Engineering Applications of Artificial Intelligence 110 (2022) 104751

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Deep Reinforcement Learning with the Random Neural Network

Will Serrano

The Bartlett, University College London, United Kingdom

ARTICLE INFO

Keywords:
Deep Reinforcement Learning
Random Neural Network
Trend predictions
Value predictions

ABSTRACT

This paper proposes a Deep Reinforcement Learning (DRL) algorithm that expands the Random Neural Network (RNN) Reinforcement Learning (RL) method to include the previous learnings entirely from previous rewards, rather than only the actual one. The Random Neural Network weighs are updated with the current reward and the previous values, including time and memory. This addition makes DRL slower to make decisions, although it also increases its performance in some experiments. Several configurations to introduce DRL, such as sampling rate and memory duration, are also proposed and analysed in this article. The proposed DRL algorithm is included in a decision process that predicts trends: upward, downward and equal market directions in addition to values. Experimental results based on market prices demonstrate that the addition of Deep Learning to the Reinforcement Learning algorithm increases its performance slightly in some experiments; however, it also increases its computational cost. In random environments such as the stock market, it is preferable to make decisions based on the previous values (short memory) rather than historical records (long memory).

1. Introduction

Reinforcement Learning (RL) analyses the interaction between the action of agents and the subsequent reward they obtain from the environment. RL algorithms have been widely applied in Artificial Intelligence (AI) due to their unsupervised properties (Kaelbling et al., 1996): agents autonomously learn the best decisions to optimise a goal function balancing exploration with exploitation where the environment is typically defined as a Markov decision process. Deep Learning (DL) enables RL to scale decision making solutions that were previously unmanageable. Several methods such as Double DQN (Deep Q Network) include Deep Neural Networks to generalise an arbitrary function approximation (Hasselt et al., 2016); although DQN solves problems with high dimensional observation spaces, it can only manage discrete and low-dimensional action spaces. This effect is due to its dependency on finding the action that maximises the value function requires an iterative optimisation process at each step in the case of continuous values (Lillicrap et al., 2016).

Another approach for Deep Reinforcement Learning (DRL) is the asynchronous execution of multiple agents in parallel on multiple environment instances (Mnih et al., 2016). This parallelism decorrelates the agent's data into a higher stationary process using gradient descent to optimise Deep Neural Network controllers. To use RL successfully in situations close to real-world complexity, agents must derive efficient representations of the environment from high-dimensional sensory inputs and generalise this experience to new situations (Mnih et al., 2015). Humans and biological organisms solve this problem through a combination of RL and hierarchical sensory processing systems. End-to-end learning of communication protocols in complex environments present issues based on partial observability. Two approaches based on centralised learning but decentralised execution use the sense and act to maximise their shared utility (Foerster et al., 2016). (1) Reinforced inter-agent learning applies Deep Q-Learning, and (2) differentiable inter-agent learning exploits the fact that, during learning, agents can backpropagate error derivatives through noisy communication channels. Two less addressed issues of DRL are the lack of a generalisation capability to new target goals and data inefficiency as it requires several and costly episodes of trial and error to converge, which makes it impractical to be applied to real-world scenarios (Zhu et al., 2017). To address the first issue, an actor–critic model allows to better generalise via a policy that is a function of the goal and the current state. To address the second issue, an environment with high-quality 3D scenes and a physics engine enable agents to take actions and interact with objects to collect efficiently a significant number of training samples with end-to-end trainable. This process does not need feature engineering, feature matching between frames or 3D reconstruction of the environment.

1.1. Research motivation

This paper proposes a DRL algorithm that expands the Random Neural Network (RNN) Reinforcement Learning algorithm to include previous learnings entirely from previous rewards, rather than only the

E-mail address: w.serrano@ucl.ac.uk.

https://doi.org/10.1016/j.engappai.2022.104751
Received 1 August 2021; Received in revised form 1 February 2022; Accepted 5 February 2022
Available online 22 February 2022

actual one. The RNN (Gelenbe, 1993a,b, 2004) is a recurrent stochastic mathematical model of an interconnected neural network of neurons that exchange information via spiking signals. Excitatory spikes increase the potential of the receiving neuron, whereas inhibitory spikes decrease it. Neurons fire spikes at random only when their potential is positive. The Random Neural Network represents more closely how signals are transmitted in many biological neural networks where they travel as spikes or impulses, rather than analogue signal levels.

The RNN weights are updated with the current reward and the previous values incorporating time and memory in the proposed DRL algorithm. In addition, several configurations that introduce DRL, such as sampling rate and memory duration, are also proposed and analysed. Finally, the proposed learning algorithm is included in a decision process that predicts trends: upward, downward and equal market directions in addition to values. Experimental results based on market prices demonstrate that the addition of Deep Learning to the Reinforcement Learning algorithm slightly increases its performance in some experiments, increasing its computational cost.

1.2. Research structure

Section 2 of this paper presents the DRL algorithm related work based on algorithms and their use in predictions and other applications. Section 3 defines the DRL model based on the Random Neural Network. The validation of the proposed model against market prices (Bitcoin, UK house prices, FTSE, Gold and Brent oil) along with the obtained experimental results are shown in Section 4. Finally, conclusions and future research directions are shared in Section 5.

2. Related work

2.1. Deep Reinforcement Learning algorithms

A dual neural network architecture that models free RL consists of two separate estimators: the state value function and the state dependent action advantage function (Wang et al., 2016). The two streams are combined via a particular aggregating layer that estimates the state action of the value function. Continuous simple actions, a high state with action dimensionality control, tasks with partial observations and tasks with a hierarchical structure are benchmarked where challenges posed by reproducibility, experimental techniques, and reporting procedures of DRL methods are presented within the reported metrics (Duan et al., 2016). The results are compared against standard baselines to suggest guidelines to make future results more reproducible (Henderson et al., 2018).

A deep Q-Network can learn successful policies directly from high dimensional sensory inputs using end to end RL. A learning expressive energy-based policies for continuous states and actions apply a soft Q-learning method based on learning maximum entropy policies to express the optimal policy via a Boltzmann distribution (Haarnoja et al., 2017). An h-hierarchical-DQN (h-DQN) framework integrates hierarchical action-value functions, operating at different temporal scales, with a goal-driven intrinsically motivated DRL (Kulkarni et al., 2016). A top-level q-value function learns a policy over intrinsic goals, while a lower-level function learns a policy over atomic actions to satisfy flexible goals specifications. There have been six extensions to the DQN algorithm: double-q learning, prioritised replay, duelling networks, multi-step learning, distributional reinforcement learning and noisy sets; however, it is unclear if these extensions are complementary and can be successfully combined (Hessel et al., 2018). Two actor–critic DRL methods adjust agent behaviours to efficiently achieve a designated goal by adopting a weighted-sum scalarisation of different objective functions (Nguyen et al., 2021). A multi-critic single-policy creates a human-centric strategy that corresponds to a predefined priority weight of different objectives. In contrast, a single-critic multi-policy can generate a mixed plan based on a set of priority weights to prioritise objectives in real-time dynamically.

2.2. Deep Reinforcement Learning in predictions

A DL based stock market prediction model includes investors' emotional tendency to improve accuracy (Jin et al., 2020). Specifically, the sentiment is analysed on many stock market comments, which are classified into either bullish or bearish. A machine learning algorithm forecasts stocks of time series data with a fine-tuned version of support vector regression (Kumar et al., 2021). The grid search technique is applied over a training dataset to select the best kernel function and optimise its parameters, volatility nature, and associated risk. A time-series predictive model is fed from additive decomposition as a preprocessor to a deep echo state network instead of the raw time series (Kim and King, 2020). The three variables are obtained from the additive decomposition: seasonality, trend, and residual.

Financial product price data is treated as a one-dimensional series generated by the projection into the time dimension of a chaotic system composed of multiple factors. A deep neural network based prediction model that forecasts stock prices is designed based on the phase space reconstruction method and a Long Short-Term Memory (LSTM) network (Yu and Yan, 2020). The potential of LSTM networks for traffic flow prediction models is analysed in terms of the effect from the different sized training sets and a clustering strategy that adjusts the training set size (Dogan, 2021). The clustering algorithms included K-nearest neighbour, support vector machine, logistic regression and pattern recognition networks. An LSTM deep neural network models and predicts the financial transaction data of the Shanghai stock exchange where the three factors that affect the prediction accuracy are (1) the choice of input features, (2) the sequence length of time series, and (3) the influence of the number of samples on the prediction accuracy of the model (Yan et al., 2021). Various state-of-the-art DL methods such as a deep neural network, a convolutional neural network, a deep residual network are studied, combined and compared for Bitcoin price predictions (Ji et al., 2019). Experimental results show that LSTM-based prediction models slightly outperform the other models for Bitcoin price predictions (regression), whereas DNN-based models perform best for price trend prediction (classification). A new DL forecasting model for the accurate prediction of gold price and movement combines convolutional layers' ability to extract valuable knowledge and learn the internal representation of time-series data with the effectiveness of LSTM layers to identify short-term and long-term dependencies (Livieris et al., 2020).

The placement of a limit order within a given time horizon and integrating this process into an end-to-end learning pipeline is optimised via a DRL agent (MA and Liu, 2019). The agent learns the policy of the limit order placement via features extracted from market data related to movements of the Bitcoin/USD trading. A DRL agent based on an LSTM algorithm learns temporal patterns in data and automatically trades according to the current market condition and historical measurements (Wu et al., 2019). The goal of the agent is to maximise the ultimate profit in quantitative trading. An integration of RL and DL in a recurrent neural network weighs the influences of earlier states and actions on policy optimisation based on a non-Markov decision process to evaluate future expected rewards (Hu and Lin, 2019). An intelligent agent automates the market trade of a given commodity or currency with the objectives of maximising returns and minimising losses for the trader (Usha et al., 2019). The model learns from trends in historical market data using DRL and can buy, sell, or hold a trade at a given instance. A parallel multi-module DRL algorithm is composed of a fully connected layer of neurons to learn the current state from the market data of the traded stock and fundamental data of the issuing company, whereas the other module applies LSTM layers to detect the long-term historical trend of the market (Ma et al., 2021). A framework for market making with signals based on model-free DRL is based on a state space formulation that incorporates outputs from the standalone signal generating units and an action space and reward function formulation (Gasperov and Kostanjcar, 2021). A multivariate time series prediction method of high-dimensional data based on DRL improves the prediction accuracy of high-dimensional data time series (Ji et al., 2021). The DRL method solves the time delay of each variable while mining the data characteristics. Adversarial learning is used during the RL process to enhance a portfolio management model (Huang et al., 2021). A multi-agent framework is adopted where different agents are designed to deal with different portfolio management objectives.

2.3. Deep Reinforcement Learning applications

In addition to finance, DRL has also been applied to other fields such as resource management tasks for systems and networking (Mao et al., 2016). Decision-making activities and appropriate solutions are taken depending on the understanding of the workload and environment experience.

Imagination-Augmented Agents (I2As) are a novel architecture for DRL that combines model-free and model-based aspects (Racanière et al., 2017). A model-based RL and planning methods prescribe how a model should be used to arrive at a policy. I2As learn to interpret predictions from a learned environment model to construct implicit plans in arbitrary ways by using the predictions as an additional context in deep policy networks. Complex goals shall be communicated to these systems for sophisticated DRL systems to interact usefully with real-world environments (Christiano et al., 2017). However, goals can also be defined as non-expert human preferences between pairs of trajectory segments to solve complex RL tasks without access to the reward function.

A framework for autonomous driving based on DRL is a challenging model as a supervised learning problem due to its complex interactions with the environment, including other vehicles, pedestrians and roadworks. Recurrent Neural Networks based on DRL integrate information to enable the handling of partially observable scenarios between vehicles (Sallab et al., 2017). The solution also integrates recent work on attention models that focus on relevant information to reduce the computational complexity associated with the deployment of embedded hardware. An active detection and class-specific model that locates objects in scenes enables agents to focus attention on candidate regions when identifying the correct location of a target object (Caicedo and Lazebnik, 2015). The agent learns to deform a bounding box via simple transformation actions to determine the most specific location of target objects that follows top-down reasoning and DRL.

DRL enables autonomous robots to learn extensive collections of behavioural skills with minimal human intervention. However, robotic applications of RL often compromise the autonomy of the learning process in favour of achieving training times that are practical for real physical systems (Gu et al., 2017). A DRL algorithm based on off-policy training of deep Q-functions scales to complex 3D manipulation tasks to learn deep neural network policies efficiently enough to train on real physical robots. Typically, DRL methods only utilise visual inputs for training, although an innovative method augments these models to exploit 3D feature information that involves partially observable states (Lample and Chaplot, 2017). The model is trained to simultaneously learn these features to minimise a Q-learning objective that improves the training speed and performance of the agent.

A DRL process based on robotic training assists mental health patients (Altameem et al., 2020). The learning approach uses the state and action process to determine every patient's needs and the respective assistance. A DRL speed control strategy is applied to a permanent magnet synchronous motor servo system with parameter uncertainties and load variations (Song et al., 2021). The speed control problem is formulated as a Markov decision process problem, and it is computed as an optimal regulation scheme where each speed and error state is associated with a deep Q-network. A multi-task learning system simultaneously optimises two objectives: domain classification and abstractive review summarisation, in which a document modelling module is shared with the domain classifier via a DRL algorithm (Yang et al., 2020). A generic and adaptable navigation algorithm is based on a proximal policy optimisation via a DRL algorithm that learns navigation with minimum information coupled with LSTM neural networks that provides navigation memory to overcome obstacles (Hodge et al., 2021). A DRL method for continuous fine-grained drone control acquires high-quality frontal view person shots coupled with a reward-shaping approach to improve the stability of the employed continuous RL method (Passalis and Tefas, 2020). A DRL based algorithm partitions chain requests of service function across different infrastructure providers while considering service reliability and quality of service (Kibalya et al., 2020). The DRL algorithm provides the intelligence to infer undisclosed information from infrastructure providers from historical data obtained from previous experiences.

3. The Deep Reinforcement Learning Random Neural Network

3.1. Reinforcement Learning model

In a Reinforcement Learning model, agents interact with the environment via observations, O, and actions, A. At each interaction step t, the agent receives as input some indication of the current state s_t of the environment; then, the agent selects an action a_t with a probability p_t to generate as output. This action changes the environment state s_t to $s_{t+1,}$ and the value of this state evolution or transition is transmitted to the agent through a scalar reinforcement signal or reward r_{t+1} (s_t, a_t, s_{t+1}). The agent's behaviour chose actions over time that tend to increase the long term sum of values of the reinforcement signal by try and error guided by a reinforcement algorithm. Basic reinforcement learning is modelled as a Markov decision process:

- a set of environment and agent states, S
- a set of actions, A, of the agent
- a set of Observations O, from the agent
- a set of temporal measurements t
- $P_a(s, s_t)$ as the probability of transition from state s_t to state s_{t+1} under action a
- $R_a(s, s_t)$ as the immediate reward r_{t+1} after a transition from state s_t to state s_{t+1} under action a_t
- rules that describe agent observations

3.2. Deep Reinforcement learning model

The Deep Reinforcement Learning algorithm presented in this section consists of the Random Neural Network (RNN) (Gelenbe, 1989, 1993a,b). Reinforcement Learning with the RNN has been introduced in earlier work such as (Gelenbe, 2004; Gelenbe et al., 1999, 2000; Gelenbe, 2009; Gelenbe et al., 2004, 2001a,b; Gelenbe and Lent, 2004; Serrano and Gelenbe, 2020; Serrano, 2018). The number of neurons of the neural network matches the amount of decisions to be made; neurons are designated as 1, …, j, …, n; therefore for any taken decision i, there is some associated neuron i. Decisions in the DRL algorithm are selected by taking the option j that corresponds to the most excited neuron, j, which is one that has the greatest value of q_j. The state q_j is defined as the probability that neuron j is excited; these values meet the following system of non-linear equations:

$$\begin{aligned} q_j &= \frac{\lambda^+(j)}{r(j)+\lambda^-(j)} \\ \lambda^+(j) &= \sum_{i=1}^{n}\left[q_i r(i) p^+(i,j)\right] + \Lambda(j) \\ \lambda^-(j) &= \sum_{i=1}^{n}\left[q_i r(i) p^-(i,j)\right] + \lambda(j) \end{aligned} \tag{1}$$

Decision Neurons make choices about the predicted or forecasted trend based on the measured Reward on the DRL model. In particular, the agent considers if the trend of the next Reward is equal, downwards or upwards (Fig. 1). As well as trend prediction, the presented

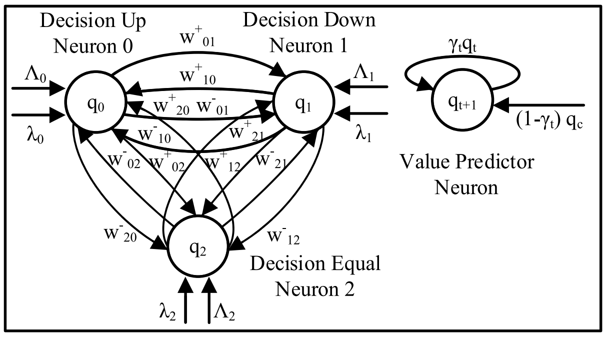

Fig. 1. Deep Reinforcement Learning model.

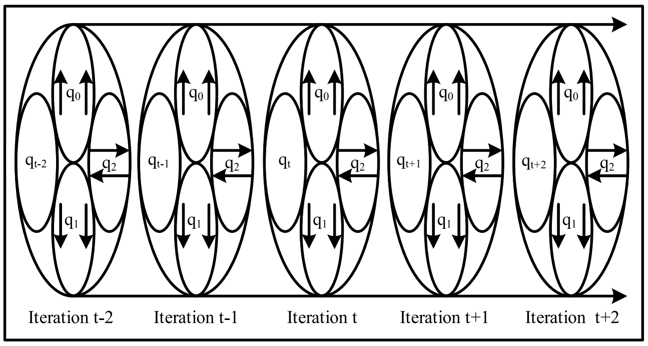

Fig. 2. Deep Reinforcement Algorithm.

model consists of a Predictor Neuron that forecasts the figure of future rewards.

The DRL algorithm modifies the reward and threshold based on a similar approach defined in the Cognitive Packet Network (Gelenbe, 2004; Gelenbe et al., 1999, 2000; Gelenbe, 2009; Gelenbe et al., 2004, 2001a,b; Gelenbe and Lent, 2004; Serrano and Gelenbe, 2020; Serrano, 2018). The agent is given a Goal G that has to achieve as a function to be optimised, where Reward R is a consequence of the interaction with the environment. R_l, l = 1, 2, … represents sequential values of the measured R, these values are used to calculate a decision threshold T_l according to the following equations:

$$R_l = \beta\left(Y_l - Y_{l-1}\right)$$
$$T_l = \alpha T_{l-1} + (1-\alpha)R_l \tag{2}$$

where β defines the Learning Gradient and α defines the Threshold Memory, $0 < \alpha < 1$. These parameters could be dynamically revised following the measurements from external observations or statically assigned. DRL takes the l_{th} decision based on the most excited neuron, lets say neuron j. After the interaction with the environment, the l_{th} Reward R_l is measured and its associated T_{l-1} is computed where neural network weights are revised according to the following equations for all neurons i ≠ j. The DRL algorithm rewards the neural network weights if the trend decision made by the agent is right; R_l>0 and j=0 (upwards) or R_l<0 and j=1 (downwards) or R_l = 0 and j=2 (equal):

$$w^+_l(i,j) = \sum_{t=0}^{l-1} \delta_t w^+_t(i,j) + T_l$$
$$w^-_l(i,k) = \sum_{t=0}^{l-1} \delta_t w^-_t(i,k) + T_l if k \neq j$$

Otherwise, it punishes the neural network weights by:

$$w^+_l(i,k) = \frac{1}{l-1}\sum_{t=0}^{l-1} \delta_t w^+_t(i,k) + T_l if k \neq j$$
$$w^-_l(i,j) = \frac{1}{l-1}\sum_{t=0}^{l-1} \delta_t w^-_t(i,j) + T_l \tag{3}$$

where δ_t defines a weighting factor that is variable and depends on t, $0 < \delta_t < 1$, and l the stage decision. On the above equations, w^+_{ij} represents the rate at which neuron i emits excitation spikes to neuron j and w^-_{ij} represents the rate at which neuron i releases inhibitory spikes to neuron j, neuron i must be excited to transmit excitation on inhibition spikes. The fundamental property of the DRL algorithm is that learning considers time, therefore memory, when it updates the neural network weights; values consist of the entire previous measurements rather than the only is previous neural state (Fig. 2).

In addition to the DRL algorithm for trend decisions, the agent also uses DRL to make predictions on the quantitative value of future rewards. The value Predictor Neuron is based on the current measurement q_c and the entire previous predictions q_t:

$$q_{t+1} = \frac{1}{l-1}\sum_{t=0}^{l-1} \gamma_t q_t + (1-\gamma_t) q_c \tag{4}$$

where γ_t represents the prediction memory, a weighting factor that is variable and depends on t, $0 < \gamma_t < 1$. This parameter can be dynamically updated based on the measurements from external observations or statically assigned.

4. Experimental results

The Deep Reinforcement Learning algorithm has been validated with simple linear functions and market datasets that cover Bitcoin, House, FTSE Stock, Gold and Brent oil prices. Although market behaviour and trends depend on external inputs from the market itself, the validation only includes internal market values. External market inputs such as political decisions, influencer comments, and natural events can be included into DRL the model as a weighted factor in the reward R_l. These external inputs can be extracted from news, and their sentiment quantified via Natural Language Processing (NLP). The finite choices of the DRL algorithm are based on the trend of the values of a function: Upward, Downwards or Equal. The DRL reward $R_l = \beta(Y_l - Y_{l-1})$ is calculated as the difference between two consecutive values of the function. This is also included in the article. The validation is based on four different memory configurations, including sampling rate and memory duration, as shown in Table 1.

Table 1
Deep Reinforcement Learning Configurations.

Type	Time	Description
MP1-0M	t = l − 1	Learning only applies to the previous day
MP2-FM	t = 0	Learning applies entirely from day 1
MP3-ΔR	t = l − 1 − Δ	Learning applies only the last Δ period
MP4-ΔS	t = Δ	Learning applies only at every Δ step

Trend neurons are validated based on the accuracy (%) of their combined upward, downward and equal predictions:

$$\text{Accuracy} = \frac{\text{Correct predictions}}{\text{Total predictions}} \cdot 100 \tag{5}$$

The predictor neuron is validated based on the error against the real measurements using three metrics:

The Root Mean Square Error (RMSE):

$$\text{RMSE} = \sqrt{\sum_{i=1}^{N} \frac{\left(y_i - x_i\right)^2}{N}} \tag{6}$$

Mean Absolute Error (MAE):

$$\text{MAE} = \frac{\sum_{i=1}^{N} \left|y_i - x_i\right|}{N} \tag{7}$$

Mean Absolute Percentage Error (MAPE):

$$\text{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i - x_i}{y_i}\right| \tag{8}$$

where x_i are the predicted values, y_i are the measured values and N the total number of measurements.

The proposed DRL algorithm is compared against the simple Reinforcement Learning and the linear prediction based on the two previous values:

$$\begin{aligned} \text{Upward: if } y_i - y_{i-1} > 0 \\ \text{Downward: if } y_i - y_{i-1} < 0 \\ \text{Equal: if } y_i - y_{i-1} = 0 \end{aligned} \tag{9}$$

4.1. Linear function validation

Four linear functions have been selected to validate the proposed DRL algorithm. The following tables show the number of upwards, downwards and equal predictions, Rewards (R) or success, Penalisations (P) or misses, and Accuracy (A) for the trend neuron. Experiments cover the 100 data measurements for the MP1-0M, MP2-FM, MP3-ΔR and MP4-ΔS configuration respectively.

4.1.1. Function 1 validation

Function 1 presents a gradual change between the three decisions, upwards, downwards and equal (Fig. 3).

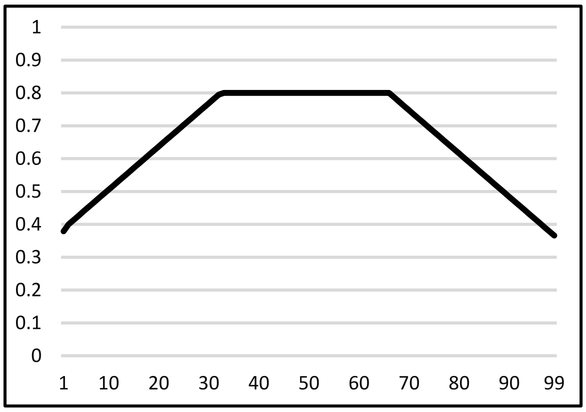

Fig. 3. Linear Validation — Function 1.

Table 2 shows the values for medium Threshold Memory ($\alpha = 0.5$) and large Learning Gradient ($\beta = 1E5$) and $\Delta = 5$.

The addition of memory to the Reinforcement Learning algorithm does not improve its decisions. Furthermore, predictions based on the previous two values provide more accuracy. Figs. 4 and 5 show that the Learning Gradient β has a more significant effect than the Threshold Memory α.

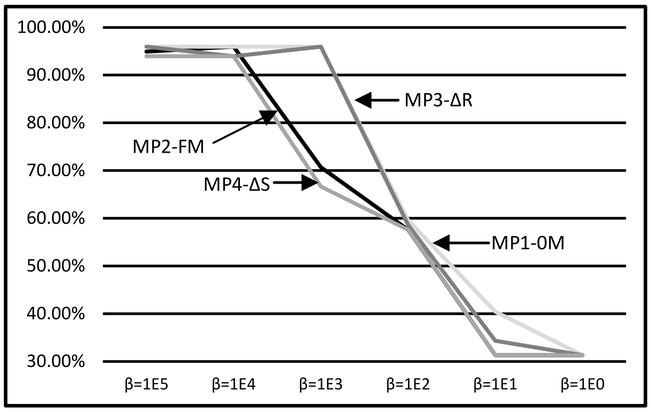

Fig. 4. Function 1 Validation — β value.

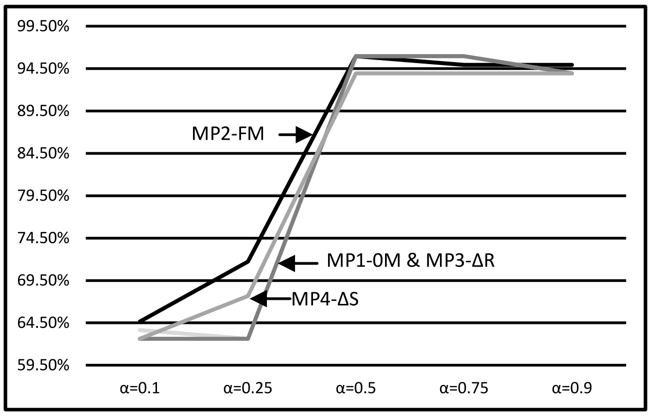

Fig. 5. Function 1 Validation — α value.

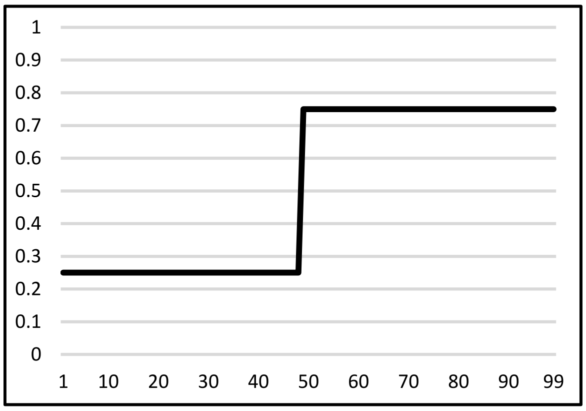

Fig. 6. Linear Validation — Function 2.

Table 2
Function 1-Trend Neuron.

Parameter	Deep Reinforcement Learning				Previous two values
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS	
Upward Prediction	32	32	32	33	32
Downward Prediction	33	33	33	33	33
Equal Prediction	34	34	34	33	33
Reward	95	95	95	93	95
Penalisation	4	4	4	6	3
Accuracy	95.96%	95.96%	95.96%	93.94%	96.94%

Table 3
Function 1 validation — Predictor Neuron.

Error	Deep Reinforcement Learning			
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS
RSME	4.03E−03	1.19E−01	2.12E−02	3.06E−02
MAE	1.38E−03	9.65E−02	1.72E−02	2.20E−02
MAPE	2.83E−01	1.69E+01	3.06E+00	3.82E+00

Table 3 shows the error of the predictor neuron. Additional memory in the DRL (MP2-FM) does not enhance the prediction with the best results without memory (MP1-0M).

4.1.2. Function 2 validation

Function 2 represents the same equal trend at two different values (Fig. 6).

Table 4 shows the values for medium Threshold Memory ($\alpha = 0.5$) and large Learning Gradient ($\beta = 1E4$) and $\Delta = 5$.

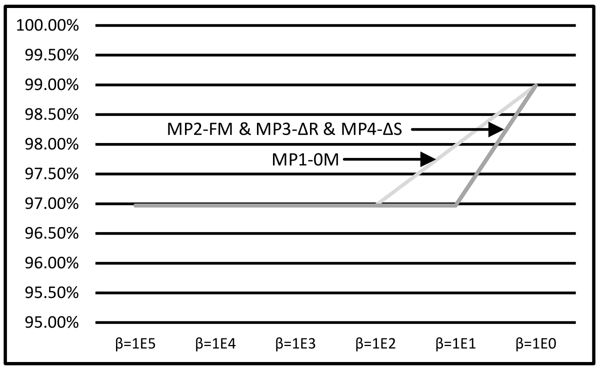

Fig. 7. Function 2 Validation — β value.

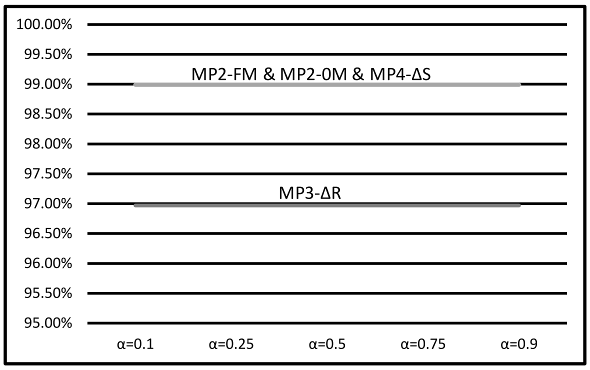

Fig. 8. Function 2 Validation — α value.

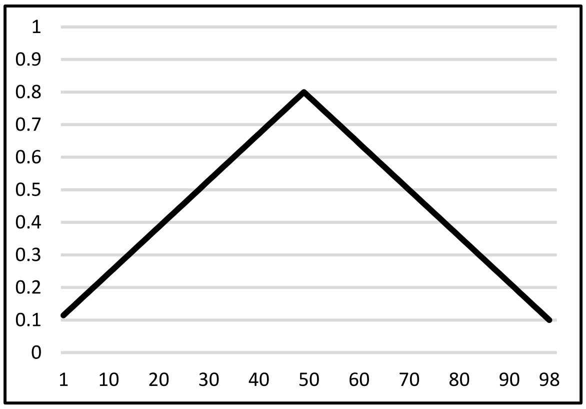

Fig. 9. Linear Validation — Function 3.

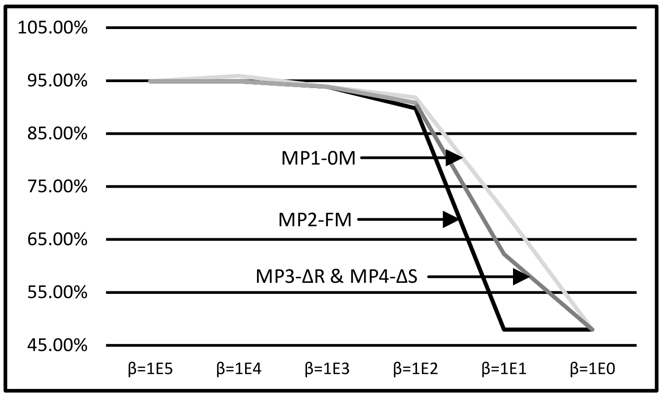

Fig. 10. Function 3 Validation — β value.

Table 4
Function 2-Trend Neuron.

Parameter	Deep Reinforcement Learning				Previous two values
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS	
Upward Prediction	0	0	1	0	1
Downward Prediction	0	0	1	0	0
Equal Prediction	99	99	97	99	97
Reward	98	98	96	98	95
Penalisation	1	1	3	1	3
Accuracy	98.99%	98.99%	96.97%	98.99%	96.94%

Table 5
Function 2 validation — Predictor Neuron.

Error	Deep Reinforcement Learning			
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS
RSME	5.65E−03	1.88E−01	5.62E−02	1.84E−02
MAE	8.42E−04	1.24E−01	1.07E−02	1.37E−02
MAPE	1.87E−01	1.67E+01	1.51E+00	3.91E+00

Table 6
Function 3-Trend Neuron.

Parameter	Deep Reinforcement Learning				Previous two values
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS	
Upward Prediction	49	49	49	49	48
Downward Prediction	48	47	47	47	49
Equal Prediction	1	2	2	2	0
Reward	94	93	93	93	95
Penalisation	4	5	5	5	2
Accuracy	95.92%	94.90%	94.90%	94.90%	97.94%

Table 7
Function 3 validation — Predictor Neuron.

Error	Deep Reinforcement Learning			
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS
RSME	1.96E−03	1.69E−01	2.81E−02	1.59E−02
MAE	1.68E−03	1.42E−01	2.75E−02	1.30E−02
MAPE	5.68E−01	4.62E+01	7.94E+00	3.84E+00

The four different DRL algorithms present similar results, similar to the values obtained from the predictions based on the previous two values. Figs. 7 and 8 show that the Learning Gradient β and the Threshold Memory α do not largely affect the prediction. This is due there is only one trend in the function.

Table 5 shows the error of the predictor neuron. The more memory included in the algorithm, the more error in the prediction.

4.1.3. *Function 3 validation*

Function 3 represents a gradual change between two decisions: upwards and downwards (Fig. 9).

Table 6 shows the values for medium Threshold Memory (α = 0.5) and large Learning Gradient (β = 1E5) and Δ = 5.

The addition of memory to the Reinforcement Learning algorithm does not improve its trend decision. Instead, predictions based on the previous two values provide better accuracy. Figs. 10 and 11 show that a large Learning Gradient β provides the best results; however, there is not a large effect on the variations of the Threshold Memory α.

Table 7 shows the error of the predictor neuron. Additional memory in the DRL (MP2-FM) does not enhance the prediction with the best results without memory (MP1-0M).

4.1.4. *Function 4 validation*

Function 4 represents the same upwards trend (Fig. 12).

Table 8 shows the values for medium Threshold Memory (α = 0.5) and large Learning Gradient (β = 1E5) and Δ = 5.

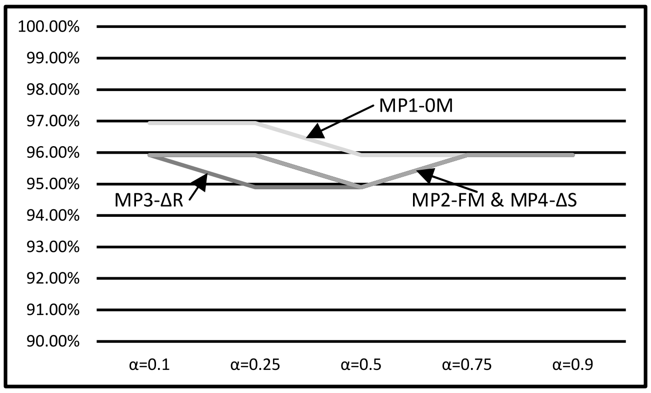

Fig. 11. Function 3 Validation — α value.

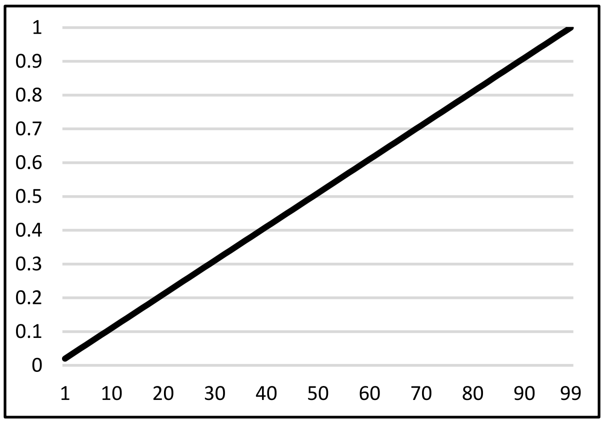

Fig. 12. Linear Validation — Function 4.

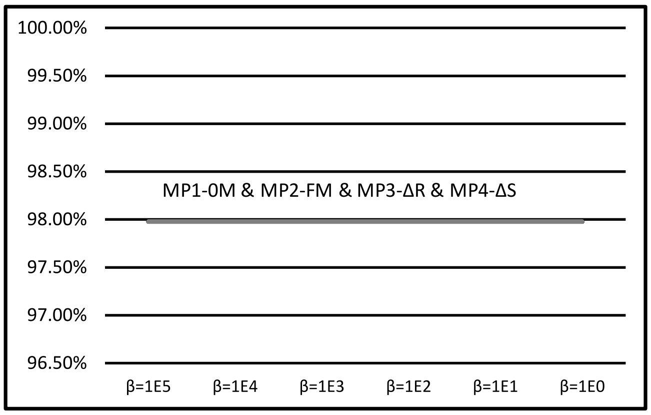

Fig. 13. Function 4 Validation — β value.

The four DRL algorithms and the decision based on the previous two values perform similarly. Figs. 13 and 14 show that there are no effects in the variation of Learning Gradient β and Threshold Memory α. This is due there is only one trend in the function.

Table 9 shows the error of the predictor neuron. The more memory included in the algorithm, the more error in the prediction.

4.2. Market validation

4.2.1. Bitcoin market validation

DRL has been validated with the price of Bitcoin from 18th July 2010 to 03rd of June 2021 with 3974 data points in total (Table 10).

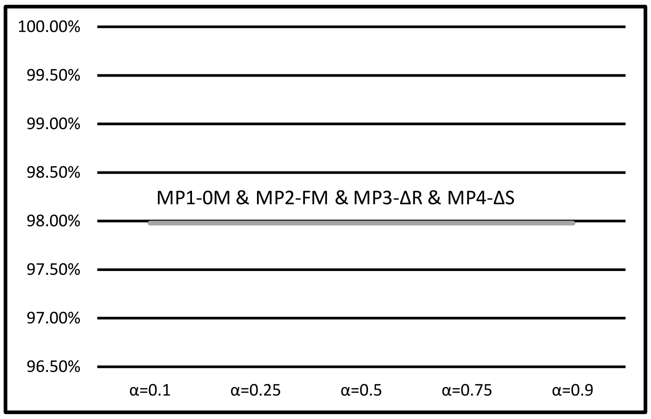

Fig. 14. Function 4 Validation — α value.

Table 8
Function 4-Trend Neuron.

Parameter	Deep Reinforcement Learning				Previous two values
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS	
Upward Prediction	97	97	97	97	98
Downward Prediction	1	1	1	1	0
Equal Prediction	1	1	1	1	0
Reward	97	97	97	97	97
Penalisation	2	2	2	2	1
Accuracy	97.98%	97.98%	97.98%	97.98%	98.98%

Table 9
Function 4 validation — Predictor Neuron.

Error	Deep Reinforcement Learning			
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS
RSME	1.12E−03	1.90E−01	2.00E−02	1.10E−02
MAE	1.12E−03	1.64E−01	1.98E−02	1.10E−02
MAPE	5.18E−01	3.09E+01	6.32E+00	4.21E+00

Table 10
Bitcoin market dataset.

Variable	Bitcoin
Period	18 July 2010 to 03 Jun 2021
Data Points	3974
Max	63,540.90
Min	0.10
Average	4,751.28
σ	9,824.73
Period	Daily

Data has been obtained from www.uk.investing.com and is publicly available (Fig. 15).

Table 11 shows the number of upwards, downwards and equal predictions, Rewards (R) or success, Penalisations (P) or misses, and Accuracy (A) for the trend neuron. Experiments cover the 3974 data measurements for the MP1-0M, MP2-FM, MP3-ΔR and MP4-ΔS configuration respectively at medium Threshold Memory ($\alpha = 0.5$), medium Learning Gradient ($\beta = 1$) and $\Delta = 5$ (weekly).

The MP2-FM DRL algorithm provides slightly better results than the other methods, although the best accuracy is when the prediction is based on the previous two values. This result is because the stock market prices tend to go either up or down, where the equal decision is low. There is a correlation between MP1-0M and MP3-ΔR, likewise between MP2-FM and MP4-ΔS (Fig. 16). In addition, the results confirm that the Learning Gradient β factor has a more significant impact than the Threshold Memory α on the final Accuracy values (Fig. 16 and Fig. 17). Optimum results are obtained with intermediate values. The

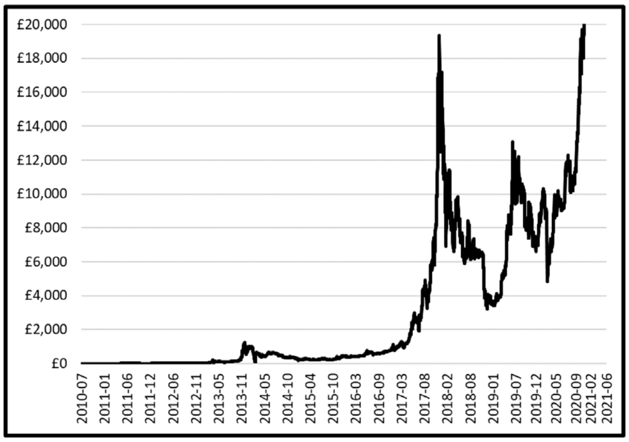

Fig. 15. Bitcoin price validation.

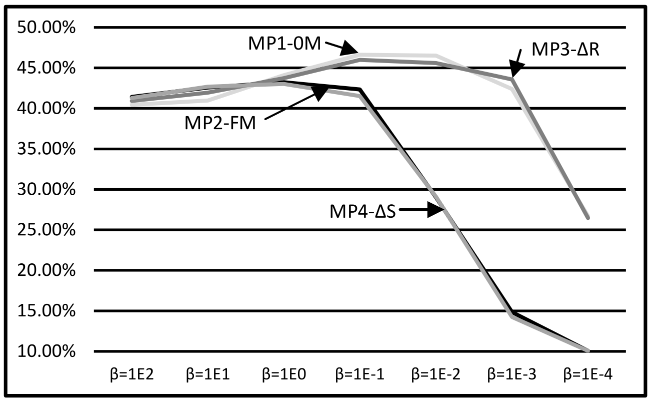

Fig. 16. Bitcoin price Validation — β value.

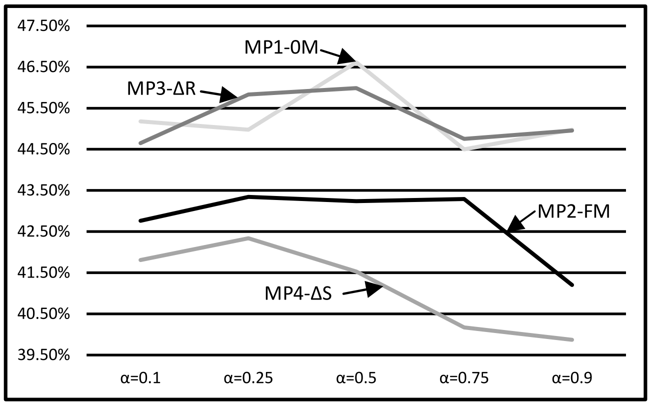

Fig. 17. Bitcoin price Validation — α value.

linear prediction based on the two previous values provides the best results.

Table 12 shows the error of the predictor neuron. Additional memory in the DRL (MP2-FM) does not enhance the prediction with the best results without memory (MP1-0M).

4.2.2. House market validation

DRL has been validated with the price of UK houses from April 1968 to March 2021 with 636 data points in total (Table 13). Data has been obtained from landregistry.data.gov.uk and is publicly available (Fig. 18).

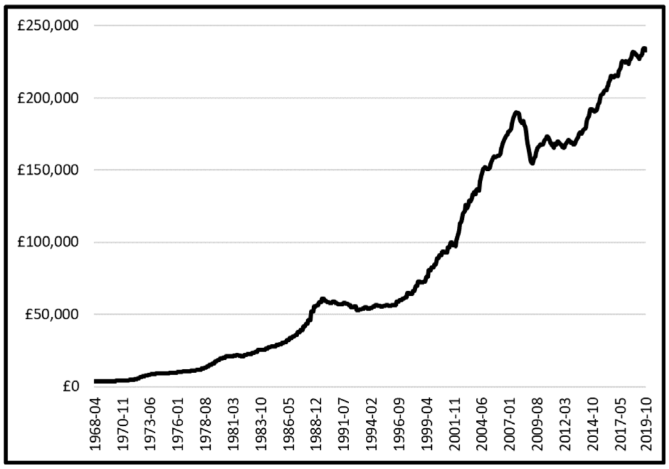

Fig. 18. House price validation.

Table 11
Bitcoin Price validation-Trend Neuron.

Parameter	Deep Reinforcement Learning				Previous two values
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS	
Upward Prediction	1515	1719	1728	1485	1930
Downward Prediction	1733	1361	1495	1752	1643
Equal Prediction	725	893	750	736	399
Reward	1718	1752	1737	1709	1901
Penalisation	2255	2221	2236	2264	2071
Accuracy	43.24%	44.10%	43.72%	43.02%	47.86%

Table 12
Bitcoin validation — Predictor Neuron.

Error	Deep Reinforcement Learning			
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS
RSME	48.37	9093.11	547.54	48.63
MAE	14.65	3504.85	177.28	14.84
MAPE	0.33	52.44	3.93	0.46

Table 13
House Price dataset.

Variable	House Price
Period	April 1968 to March 2021
Data Points	636
Max	256,405.00
Min	3,595.00
Average	88,847.64
σ	76,001.03
Period	Monthly

Table 14 shows the number of upwards, downwards, and equal predictions, Rewards (R) or success, Penalisations (P) or misses, and Accuracy (A) for the trend neuron. Experiments cover the 636 data measurements for the MP1-0M, MP2-FM, MP3-ΔR and MP4-ΔS configuration respectively at medium Threshold Memory (α = 0.5), low Learning Gradient (β = 1.00E-3) and Δ = 12 (yearly).

The MP3- ΔR DRL algorithm provides the best results. The results also confirm that the Learning Gradient β factor has a greater impact than the Threshold Memory α on the final Accuracy values (Figs. 19 and 20). Thus, DRL performs better than linear prediction.

Table 15 shows the error of the predictor neuron. Again, the additional memory of the DRL does not enhance the prediction.

4.2.3. FTSE market validation

DRL has been validated with the price of the FTSE stock prices from 03 Jan 2001 to 15 Oct 2020 with 4999 data points in total (Table 16).

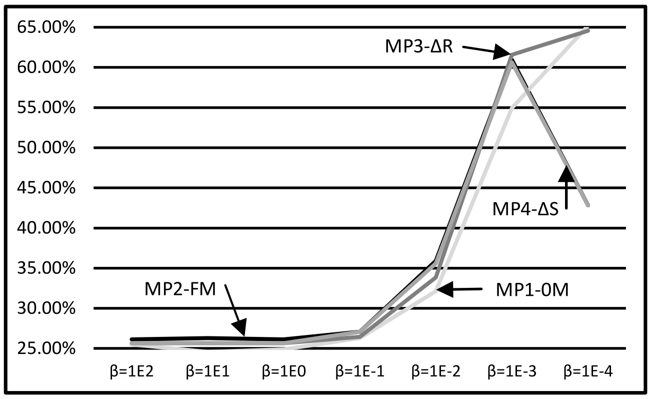

Fig. 19. House price Validation — β value.

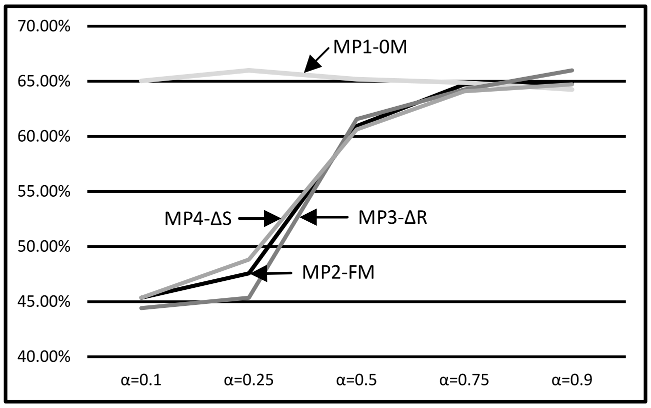

Fig. 20. House price Validation — α value.

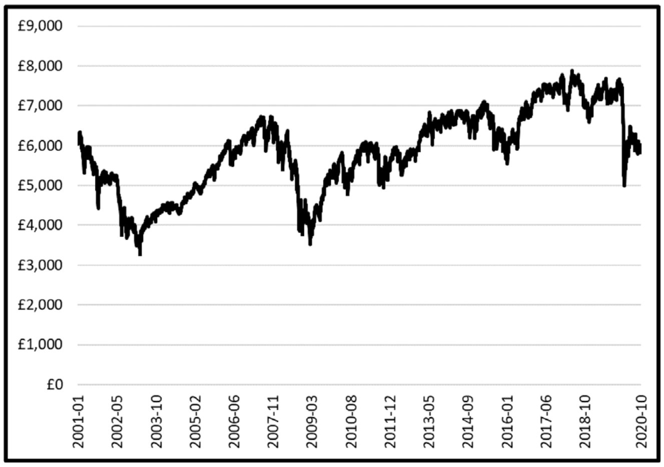

Fig. 21. FTSE price validation.

Data has been obtained from www.uk.investing.com and is publicly available (Fig. 21).

Table 17 shows the number of upwards, downwards, and equal predictions, Rewards (R) or success, Penalisations (P) or misses, and Accuracy (A) for the trend neuron. Experiments cover the 4999 data measurements for the MP1-0M, MP2-FM, MP3-ΔR and MP4-ΔS configuration respectively at medium Threshold Memory (α = 0.5), low Learning Gradient (β = 1.00E-3) and Δ = 5 (weekly).

The MP4-ΔS DRL algorithm provides similar results as the Reinforcement Learning without memory (MP1-0M). The results also confirm that the Learning Gradient β factor has a greater impact than the

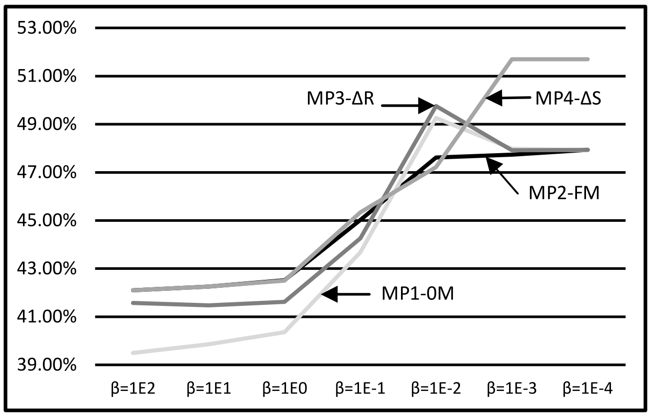

Fig. 22. FTSE price Validation — β value.

Table 14
House Price validation-Trend Neuron.

Parameter	Deep Reinforcement Learning				Previous two values
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS	
Upward Prediction	217	270	243	216	265
Downward Prediction	43	69	40	41	98
Equal Prediction	375	296	352	378	271
Reward	387	349	391	385	299
Penalisation	248	286	244	250	335
Accuracy	60.94%	54.96%	61.57%	60.63%	47.16%

Table 15
House validation — Predictor Neuron.

Error	Deep Reinforcement Learning			
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS
RSME	132.08	57087.24	4272.29	331.80
MAE	77.23	42048.05	2890.00	281.17
MAPE	0.12	40.05	4.06	0.96

Table 16
FTSE dataset.

Variable	FTSE
Period	03 Jan 2001 to 15 Oct 2020
Data Points	4999
Max	7,877.45
Min	3,287.00
Average	5,873.20
σ	1,026.30
Period	Daily

Table 17
FTSE Price validation-Trend Neuron.

Parameter	Deep Reinforcement Learning				Previous two values
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS	
Upward Prediction	2656	2533	2735	2670	2588
Downward Prediction	1946	1911	1787	1938	2403
Equal Prediction	396	554	476	390	5
Reward	2250	2183	2212	2266	2461
Penalisation	2748	2815	2786	2732	2535
Accuracy	45.02%	43.68%	44.26%	45.34%	49.26%

Threshold Memory α on the final Accuracy values (Figs. 22 and 23). Median values provide the best accuracy.

Table 18 shows the error of the predictor neuron. The additional memory DRL does not enhance the prediction of the value.

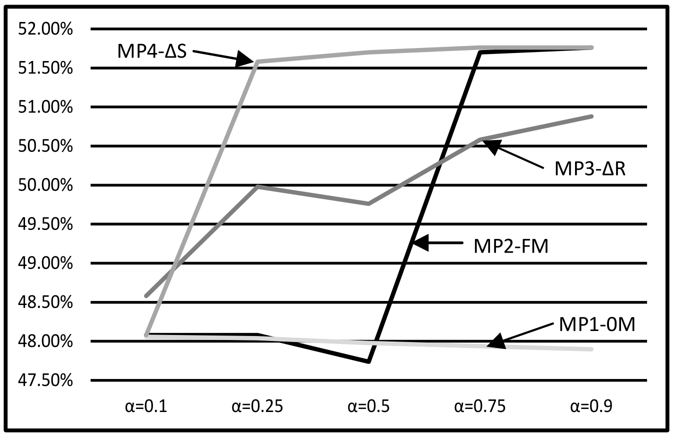

Fig. 23. FTSE price Validation — α value.

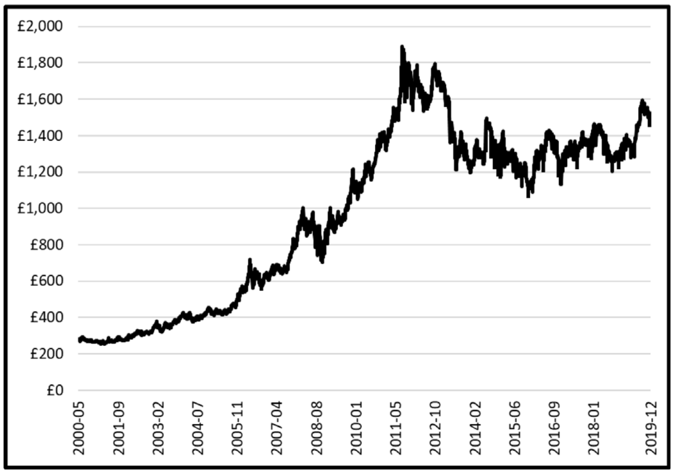

Fig. 24. Gold price validation.

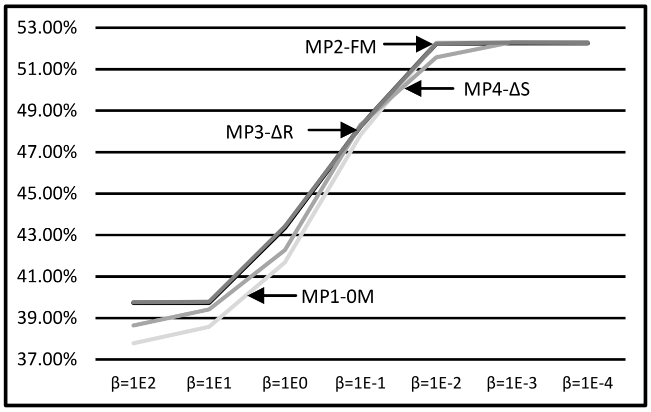

Fig. 25. Gold price Validation — β value.

4.2.4. Gold market validation

DRL has been validated with the price of the Gold market from 06 May 2002 to 04 Jun 2021 with 4895 data points in total (Table 19). Data has been obtained from www.uk.investing.com and is publicly available (Fig. 24).

Table 20 shows the number of upwards, downwards, and equal predictions, Rewards (R) or success, Penalisations (P) or misses, and Accuracy (A) for the trend neuron. Experiments cover the 4895 data measurements for the MP1-0M, MP2-FM, MP3-ΔR and MP4-ΔS configuration respectively at medium Threshold Memory (α = 0.5), low Learning Gradient (β = 1.00E-3) and Δ = 5 (weekly).

The MP2-FM DRL algorithm provides similar results as the Reinforcement Learning without memory (MP1-0M). The results also confirm that the Learning Gradient β factor has a greater impact than the Threshold Memory α on the final Accuracy values (Figs. 25 and 26). Intermediate values provide the best accuracy.

Table 21 shows the error of the predictor neuron. The additional DRL does not enhance the prediction.

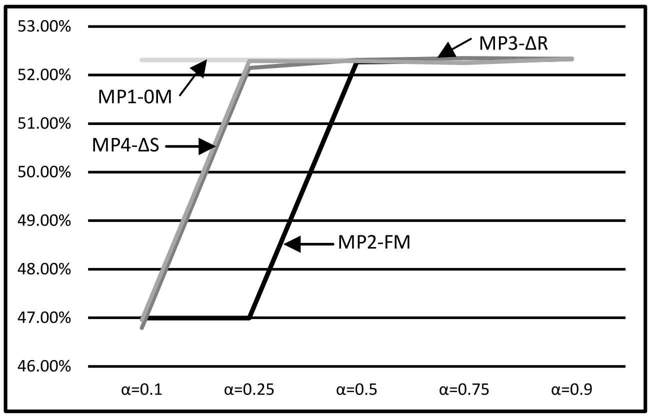

Fig. 26. Gold price Validation — α value.

Table 18
FTSE validation — Predictor Neuron.

Error	Deep Reinforcement Learning			
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS
RSME	10.90	833.17	70.90	55.09
MAE	4.69	702.89	50.73	11.14
MAPE	0.08	11.74	0.90	0.20

Table 19
Gold dataset.

Variable	Gold
Period	06 May 2002 to 04 Jun 2021
Data Points	4895
Max	2,103.20
Min	302.50
Average	1,109.58
σ	465.07
Period	Daily

Table 20
Gold Price validation-Trend Neuron.

Parameter	Deep Reinforcement Learning				Previous two values
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS	
Upward Prediction	4878	3279	3094	4878	2565
Downward Prediction	13	1597	1786	13	2299
Equal Prediction	3	18	14	3	29
Reward	2557	2559	2524	2557	2289
Penalisation	2337	2335	2370	2337	2604
Accuracy	52.25%	52.29%	51.57%	52.25%	46.78%

Table 21
Gold validation — Predictor Neuron.

Error	Deep Reinforcement Learning			
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS
RSME	1.80	314.14	17.30	3.35
MAE	1.00	234.87	11.31	1.46
MAPE	0.09	18.93	1.02	0.21

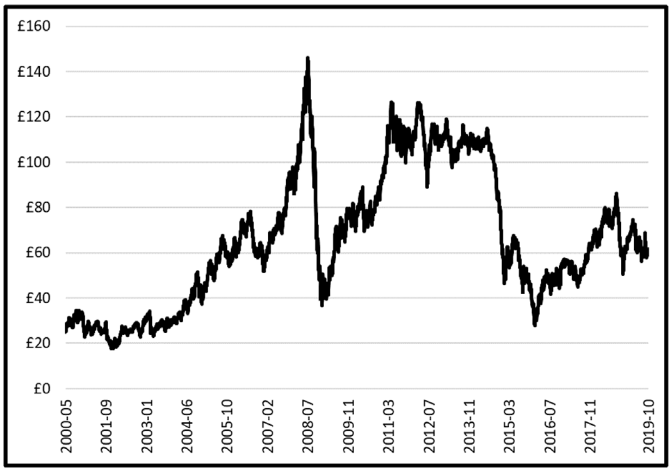

Fig. 27. Brent oil price validation.

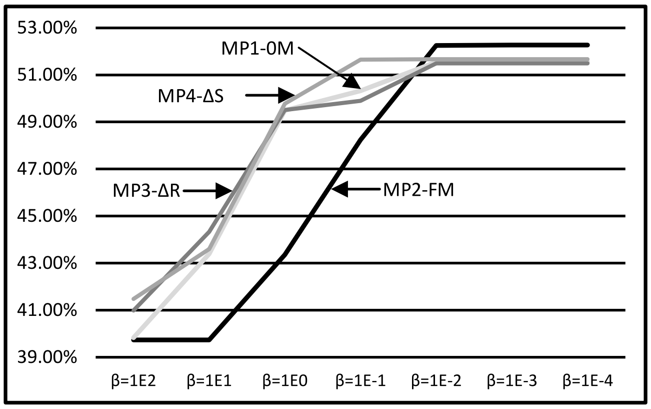

Fig. 28. Brent oil price Validation — β value.

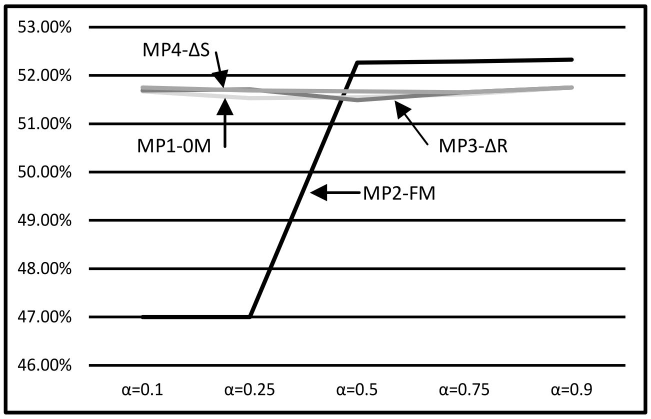

Fig. 29. Brent oil price Validation — α value.

4.2.5. *Brent oil market validation*

DRL has been validated with the price of the Brent oil market from 03 May 2000 to 24 Oct 2019 with 4907 data points in total (Table 22). Data has been obtained from www.uk.investing.com and is publicly available (Fig. 27).

Table 23 shows the number of upwards, downwards, and equal predictions, Rewards (R) or success, Penalisations (P) or misses, and Accuracy (A) for the trend neuron. Experiments cover the 4907 data measurements for the MP1-0M, MP2-FM, MP3-ΔR and MP4-ΔS configuration respectively at medium Threshold Memory (α = 0.5), low Learning Gradient (β = 1.00E-3) and Δ = 5 (weekly).

Table 22
Brent oil dataset.

Variable	Brent oil
Period	03 May 2000 to 24 Oct 2019
Data Points	4907
Max	146.08
Min	19.33
Average	68.76
σ	27.78
Period	Daily

Table 23
Brent oil price validation-Trend Neuron.

Parameter	Deep Reinforcement Learning				Previous two values
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS	
Upward Prediction	4878	4876	4875	4899	2539
Downward Prediction	13	27	26	4899	2351
Equal Prediction	3	3	5	2	15
Reward	2557	2529	2526	2535	2361
Penalisation	2337	2377	2380	2371	2544
Accuracy	52.25%	51.55%	51.49%	51.67%	48.13%

Table 24
Brent oil validation — Predictor Neuron.

Error	Deep Reinforcement Learning			
	MP1-0M	MP2-FM	MP3-ΔR	MP4-ΔS
RSME	0.14	23.90	1.53	0.28
MAE	0.10	19.43	1.11	0.13
MAPE	0.16	30.69	1.78	0.26

The DRL without memory (MP1-0M) outperforms the other DRL algorithms. The results also confirm that the Learning Gradient β factor has a greater impact than the Threshold Memory α on the final Accuracy values (Figs. 28 and 29). Middle values provide the best accuracy.

Table 24 shows the error of the predictor neuron. Similar to the previous validation, the additional DRL does not enhance the prediction.

5. Conclusions

This article has proposed a Deep Reinforcement Learning algorithm that expands the Random Neural Network Reinforcement Learning algorithm to include the previous learnings entirely from previous rewards, rather than only the actual one. The Random Neural Network weighs are updated with the current reward and the previous values to incorporate time and memory. The presented algorithm is embedded into a decision process that predicts trends: upward, downward and equal in addition to values. DRL algorithm has been analysed with different configurations that include sampling rate and memory duration. The validation process has evaluated the method against several functions and markets: Bitcoin, UK house prices, FTSE, Gold and Brent oil. Experimental results show that the addition of Deep Learning to the Reinforcement Learning Algorithm adds a higher computational cost while not increasing its performance; therefore, memory is not the principal factor in making decisions on market prices. The Learning Gradient β factor has a more significant impact than the Threshold Memory α on the final accuracy values where intermediate values present optimum results. The comparison between DRL and linear prediction based on the previous two values presents mixed results.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Altameem, T., Amoon, M., Altameem, A., 2020. A deep reinforcement learning process based on robotic training to assist mental health patients. Neural Comput. Appl. 1–20.

Caicedo, J., Lazebnik, S., 2015. Active object localisation with deep reinforcement learning. In: IEEE International Conference on Computer Vision. pp. 1–9.

Christiano, P., Leike, J., Brown, T., Martic, M., Legg, S., Amodei, D., 2017. Deep reinforcement learning from human references. In: 31st Conference on Neural Information Processing Systems. pp. 1–17.

Dogan, E., 2021. LSTM training set analysis and clustering model development for short-term traffic flow prediction. Neural Comput. Appl. 1–14.

Duan, Y., Chen, X., Houthooft, R., Schulman, J., Abbeel, P., 2016. Benchmarking deep reinforcement learning for continuous control. In: International Conference on Machine Learning. pp. 1329–1338.

Foerster, J., Assael, Y., Freitas, N., Whiteson, S., 2016. Learning to communicate with deep multi-agent reinforcement learning. In: 30th Conference on Neural Information Processing Systems. pp. 2145–2153.

Gasperov, B., Kostanjcar, Z., 2021. Market making with signals through deep reinforcement learning. IEEE Access 9, 61611, 1–12.

Gelenbe, E., 1989. Random neural networks with negative and positive signals and product form solution. Neural Comput. 1, 502–510.

Gelenbe, E., 1993a. Learning in the recurrent random neural network. Neural Comput. 5, 154–164.

Gelenbe, E., 1993b. G-Networks with triggered customer movement. J. Appl. Probab. 30, 742–748.

Gelenbe, E., 2004. Cognitive Packet Network. Patent US 6804201 B1.

Gelenbe, E., 2009. Steps toward self-aware networks. Commun. ACM 52 (7), 66–75.

Gelenbe, E., Lent, R., 2004. Power-aware ad hoc cognitive packet networks. Ad Hoc Netw. 2 (3), 205–216.

Gelenbe, E., Lent, R., Nunez, A., 2004. Self-aware networks and QoS. Proc. IEEE 92 (9), 1478–1489.

Gelenbe, E., Lent, R., Xu, Z., 2000. Networks with cognitive packets. In: IEEE International Symposium on the Modeling, Analysis, and Simulation of Computer and Telecommunication Systems. pp. 3–10.

Gelenbe, E., Lent, R., Xu, Z., 2001a. Design and performance of cognitive packet networks. Perform. Eval. 46 (2–3), 155–176.

Gelenbe, E., Lent, R., Xu, Z., 2001b. Measurement and performance of a cognitive packet network. Comput. Netw. 37 (6), 691–701.

Gelenbe, E., Xu, Z., Seref, E., 1999. Cognitive packet networks. In: International Conference on Tools with Artificial Intelligence. pp. 47–54.

Gu, S., Holly, E., Lillicrap, T., Levine, S., 2017. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In: IEEE International Conference on Robotics and Automation. pp. 1–9.

Haarnoja, T., Tang, H., Abbeel, P., Levine, S., 2017. Reinforcement learning with deep energy-based policies. In: Proceedings of the 34th International Conference on Machine Learning. pp. 1–10.

Hasselt, H., Guez, A., Silver, D., 2016. Deep reinforcement learning with double Q-learning. In: Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence. pp. 2094–2100.

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., Meger, D., 2018. Deep Reinforcement Learning that Matters. Association for the Advancement of Artificial Intelligence, pp. 1–26.

Hessel, M., Modayil, J., Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., Silver, D., 2018. Rainbow: Combining improvements in deep reinforcement learning. In: AAAI Conference on Artificial Intelligence. Association for the Advancement of Artificial Intelligence, pp. 1–14.

Hodge, V., Hawkins, R., Alexander, R., 2021. Deep reinforcement learning for drone navigation using sensor data. Neural Comput. Appl. 33, 2015–2033.

Hu, Y-J., Lin, S.-J., 2019. Deep reinforcement learning for optimizing finance portfolio management. In: Amity International Conference on Artificial Intelligence. pp. 14–20.

Huang, S., Miao, Y., Hsiao, Y., 2021. Novel deep reinforcement algorithm with adaptive sampling strategy for continuous portfolio optimization. IEEE Access 9 (77371), 1–15.

Ji, S., Kim, J., Im, H., 2019. A comparative study of bitcoin price prediction using deep learning. Mathematics 7 (898), 1–20.

Ji, X., Zhang, H., Li, J., Zhao, X., Li, S., Chen, R., 2021. Multivariate time series prediction of high dimensional data based on deep reinforcement learning. E3S Web Conf. 256 (02038), 1–4.

Jin, Z., Yang, Y., Liu, Y., 2020. Stock closing price prediction based on sentiment analysis and LSTM. Neural Comput. Appl. 32, 9713–9729.

Kaelbling, L., Littman, M., Moore, A., 1996. Reinforcement learning: A survey. J. Artificial Intelligence Res. 4, 237–285.

Kibalya, G., Serrat, J., Gorricho, J., Okello, D., Zhang, P., 2020. A deep reinforcement learning-based algorithm for reliability-aware multi-domain service deployment in smart ecosystems. Neural Comput. Appl. 1–20.

Kim, T., King, B., 2020. Time series prediction using deep echo state networks. Neural Comput. Appl. 32, 17769–17787.

Kulkarni, T., Narasimhan, K., Saeedi, A., Tenenbaum, J., 2016. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In: 30th Conference on Neural Information Processing Systems. pp. 1–14.

Kumar, R., Nguyen, T., Cengiz. A. Sharma, K., 2021. Fine-tuned support vector regression model for stock predictions. Neural Comput. Appl. 1–15.

Lample, G., Chaplot, D., 2017. Playing FPS games with deep reinforcement learning. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence. pp. 1–7.

Lillicrap, T., Hunt, J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D., 2016. Continuous control with deep reinforcement learning. In: International Conference on Learning Representations. pp. 1–14.

Livieris, I., Pintelas, E., Pintelas, P., 2020. A CNN–LSTM model for gold price time-series forecasting. Neural Comput. Appl. 32, 17351–17360.

MA, L., Liu, Y., 2019. Application of a deep reinforcement learning method in financial market trading. In: International Conference on Measuring Technology and Mechatronics Automation. pp. 421–425.

Ma, C., Zhang, J., Liu, J., Ji, L., Gao, F., 2021. A parallel multi-module deep reinforcement learning algorithm for stock trading. Neurocomputing 449, 290–302.

Mao, H., Alizadeh, M., Menache, I., Kandula, S., 2016. Resource management with deep reinforcement learning. ACM Workshop on Hot Topics in Networks. pp. 50–56.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A., Veness, J., Bellemare, M., Graves, A., Riedmiller, M., Fidjeland, A., 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. Res. Lett. Nat. 2 (6), 1–13, 518, 529.

Mnih, V., Puigdomenech, A., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., Kavukcuoglu, K., 2016. Asynchronous methods for deep reinforcement learning. In: International Conference on Machine Learning. Vol. 48, pp. 1928–1937.

Nguyen, N., Nguyen, T., Vamplew, P., Dazeley, R., Nahavandi, S., 2021. A prioritised objective actor-critic method for deep reinforcement learning. Neural Comput. Appl. 33, 10335–10349.

Passalis, N., Tefas, A., 2020. Continuous drone control using deep reinforcement learning for frontal view person shooting. Neural Comput. Appl. 32, 4227–4238.

Racanière, S., Weber, T., Reichert, D., Buesing, L., Guez, A., Rezende, D., Badia, A., Vinyals, O., Heess, N., Li, Y., Pascanu, R., Battaglia, P., Hassabis, D., Silver, D., Wierstra, D., 2017. Imagination-augmented agents for deep reinforcement learning. In: 31st Conference on Neural Information Processing Systems. pp. 1–10.

Sallab, A., Abdou, M., Perot, E., Yogamani, S., 2017. Deep reinforcement learning framework for autonomous driving. In: IS & T International Symposium on Electronic Imaging Autonomous Vehicles and Machines. pp. 70–76.

Serrano, W., 2018. The cognitive packet network with QoS and cybersecurity deep learning clusters. In: Intelligent Systems Conference. pp. 62–85.

Serrano, W., Gelenbe, E., 2020. Deep learning clusters in the cognitive packet network. Neurocomputing 396, 406–428.

Song, Z., Yang, J., Mei, X., Tao, T., Xu, M., 2021. Deep reinforcement learning for permanent magnet synchronous motor speed control systems. Neural Comput. Appl. 33, 5409–5418.

Usha, B., Manjunath, T., Mudunuri, T., 2019. Commodity and forex trade automation using deep reinforcement learning. In: International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering. pp. 27–31.

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., Freitas, N., 2016. Dueling network architectures for deep reinforcement learning. In: International Conference on Machine Learning. pp. 1995–2003.

Wu, J., Wang, C., Xiong, L., Sun, H., 2019. Quantitative trading on stock market based on deep reinforcement learning. In: International Joint Conference on Neural Networks. pp. 1–8.

Yan, X., Weihan, W., Chang, M., 2021. Research on financial assets transaction prediction model based on LSTM neural network. Neural Comput. Appl. 33, 257–270.

Yang, M., Qu, Q., Shen, Y., Lei, K., Zhu, J., 2020. Cross-domain aspect/sentiment-aware abstractive review summarisation by combining topic modelling and deep reinforcement learning. Neural Comput. Appl. 32, 6421–6433.

Yu, P., Yan, X., 2020. Stock price prediction based on deep neural networks. Neural Comput. Appl. 32, 1609–1628.

Zhu, Y., Mottaghi, R., Kolve, E., Lim, J., Gupta, A., 2017. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In: IEEE International Conference on Robotics and Automation. pp. 3357–3364.