2022 13th International Congress on Advanced Applied Informatics Winter (IIAI-AAI-Winter) | 979-8-3503-0992-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/IIAI-AAI-WINTER58034.2022.00036

Reinforcement Learning Approach to Forex Trading using Price Changes Trend

Sakulpetch Buaphan
Department of Computer and Information Science
King Mongkut's University of Technology
North Bangkok, Bangkok, Thailand
s6004062856076@email.kmutnb.ac.th

Luepol Pipanmaekaporn
Department of Computer and Information Science
King Mongkut's University of Technology
North Bangkok, Bangkok, Thailand
luepol.p@sci.kmutnb.ac.th

Suwatchai Kamonsantiroj
Department of Computer and Information Science
King Mongkut's University of Technology
North Bangkok, Bangkok, Thailand
suwatchai.k@sci.kmutnb.ac.th

***Abstract*— An automated program that generates profits from the foreign exchange market is attractive to all forex investors. Reinforcement learning has been widely used in financial market trading to maximize profits. In this study, we interested to generate profit from high-frequency currency trading with Deep Q-Network. It has known that the state signals are extracted from forex price data, e.g. mean, maximum, minimum, standard deviation in the interval time. We introduced novel price change trend features that aim to capture directions and speed of price movement over different time and apply it to be the state signal for reinforcement learning. The state signals are consist of the following two parts: the market features that including novel slope of price changes trends features and the agent's current position features. There are three actions –open position, close position, and hold or do nothing position. The reward signal was formulated by tanh function that is used to maximize the profit. We tested our proposed features with Deep Q-Network in the EUR/USD market from 01-Aug-2021 to 30-Dec-2021. The experimental results shown that in comparisons of reinforcement learning with base line features, our proposed features outperforms better than for all performance metrics in both 3000 and 6000 ticks feature vector. These results confirm that the slope of price changes trends features are useful for forecasting under the reinforcement learning approach.**

***Keywords*— DQN, Reinforcement Leaning, Slope of Price Changes, Automated Trading.**

I. Introduction

In foreign exchange markets, commonly recalled as the Forex or FX. The forex is a global marketplace for exchanging national currencies. The currency pairs are traded. In April 2021, the average of daily forex volume increased to $966.7 billion, up 3.6% from the October 2020 [1]. Currency is a type of asset and it has two distinct property. First differential interest rates in different countries. Next, the fluctuation in the exchange rate.

Usually, Trader or people use a variety of strategies to trade in the FX market, such as statistical or algorithmic trading [2]. An automated program that generates profits from the foreign exchange market is attractive to all forex investors. The objective of automated trading is to detect buy or sell signals of market movements.The rules-based approaches are basically applied to create an automated trading, such as the technical analysis of the price time series. If such signals occur, a trading actions is executed to yield a profit. At recent, machine learning techniques have been applied to determine the pattern of market movement [3][4]. Most of these machine learning systems use supervised learning based on historical market data to predict prices of market movements. All supervised learning techniques require the labels data to explain the trend of the future price. However, supervised label take a lot of time and effort to label data. Comparing with supervised learning and unsupervised learning approaches, reinforcement learning has some interested properties. Reinforcement learning is currently the most effective learning model because it can automatically generate labels to data.

Reinforcement learning has to interact and make decisions with an environment. Reinforcement learning uses rewards and punishments as signals for positive and negative behaviour unlike supervised learning that get feedback from the correct labels. The supervised learning learn to establish a best mapping between inputs and outputs while reinforcement learning learn to find appropriate actions that would maximize the total cumulative reward. Reinforcement learning technique was used to apply for automatically trades in the foreign exchange market to maximize the profit [5][7]. The state signal of these papers are features extraction from market data; for example, mean, maximum, standard deviation and minimum value in the interval. In particular, the question of whether the current trend will continue or the current market switches its trend. Many trading algorithm try to identify the current trend or market price movement to forecast whether the current trend either uptrend or downtrend [8][9].

In this paper, we did not predict the end of trend like as the paper [8] but we applied the reinforcement learning to learns in an interactive environment by trial and error using feedback from its own actions and experiences. For each good action, the agent gets positive feedback, and for each bad action, the agent gets negative feedback. We proposed the speed of price movement over different time and trend direction to be the state features for the reinforcement learning. Our experiment setup is following step in the paper [5].

II. Background

A. A price change

A price change in the forex market is a shift in the price value to either a higher or lower level [10]. Let P_Close_t is a forex's closing price on a trading day, P_Close_{t-1} is its closing price on the previous trading day. For example, we assume $P\_Close_{t-1} = 20$ and P_Close_t=15, then a price change is 20-15 = 5.

979-8-3503-0992-8/22/$31.00 ©2022 IEEE
DOI 10.1109/IIAI-AAI-Winter58034.2022.00036

B. Price trends and Slope of price change

A trend is the overall direction of a market price. The market divided into uptrends and downtrends. An uptrend is an overall direction move higher in price. A downtrend describes the price movement when the overall direction is downward. A downtrend can be contrasted with an uptrend. Usually, the price signals contain noise. We need to eliminate random price fluctuations and attempt to only show price trends.

Consider a market price in downtrend. Let P_{low} is the lowest price in this downtrend, P_{cur} is the current price and θ is the threshold pre-assigned by the user. If $P_{cur} > P_{low}$ && $P_{cur} - P_{low} \geq \theta$ then the market switches its direction from downtrend to an uptrend. On the other hand, if $P_{cur} < P_{low}$ then its direction still downtrend and set $P_{low} = P_{cur}$. Similarly, if the market price is in the uptrend, P_{high} is the highest price in this uptrend. If $P_{cur} > P_{high}$ then it still uptrend and set $P_{high} = P_{cur}$. If $P_{cur} < P_{high}$ && $P_{high} - P_{cur} \geq \theta$ then the market is switched its direction form uptrend to downtrend. These definitions are shown in Fig. 1.

Fig. 1 The slope of price change and their definitions

In this paper, we applied the ZigZag indicator in Metatrader software MT4. It is a basic tool for users to identify price trends and changes in price trends. The ZigZag lines are drawn on the price chart when the price movement between P_{low} and P_{high} is greater than a pre-defined threshold. The zigzag' s parameters consist of the Depth, Deviation, and Backstep. First, Depth is the minimal amount of bars where there will not be the second maximum or minimum. Next, Deviation is the minimal distance in percentage between the highs and the lows of the 2 consecutive bars. Finally, Backstep is the minimal amount of bars between maximum/minimum.

Following the above paragraph, Let P_{Low} and P_{High} act as swing high and swing low. These parameters are used to determine the slope of price change $(\Delta P/\Delta T)$. It can interpret as the strength of price movement over different time. In this paper, we proposed novel slope of price changes features that can defined as:

$$\frac{\Delta P}{\Delta T} = \begin{cases} \frac{P_{high} - P_{cur}}{T_{P\_high} - T_{P\_cur}} & \text{if market price in downtrend} \\ \frac{P_{cur} - P_{low}}{T_{P\_cur} - T_{P\_low}} & \text{if market price in uptrend} \end{cases} \quad (1)$$

where $\Delta P/\Delta T$ is the slope of price change, P_{high} is the highest price in the uptrend, P_{low} is the lowest price in the downtrend and T_{P_cur} is the currently time stamp of P_{cur} ,T_{P_high} is the time stamp of P_{high}, and T_{P_low} is the time stamp of P_{low}. The $\Delta P/\Delta T$ is very high it mean that there has been an extreme price change over time or the speed of price movement over different time. It could help a trader to decide whether to take a long or short position.The slope of price change is illustrated in Fig. 1.

C. Reinforcement Learning

Reinforcement learning is one type of machine learning technique. The agent continuously interacting and learning from a complex environment and it learns in an interactive environment by trial and error using feedback from its own actions and experiences. For each good action, the agent gets positive feedback, and for each bad action, the agent gets negative feedback. In this section, we describe how the reinforcement learning model can be formulated.

Given discrete time steps $t = 1,2,3, \ldots$,the agent is interacting with the environment at each of discrete time steps. At each time step t, the agent receives the environment' s state , $s_t \in S$; S is the state space, and then the agent selects an action, $a_t \in A$; A is the action space. Given one time step later $t+1$, the agent obtains a numerical values or rewards, $r_{t+1} \in R$; R is set of possible rewards. After that the state s_t transition to new state $s_{t+1} \in S$ and the agent obtains that new state [11]. This is illustrated in Fig 2.

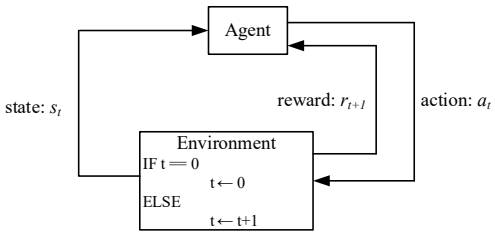

Fig. 2 The agent-environment interaction in Reinforcement learning system

The goal with reinforcement learning is to calculate consequences of agent actions that maximizes the cumulative rewards. Given a time step t, the cumulative rewards defined follow:

$$r_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} = r_{t+1} + \gamma r_{t+1} + \gamma^2 r_{t+1} + \cdots \quad (2)$$

where $\gamma \in [0,1)$ is a discount factor.

A policy dictates the actions that the agent selects. The agent goal is to determine a policy $\pi(s,a)$ as a function of the agent' s state and the environment to optimize the expected reward $Q_t^{\pi}(s,a): S \times A \rightarrow R$. The expected reward rewards or Q-value function as follow :

$$Q_t^{\pi}(s,a) = \mathbb{E}[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a, \pi] \quad (3)$$

Then the optimal Q-value function $Q^*(s,a)$ can be defined as follow:

$$Q^*(s,a) = \text{argmax}\, Q_t^{\pi}(s,a) \quad (4)$$

The algorithm Q-learning is used to estimate the optimal Q-value function $Q^*(s,a)$ and the agent improve its policy by the Bellman optimality equation as follow:

$$Q^*(s,a) = \mathbb{E}\left[r_{t+1} + \gamma \max_{a'} Q^*(s',a') | s_t = s, a_t = a\right] \quad (5)$$

This equation show that the expected reward from starting state $s_t = s$, taking action $a_t = a$. The optimal Q-value $Q^*(s,a)$ is recursive function and it is equal to the expected reward r_{t+1} at time $t+1$ plus the maximum expected discounted $Q^*(s',a')$ from any possible following state-action pair (s',a'). The Q-learning algorithm iteratively updates the Q-values function for each state-action pair until the Q-values function converges to the optimal Q-value $Q^*(s,a)$.

The loss function between the Q-value and the optimal Q-value at each iteration can be given as $Q^*(s',a') - Q(s,a)$. The iteratively updating the Q-value function can be build an approximation of $Q^*(s,a)$ that can be defined as:

$$Q_{k+1}(s,a) = (1-\alpha) * Q_k(s,a) + \alpha\left[r_{t+1} + \gamma \max_{a'} Q^*(s',a')\right] \quad (6)$$

where $Q_{k+1}(s,a)$ is new Q-value function at iteration $k+1$, $Q_k(s,a)$ is old Q-value function at iteration k, and α is the learning rate. By iteratively updating the Q-value function will eventually converge to the optimal Q-value function $Q^*(s,a)$[11].

D. Deep Q-learning

The problem of iteratively updating the Q-value is when the states and actions are continuous or infinite discrete. It becomes impossible to iterate the Q-value in Q-table with finite samples. So the Q-learning with the Q-table is replaced with a deep neural network or deep Q-learning. A deep Q-learning is used to approximate the Q-value function. Let us mention Q-network is a deep neural network function approximator with weights $\mathbb{W}$. The loss functions $L_k(\mathbb{W})$ that changes at each iteration k can defined as:

$$L_k(\mathbb{W}_k) = \mathbb{E}[(y_k - Q(s,a;\mathbb{W}_k))^2] \quad (7)$$

where $y_k = \mathbb{E}\left[r + \gamma \max_{a'} Q^*(s',a':\mathbb{W}_{k-1})\right]$ is the target for iteration k and the parameters from the previous iteration $\mathbb{W}_{k-1}$. Differentiating the loss function with respect to the weights $\mathbb{W}$ defined as:

$$\frac{\partial L_k(\mathbb{W}_k)}{\partial \mathbb{W}_k} = \mathbb{E}\left[\begin{array}{c}(r + \gamma \max_{a'} Q^*(s',a':\mathbb{W}_{k-1}) \\ - Q(s,a;\mathbb{W}_k))\frac{\partial Q(s,a;\mathbb{W}_k)}{\partial \mathbb{W}_k}\end{array}\right] \quad (8)$$

Then formula (8) is the familiar Q-learning algorithm [10].

III. Methodology

A. Market environment

Given a tick data set $T = \{T_0, T_1, \ldots, T_K\}$, A tick data refer to the change in the value from one trade to the next trade. Each tick $T_i \in R^4$ is consist of a list of features such as bid price, ask price, bid volume, and ask volume. Due to a tick data would be too high-frequency for trading, so we allowed to make decision every $time\_skip$ ticks. The $time\_skip$ is the set of tick data. In this paper, we focus on trading EUR/USD and the $time_{skip} = 3000, 6000$ ticks are selected for the market simulation. Our market environment setup is following step in the paper [5].

At the step t, the market environment is at the price in tick T_i. A state s_t is sent to the Reinforcement learning system and an action signal a_t is responded. Then the market environment goes to the price in T_{i+time_skip}. A scalar reward r_{t+1} is calculated and the state s_t is changed to a new state s_{t+1}. At a step t, the agent has only one position or without having position. For example, if agent hold a short-position, then it cannot hold long-position. At each iteration weights update, the samples are randomly selected for these experiment.

B. State signal

The proposed state signal is a 25-dimentional vector that consists of the following two parts:

1) Market feature $\in R^{22}$

With the simulated market at $time_{skip}$ tick $\{T_{i+0*time_{skip}}, T_{i+1*time_{skip}}, T_{i+2*time_{skip}} \ldots, T_{i+n*tim\_skip}\}$, From each $time\_skip$, the set of tick data are divide into 2 sub windows. The first window contain 80% of $time\_skip$ data and the second window contain remain data. These features are extracted:

$$F_i = \begin{vmatrix} \bar{x}^1_{bid} & \bar{x}^2_{bid} & \bar{x}^1_{ask} & \bar{x}^2_{ask} \\ \bar{x}^1_{vol_{bid}} & \bar{x}^2_{vol_{ask}} & max^1_{bid} & max^2_{bid} \\ max^1_{ask} & max^2_{ask} & max^1_{vol_{bid}} & max^2_{vol_{ask}} \\ std^1_{bid} & std^2_{bid} & std^1_{ask} & std^2_{ask} \\ std^1_{vol_{bid}} & std^2_{vol_{bid}} & min^1_{bid} & min^2_{bid} \end{vmatrix} \quad (9)$$

where $\bar{x}^q_p, max^q_p, std^q_p, min^q_p$ are the mean, maximum, standard deviation and minimum values in the interval $time\_skip$, $p \in \{bid, ask, vol\_bid, vol\_ask\}$ is set of all features, and $q \in \{1,2\}$ is an order number of sub window.

- From each $time\_skip$, the $\Delta P/\Delta T$ that defined as equation (1) is added into the feature vector.
- In addition to, the direction of uptrend and downtrend feature is used to indicates the direction of $\Delta P/\Delta T$.

Let $d, d \in \{-1, +1\}$ is set of direction, if the direction is of +1, the direction is the uptrend. If the direction is of -1, the direction is the downtrend.

2) Position feature $\in R^3$

- The agent' s current position $h_t \in \{+1,0,-1\}$ is encode with 1-dimentional vector. If the h_t is of +1, the long position is open. If the h_t is of -1, the short position is currently open. If there is no position open, the h_t is equal to zero.
- The agent' s unrealized profit of the currently open position v_t. In this paper, the v_t is normalized to the range [-1, 1] through tanh function.
- The current account size compared its initial account size c_t. The c_t reveals the status of current account size as safe or failure. The c_t is normalized to the range [-1,1]. A reference [5] provides complete details about c_t.

C. Action signal

Action signals should be actions available to foreign exchange traders, namely, open position, close position and do nothing. In this paper, the agent can only hold one position at a time or without having position. The agent can only invest one unit of position size at a time. Let $A = \{a_0, a_1, a_2\}$ is a set of action signal, where a_0, a_1, a_2 are interpreted by the market environment and it is descripted below.

Action signal procedure:

```
IF action signal == a0:
  IF position == "short" or position == "long":
    action = "hold"
  ELSE:
    action = "nothing"
ELSE IF action signal == a1:
  IF position == "short":
    action = "closeShort"
  ELSE IF position == "long":
    action = "hold"
  ELSE:
    action = "openLong"
ELSE IF action signal == a2:
  IF position == "short":
    action = "hold"
  ELSE IF position == "long":
    action = "closeLong"
  ELSE:
    action = "openShort"
```

D. Reward signal

In this paper, there are two cases in calculating the reward. Calculating the reward depends the action and current position. If the action is the close positions (closeLong or closeShort) , the reward equal to $unrealized\_profit_t \in [-1,1]$. If the action is the open positions (openLong or openShort) or hold position, the reward equal to $(unrealized\_profit_{t+1} - unrealized\_profit_t) \in [-1,1]$. The reward signal is normalized to the range $[-1,1]$ with tanh function for gradient learning stability. The reward signal is calculated as:

Reward signal procedure:

IF(action== 'hold') :
  rewardReturn = $unrealized\_profit_{t+1}$ - $unrealized\_profit_t$
  return tanh(rewardReturn)
ELSE IF (action == 'closeLong' or action == 'closeShort') :
  rewardReturn = $unrealized\_profit_t$
  return tanh(rewardReturn)
ELSE:
  return 0

E. Deep-Q network training

The training process defines the neural network' architecture and tunes the hyperparameter to achieve optimal Q-value function. The network compute an approximation of Q- value function $Q_a(s; \mathbb{W}_k)$ for the market environment. The network is specified in Fig. 3. The mean squared error is used as the loss function. The optimizer is Adam optimizer. The learning rate is 10^{-3} and the training is carried out for 10 episodes in mini batchs of 512.

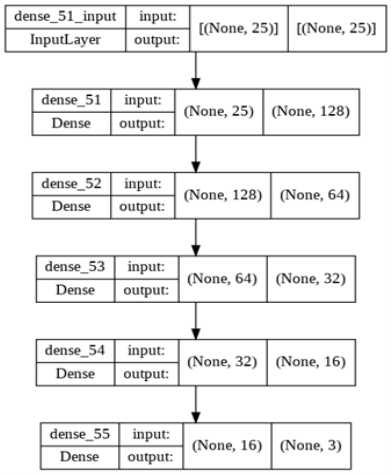

Fig. 3 The neural network' architecture for training process

IV. EXPERIMENTAL AND PERFORMANCE ANALYSIS

A. DataSet

The proposed method was evaluated using a financial data set contain ticks price data starting from 01-Aug-2021 to 30-Dec-2021 that were collected by MT4 Meta trader. The data collection was split into two sets: a training and a testing set. The training set start 01-Aug-2021 to 31-Oct-2021. There are 2 sets for testing the model which are denoted as

Nov-2021 and Dec-2021. The Nov-2021 set contain ticks price data starting from 01-Nov-2021 to 30-Nov-2021 and the Dec-2021 set contain ticks price data starting from 01-Dec-2021 to 30-Dec-2021.

We compared our experimental results that obtained with our proposed features with baseline of automatic forex trading using different features. There are three vectors set that are described as follows :

- h v c F (hvcF) : the vector consist of list of features h_t, v_t, c_t and F_t.
- h v c F $\frac{\Delta P}{\Delta T}$ (hvcFΔP) : the vector consist of list of features h_t, v_t, c_t, F_t, and $\frac{\Delta P}{\Delta T}$
- h v c F $\frac{\Delta P}{\Delta T}$ d (hvcFΔPd) : the vector consist of list of features h_t, v_t, c_t, F_t, $\frac{\Delta P}{\Delta T}$ and d

In this paper, we used these features vector act as the input layer of reinforcement learning and the Deep Q-network was applied to evaluate the model performance.

B. Performance evaluation

1) Training Result

Table II shows comparison learning results for three different features for reinforcement learning. We used two $time\_skip$ i.e.3000 and 6000 ticks for simulate possible market environment. The loss function of baseline features versus the proposed features is illustrated in Table II.

Table II The training result of baseline Features versus the proposed features

Ticks	Model	Train Aug - Oct 2021
		Loss
3000	hvcF	0.097
	hvcFΔP	0.095
	hvcFΔPd	**0.095**
6000	hvcF	0.098
	hvcFΔP	0.098
	hvcFΔPd	**0.098**

As shown in Table II, First, we considered the experimental with 3000 ticks. The loss of hvcF ΔP and hvcFΔPd is 0.095 and 0.095, respectively. The loss of hvcF is 0.097. The result show that the proposed features $\Delta P/\Delta T$and direction outperform baseline features. The experimental with 6000 ticks, the loss function value were similar to the experimental with 3000 ticks.

2) Testing Result

In this experimental, the metric used to evaluate our result consists of the average profit (Avg_profit) , average profit relative (Avg_Rel_Profit) , average max drawdown (Avg_MaxDD), and average number of orders. Avg_profit is calculated the average effective profit or loss of an agent that are recommended action from Deep RL model.Avg Rel_Profit is average profit relative to the initial account size The max drawdown of profits is calculated as the maximum absolute difference of the highest peak in profits to the lowest following drop. The average number of orders is shown as the average number of order. The reward return are calculated as change in the price of the equity between the first and last date of the given testing period. The performance of baseline versus the proposed features are illustrated in Table III.

Table III The performance of baseline features versus the proposed features

Ticks	Model	Test Set#1 (Nov-2021)				
		Avg Profit (USD)	Avg Rel_ Profit (%)	Avg MaxDD (USD)	Avg Number of Orders	
					Long	Short
3000	hvcF	3.83	0.03	9.31	0	1.40
	hvcFΔP	65.32	0.65	46.75	26.80	2.20
	hvcFΔPd	**95.23**	**0.95**	**68.61**	**15.50**	**20.30**
6000	hvcF	23.39	0.23	9.62	2.70	0.20
	hvcFΔP	66.72	0.66	35.99	0	6.30
	hvcFΔPd	**116.36**	**1.16**	**78.78**	**4.70**	**21.30**

Ticks	Model	Test Set#2 (Dec-2021)				
		Avg Profit (USD)	Avg Rel_ Profit (%)	Avg MaxDD (USD)	Avg Number of Orders	
					Long	Short
3000	hvcF	0.00	00.0	0.00	0	0
	hvcFΔP	30.60	0.31	127.30	13.00	0
	hvcFΔPd	**105.06**	**1.05**	**109.99**	**31.50**	**18.40**
6000	hvcF	20.22	0.20	42.24	6.20	0
	hvcFΔP	30.19	0.30	53.49	0	6.20
	hvcFΔPd	**121.75**	**1.22**	**112.75**	**13.50**	**18.00**

As shown in Table III, First, we considered the features with 3000 ticks from Test Set#1(Nov-2021) . The outcome of hvcF offers average profit of 3.83, average relative profit of 0.03% and average MaxDD of 9.31, Average number of long position is 0 order and average number of short position is 1.4 orders.

The outcome of hvcFΔP offers average profit of 65.32, average relative profit of 0.65% and average MaxDD of 46.75, Average number of long position is 26.80 orders and average number of short position is 2.20 orders.

The outcome of hvcFΔPd offers average profit of 95.23, average relative profit of 0.95% and average MaxDD of 68.61, Average number of long position is 15.50 orders and average number of short position is 20.30 orders.

In comparisons of Reinforcement learning with hvcFΔP or hvcFΔPd with HvcF, our proposed features performs better than for all performance metrics.The remaining experiment, the performance metric were similar to the result with 3000 ticks. These results support that the slope of price change and direction significantly increase reward for automatic financial trading market.

V. CONCLUSION

This paper presents a novel features that is the slope of price change and direction of uptrend or downtrend to improve performance evaluation for automatic trading on financial markets to maximize profit. The proposed features can interpret as the strength of price movement over different time. It may

be directly modified to generate new robust features. The experimental results shows that the proposed features achieves improvements of average absolute profit and average relative profit in comparing with reference features based on reinforcement learning. The researchers believe that the presented features is a novel approach for the financial trading in the future direction.

VI. Reference

[1] Saqib Iqbal Ahmed. "N. America average daily FX volume rose in April 2021-NY Fed FX survey." reuters.com. https://www.reuters.com/article/us-global-forex-volume-idUSKBN2EX23C (accessed Oct. 1,2022).

[2] Benjamin Anderson, Thar Min Htet, Nicholas Nugent and Andrew VanOsten, " Forex Trading Systems: An Algorithmic Approach to Technical Trading," Worcester Polytechnic Institute, Accessed: Oct. 5, 2022.

[3] Thuy Nguyen Thi Thu, Vuong Dang Xuan. FoRex Trading Using Supervised Machine Learning. International Journal of Engineering & Technology.

[4] Laurids Gert Nielsen, "MACHINE LEARNING FOR FOREIGN EXCHANGE RATE FORECASTING," MSc dissertation, Department of Mathematics., Imperial College London., London, United Kingdom, 2017-2018.

[5] João Carapuço, Rui Neves and Nuno Horta, "Reinforcement learning applied to Forex trading" in Applied Soft Computing Journal., Volume 73, Dec. 2018, pp. 783-794

[6] Francesco Rundo. (Oct. 2019). Deep LSTM with Reinforcement Learning Layer for Financial Trend Prediction in FX High Frequency Trading Systems.

[7] C. Gold. (Apr. 2003). FX trading via recurrent reinforcement learning. Proceedings in IEEE International Conference on Computational Intelligence for Financial Engineering.

[8] Amer Bakhach, Edward P. K. Tsang, Hamid Jalalian. (Dec. 2016). Forecasting Directional Changes in the FX Markets. IEEE Symposium Series on Computational Intelligence (SSCI).

[9] Aloud, Monira and Tsang, Edward P. K. and Olsen, Richard B. and Dupuis, Alexandre. A Directional-Change Events Approach for Studying Financial Time Series. Economics Discussion Paper No. 2011-28.

[10] Carla Tardi. "Price Change." investopedia.com. https://www.investopedia.com/terms/p/price-change.asp (accessed Oct. 1,2022).

[11] Richard S. Sutton and Andrew G, Reinforcement Learning: An Introduction Second Edition MIT Press. Cambridge, MA, USA: MIT Press, 2015.

[12] Christopher JCH Watkins and Peter Dayan, "Q-learning," in Kluwer Academic Publishers, Boston, Manufactured, Netherlands: KAP, 1992,pp. 279-292.