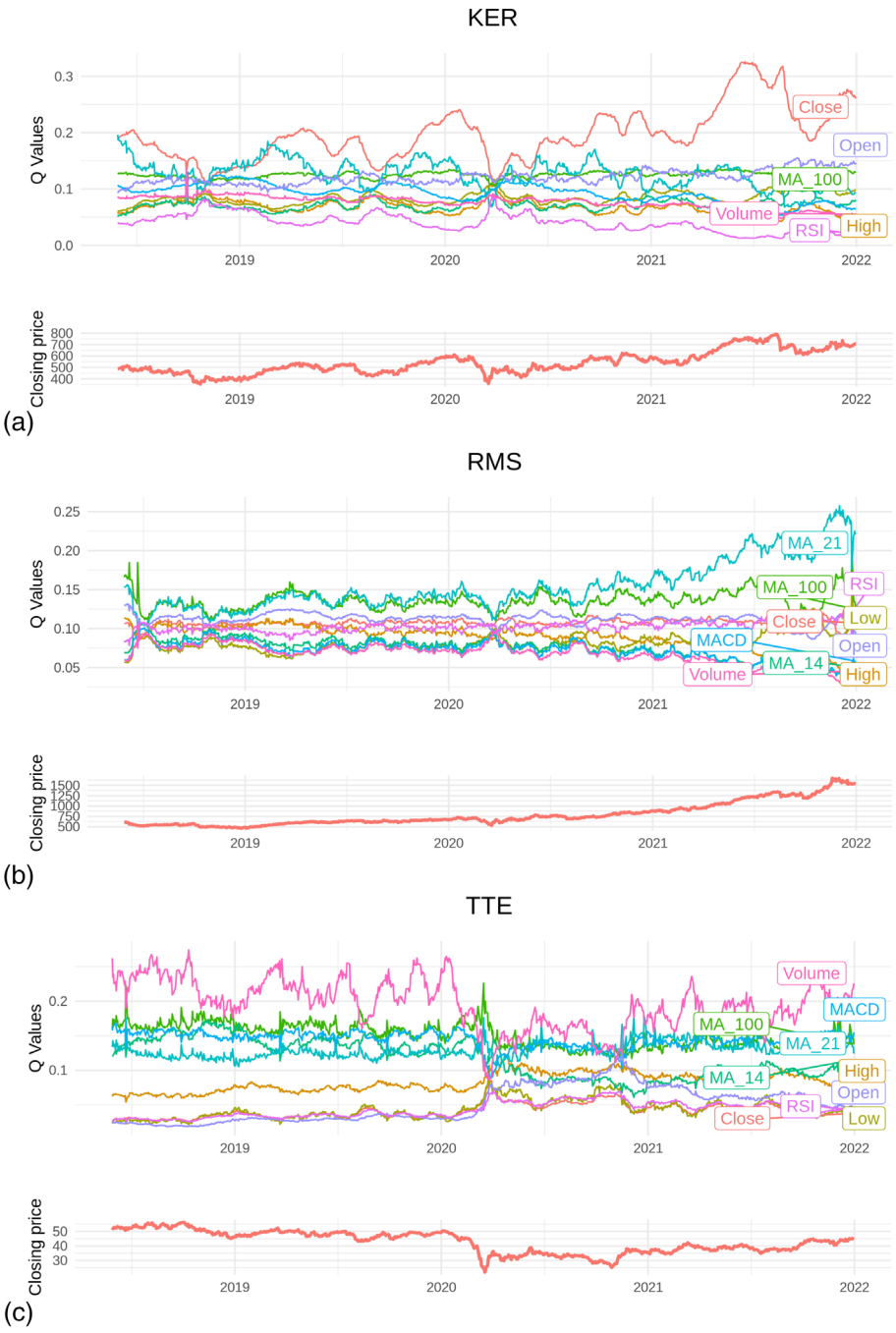

FIGURE B2 Graphical representation of Q values and closing prices in the out-of-sample period for (a) KER (b) RMS, and (c) TTE stock.