A thesis submitted in partial satisfaction of the requirements
for the degree of Master of Computer Science and Engineering
in the Graduate School of the University of Aizu

Deep Reinforcement Learning in Forex Trading Using Metrics

by

HARUGUHCI Takuma

March 2021

The thesis titled

Deep Reinforcement Learning in Forex Trading Using Metrics

by

HARUGUHCI Takuma

is reviewed and approved by:

Main referee

Associate Professor

LI Xiang

Professor

MORI Kazuyoshi

Senior Associate Professor

JING Lei

THE UNIVERSITY OF AIZU

March 2021

Contents

Chapter 1 Introduction **1**
1.1 Overview 1
1.2 Forex Trading 1
1.2.1 Details of Forex System 1
1.2.2 Simplification 3
1.3 Reinforcement Learning 5
1.4 Q-learning 6
1.5 Deep Q-learning 6
1.6 Previous Research and Motivation 6

Chapter 2 Method and Algorithm **8**
2.1 Overview 8
2.2 State 8
2.3 Deviation of Exchange Rate Data 9
2.4 Action 9
2.5 Action Modification 10
2.6 Episode 10
2.7 P/L and Reward 10
2.8 DQN 10

Chapter 3 Experiment **12**
3.1 Dataset 12
3.2 Evaluation Method 13
3.3 Details 13

Chapter 4 Result **14**
4.1 Accumulated Reward 14
4.2 Waiting Ratio 15

Chapter 5 Conclusion **21**

References **22**

List of Figures

Figure 1.1 The overview of Forex entities 2
Figure 1.2 Timeline for long position when getting profit 2
Figure 1.3 Timeline for long position when suffering loss 3
Figure 1.4 Timeline for short position when suffering loss 4
Figure 1.5 Timeline for short position when getting profit 4
Figure 1.6 The diagram of the position state transition 5
Figure 1.7 The agent–environment interaction in a MDP [1] 5
Figure 1.8 Difference between Q-learning and DQN [2] 7

Figure 2.1 RL for Forex trading 8

Figure 4.1 Accumulated reward in training for the number of moving average (MA) 14
Figure 4.2 Accumulated reward in testing for the number of MA 15
Figure 4.3 Waiting ratio in training for the number of MA 16
Figure 4.4 Average of waiting ratio in testing for the number of MA 16
Figure 4.5 Scatter plot between waiting ratio and accumulated reward in training of MA0 17
Figure 4.6 Scatter plot between waiting ratio and accumulated reward in training of MA1 17
Figure 4.7 Scatter plot between waiting ratio and accumulated reward in training of MA2 17
Figure 4.8 Scatter plot between waiting ratio and accumulated reward in training of MA3 18
Figure 4.9 Scatter plot between waiting ratio and accumulated reward in training of MA4 18
Figure 4.10Scatter plot between waiting ratio and accumulated reward in training of MA5 19
Figure 4.11Waiting ratio in testing for the number of MA 19
Figure 4.12Average of waiting ratio in testing 20

List of Tables

Table 1.1 P/L and cover deal for long position when getting profit 2
Table 1.2 P/L and cover deal for long position when suffering loss 3
Table 1.3 P/L and cover deal for short position when suffering loss 4
Table 1.4 P/L and cover deal for short position when getting profit 4

Table 3.1 Dataset of training and testing 13

Table 4.1 The average of accumulated reward in the last ten trainings for each MA . 14

Acknowledgment

I express our sincere thanks to Prof. LI Xiang for helpful discussions and suggestions during COVID-19 pandemic. Also, Referees Prof. MORI Kazuyoshi and Prof. JING Lei gave me advice for my thesis. I also thank the lab members.

Abstract

In recent years, Deep Q-learning (DQL) becomes more and more important to any field. Deep Q-learning, which is also one of deep reinforcement learning (DRL), is Q-learning with Deep Q Network (DQN). Before employing reinforcement learning for finance in earnest, the papers that applied machine learning tended to focus on predicting the future. The weakness in the predictive approach is to ignore the option to wait. Therefore, this thesis studies DQN for the retail foreign exchange trading (Forex, A.K.A. FX). The DQN utilizes multiple moving averages (MA) of the exchange rate history like USD/JPY as state element. Based on the results, this thesis presented two conclusions. Firstly, unfortunately, the agent failed to learn to get profit in both training and testing because MA may be invalid metrics for Forex trading. Secondly, reinforcement learning itself was presumed to help to avoid losses. Accordingly, further investigation is needed to make this DQN method practical.

Chapter 1

Introduction

1.1 Overview

Foreign exchange is trading of currencies between two countries. For example, when a company in Japan imports some products from the U.S., it exchanges the yen into the U.S. dollar to pay.

In addition to importers and exporters, there are speculators in Forex markets. Individual traders of them often utilize the retail foreign exchange trading (Forex, A.K.A. FX). In Forex, expecting the exchange rate among currencies and taking the long or the short position, a trader tries to get the gain from the difference of the rate between present and future.

The long position is the buying position where he gets the profit if the rate rises (e.g., \$1=¥100 $\Longrightarrow$ \$1=¥120: Profit=¥20). In the opposite case, he suffers a loss. On the other hand, The short position is the selling position where he gets the profit if the rate decrease (e.g., \$1=¥100 $\Longrightarrow$ \$1=¥90 : Profit=¥10). In the opposite case, he suffers a loss. Taking no position is called the square, which means that you get no profit and suffer no loss.

Instead of prediction approach for the future rate, this paper focuses on the strategy to get the profit using Deep Q-learning. The contribution of the thesis is to confirm the performance of moving average as state in the learning because moving average is one of the most basic metrics in the Forex technical analysis.

This thesis has been organized as follows. The rest of this section describes Forex trading system, reinforcement learning, and research motivations. In Section 2, the algorithm and definitions about Deep Q-learning are detailed. In Section 3, the way of the experiment, dataset, and the evaluation methods are detailed. Section 4 shows the results of the experiment and considers the meanings of them. In Section 5, conclusions are discussed.

1.2 Forex Trading

1.2.1 Details of Forex System

First, we look at the entities of Forex. In Figure 1.1 that simplifies the actual situation [3], there are traders, Forex companies, and the interbank market. Traders order a Forex company to take a long/short position or to liquidate the position, and then the Forex company conducts the cover deal in the interbank market.

Ignoring the revenue sources of the forex company such as transaction fee, let us consider the relationship among the long/short position, profit and loss (P/L), and the cover deal.

Figure 1.2 shows that trader A makes a profit of ¥ 20 on taking the long position. When the trader takes the long position at time t_1, the Forex company exchanges yens for dollars as cover deal at time t_2. In general, the cover deal means the real trading in the interbank market corresponding to a trader's position.

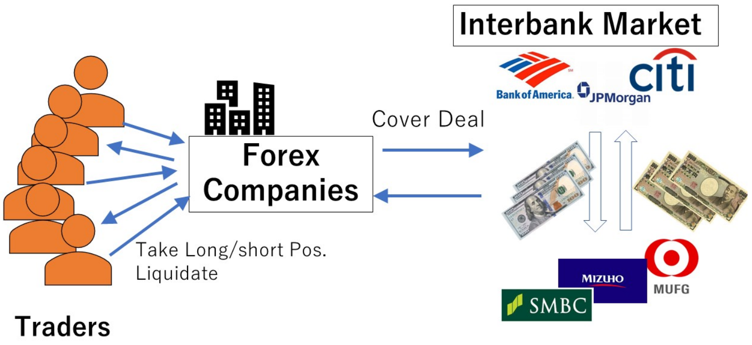

Figure 1.1: The overview of Forex entities

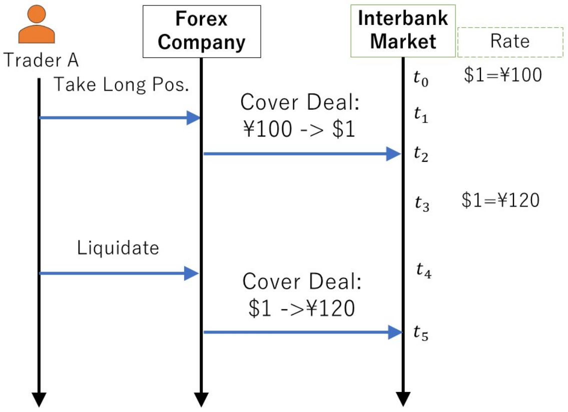

Figure 1.2: Timeline for long position when getting profit

Table 1.1: P/L and cover deal for long position when getting profit

	Trader A			
Time	Position	Floating P/L	P/L	Forex Capital
t_0	Square	¥0	¥0	¥100
t_1	Long	¥0	¥0	¥100
t_2	Long	¥0	¥0	$1
t_3	Long	+¥20	¥0	$1
t_4	Square	¥0	+¥20	$1
t_5	Square	¥0	+¥20	¥120 - ¥20 = ¥100

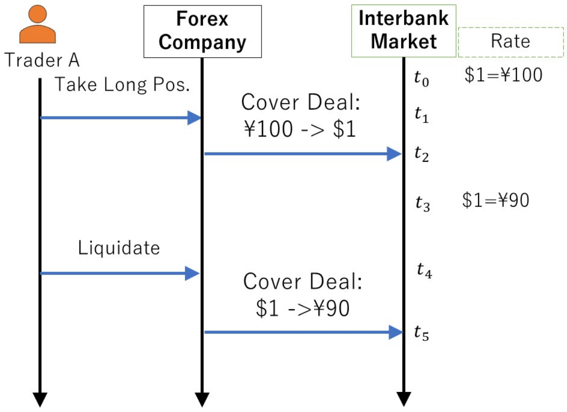

Figure 1.3: Timeline for long position when suffering loss

Table 1.2: P/L and cover deal for long position when suffering loss

Time	Trader A			Forex Capital
	Position	Floating P/L	P/L	
t_0	Square	¥0	¥0	¥100
t_1	Long	¥0	¥0	¥100
t_2	Long	¥0	¥0	$1
t_3	Long	-¥10	¥0	$1
t_4	Square	¥0	-¥10	$1
t_5	Square	¥0	-¥10	¥90 + ¥10 = ¥100

And then, suppose that the exchange rate $1=¥100 changes to $1=¥120 at time t_3. The rate change brings the trader floating P/L. The floating P/L is unrealized profit or loss which floats (changes) in correspondence with the exchange rate and which position he has. For example, if the exchange rate $1=¥120 at time t_3 changes to $1=¥110 at time $t_{3.5}$ unlike Figure 1.2, his floating P/L as +¥20 also changes to +¥10.

At time t_4, the trader liquidates his position to finally realize the profit as +¥20. Liquidating a position means changing the position into the square, finalizing his P/L. And then, the Forex company exchanges dollars for yen as cover deal at time t_5 to pay the trader ¥20.

There are four patterns for the taking positions and trader's P/L:

1. Long position when getting profit $\Longrightarrow$ Figure 1.2 and Table 1.1
2. Long position when suffering loss $\Longrightarrow$ Figure 1.3 and Table 1.2
3. Short position when suffering loss $\Longrightarrow$ Figure 1.4 and Table 1.3
4. Short position when getting profit $\Longrightarrow$ Figure 1.5 and Table 1.4

Look at following figures and tables to understand all relationships among positions, P/L, and the cover deal.

The four patterns suggest two features. Firstly, the Forex company keeps the initial capital as ¥100 or $1 without any profit and any loss in any case. It shows that the cover deal literally covers the Forex company from the loss.

Secondly, the position has the state transition as Figure 1.6.

1.2.2 Simplification

Within the scope of this research, all you have to do is to learn the below rules:

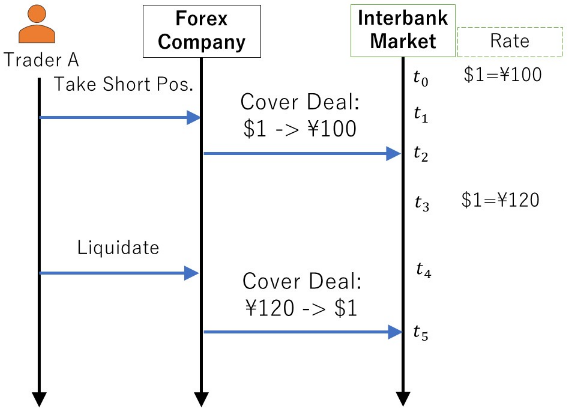

Figure 1.4: Timeline for short position when suffering loss

Table 1.3: P/L and cover deal for short position when suffering loss

	Trader A			
Time	Position	Floating P/L	P/L	Forex Capital
t_0	Square	¥0	¥0	$1
t_1	Short	¥0	¥0	$1
t_2	Short	¥0	¥0	¥100
t_3	Short	-¥20	¥0	¥100
t_4	Square	¥0	-¥20	¥100
t_5	Square	¥0	-¥20	¥100 + ¥20 = $1

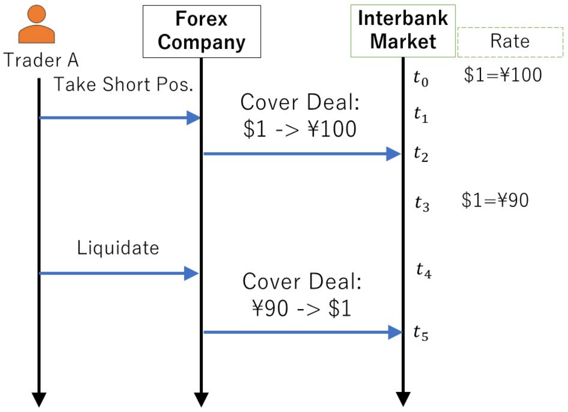

Figure 1.5: Timeline for short position when getting profit

Table 1.4: P/L and cover deal for short position when getting profit

	Trader A			
Time	Position	Floating P/L	P/L	Forex Capital
t_0	Square	¥0	¥0	$1
t_1	Short	¥0	¥0	$1
t_2	Short	¥0	¥0	¥100
t_3	Short	+¥10	¥0	¥100
t_4	Square	¥0	+¥10	¥100
t_5	Square	¥0	+¥10	¥100 - ¥10 = $1

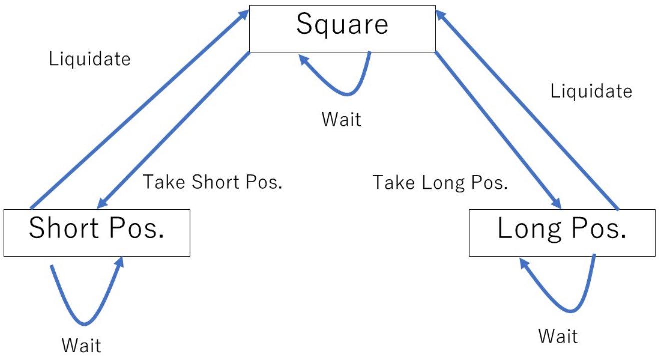

Figure 1.6: The diagram of the position state transition

1. Position State Transition as Figure 1.6

2. Position and floating P/L

   - Long position: The dollar rate rises $\Longrightarrow$ Profit as floating P/L
   - Long position: The dollar rate decreases $\Longrightarrow$ Loss as floating P/L
   - Short position: The dollar rate rises $\Longrightarrow$ Loss as floating P/L
   - Short position: The dollar rate decreases $\Longrightarrow$ Profit as floating P/L

3. The P/L is finally realized after liquidating the position.

You do not have to consider the Forex company because it does not affect the trader's P/L.

1.3 Reinforcement Learning

Reinforcement learning (RL) is one of machine learning which learns mapping the pairs of situations-to-actions so as to maximize a reward [1]. In general, the reinforcement learning is modeled as Markov decision process (MDP) like Figure 1.7. At the very beginning, the agent receives the state S_0 from the environment, and then the agent takes the action A_0. The environment gives the agent the reward R_1 and the state S_1 in turn, and then the agent takes the action A_1, and so on. The (1.1) shows the trajectory:

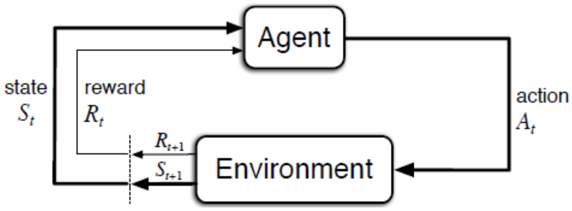

Figure 1.7: The agent–environment interaction in a MDP [1]

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, ..., R_{T-1}, S_{T-1}, A_{T-1}, R_T, S_T, \tag{1.1}$$

where T is a final time step [1]. The range of time steps between 0 and T is called *episode* [1].

1.4 Q-learning

One of the methods in RL is Q-learning. Before explaining it, we introduce some related equations. First, let us define q_π which is called *action-value function for policy* π [1] such as

$$q_\pi(s,a) := \mathbb{E}[G_t|S_t = s, A_t = s]. \tag{1.2}$$

Equation (1.2) means that $q_\pi(s,a)$ outputs the expected return starting from state s and taking the action a, after that following policy π. A policy π is a rule where the agent determines the action, and the policy is calculated as conditional probability $\pi(a|s)$.

Second, we introduce *optimal action-value function* q_*, and define it as

$$q_*(s,a) := max\ q_\pi(s,a), \tag{1.3}$$

for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$ where $\mathcal{S}$ is set of all nonterminal states and the $\mathcal{A}(s)$ is set of all actions available in state s [1].

Third, based on Equation 1.3, it is known to be able to obtain the *Bellman optimality equation* for q_* [4] which is

$$q_*(s,a) = \sum_{s' \in \mathcal{S}} p(s'|s,a)[r(s,a,s') + \gamma \max_{a' \in \mathcal{A}(s')} q_*(s',a')]. \tag{1.4}$$

The $p(s'|s,a)$ is probability of transition to state s', from state s taking action a, the $r(s,a,s')$ is expected immediate reward on transition from s to s' under action a, and the γ is discount-rate parameter [1].

Lastly, let us consider Q-learning [4] [1] [5] which is an algorithm defined by

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_{a' \in \mathcal{A}(S_{t+1})} Q(S_{t+1}, a') - Q(S_t, A_t)], \tag{1.5}$$

where the α is learning rate.

The (1.5) shows that Q-learning iterates updating the action-value function Q to directly approximate q_*. When the learning converges, the second term of the (1.5) converges to zero, which means approximating q_* since the $R_{t+1} + \gamma \max_{a' \in \mathcal{A}(S_{t+1})} Q(S_{t+1}, a')$ in the (1.5) is similar to Equation (1.4).

The results of updated the action-value function Q are called Q-value and stored into the table which is called Q-table shown as the left side of Figure 1.8.

1.5 Deep Q-learning

It is difficult for Q-learning to solve the problem which has a large state space since the size of Q-table becomes huge [2]. This is because all Q-values are allocated as the entire combination of both action and state which are discrete value. Its trouble is called as the curse of dimensionality.

Deep Q Network (DQN) [6], which is a neural network used by Deep Q-learning, can solve the trouble [2]. As the right side of Figure 1.8 shows, Deep Q-learning regards each state element as each DQN input node, which means reducing the size of calculating Q-value. DQN outputs only each Q-value for each action.

1.6 Previous Research and Motivation

This paper includes three motivations. First motivation explains the reason to employ RL for Forex trading.

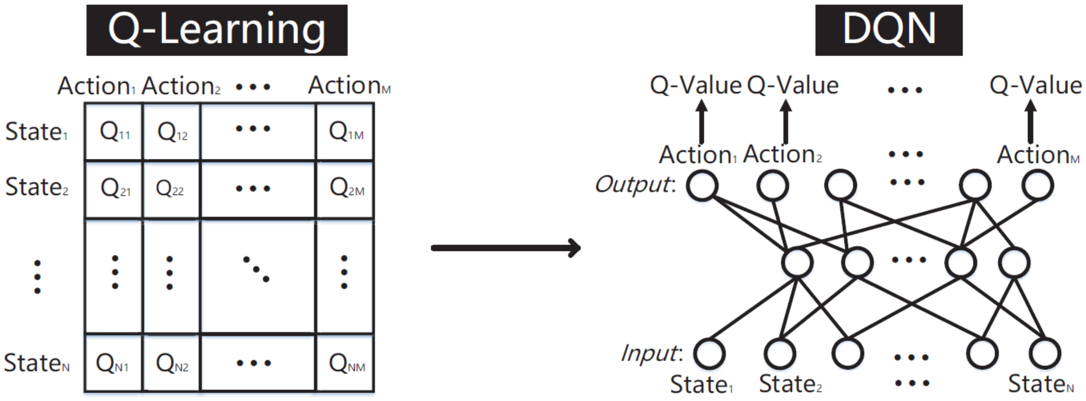

Figure 1.8: Difference between Q-learning and DQN [2]

Before employing RL for finance in earnest, the papers that applied machine learning tended to focus on predicting the future. For example, Arash's survey [7] showed that many papers about trading were related to prediction.

The weakness in the predictive approach is to ignore the option to wait. For example, when a trader cannot be sure the direction of the exchange rate, the best strategy should be to wait without bringing any profit and any loss. However, the predictive approach completely disregards that option.

On the other hand, RL allows the agent to consider waiting as part of actions. This is the reason to employ RL in my research.

Second motivation is utilization of DQN. In terms of RL modeling, Forex trading can be characterized by the continuous of the state while the action is discrete. This is because, in most cases, the state definition includes the exchange rate history which is continuous. On the other hand, the action can be defined as discrete like the position transition in Figure 1.6.

As mentioned in Section 1.5, DQN is suitable for the modeling where the state and the action are defined as continuous and discrete respectively. This is why this research employs DQN.

The third motivation is to confirm the effect of metrics. Among Forex technical analysis, many metrics are utilized. This research focuses on simple moving average (MA) which is one of the most basic metrics in the analysis [8]. The experiment verifies whether MA as state element of RL improves the performance of the agent.

Chapter 2

Method and Algorithm

2.1 Overview

This paper employs RL as Figure 2.1. To maximize the reward, the Q-values are calculated as the right side of Figure 1.8 using DQN.

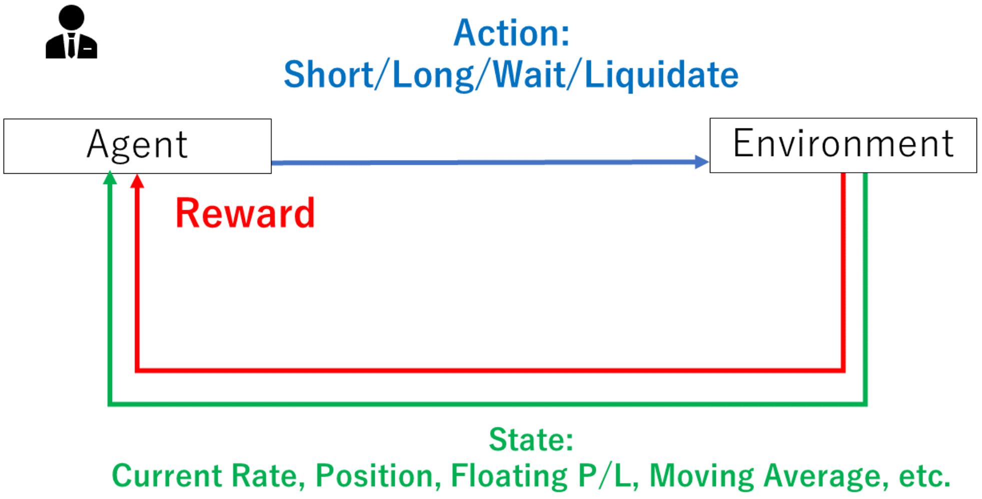
Action:
Short/Long/Wait/Liquidate
Agent
Environment
Reward
State:
Current Rate, Position, Floating P/L, Moving Average, etc.

Figure 2.1: RL for Forex trading

2.2 State

The *state element* is defined as Equation (2.1)

$$\boldsymbol{S_t} = (cu_t, pos, ma1_t, ma2_t, ma3_t, ma4_t, ma5_t, fpl_t, pos\_rate). \tag{2.1}$$

The subscript t means *time step*, and the same applies hereafter. The t starts with 120 to calculate the moving average $ma1_t$.

The cu_t means *current exchange rate*.

Denoted as pos, the (2.2) defines *current position* as

$$pos \in \{SQUARE, SHORT, LONG\}, \tag{2.2}$$

where the agent is in.

Equation (2.3) means *moving averages*:

$$\begin{aligned} ma1_t &= \tfrac{1}{120}\sum_{i=t-120}^{t} p_i, \\ ma2_t &= \tfrac{1}{80}\sum_{i=t-80}^{t} p_i, \\ ma3_t &= \tfrac{1}{50}\sum_{i=t-50}^{t} p_i, \\ ma4_t &= \tfrac{1}{30}\sum_{i=t-30}^{t} p_i, \\ ma5_t &= \tfrac{1}{20}\sum_{i=t-20}^{t} p_i. \end{aligned} \tag{2.3}$$

The p_i means the exchange rate (price) at the time step i.

Equation (2.4) means *floating P/L* which corresponds to Section 1.2.1:

$$fpl_t = \begin{cases} cu_t - pos\_rate \ (pos = LONG) \\ pos\_rate - cu_t \ (pos = SHORT) \\ 0 \ (otherwise) \end{cases} . \tag{2.4}$$

The pos_rate means the *exchange rate when taking the position*. For example, the $pos\_rate = 112.55$ means that the 1 dollar had equaled 112.55 yen when the agent had taken the long or the short position. If the agent is in the square, the pos_rate becomes zero.

2.3 Deviation of Exchange Rate Data

In fact, all data of the exchange rate is normalized as Equation 2.5

$$p_t = d_t - \frac{1}{N}\sum_{i=1}^{N} d_i, \tag{2.5}$$

where the N means the time period of the entire data history. The d_t means original data of the dataset in Section 3.1. The time period N depends on the number of the data in Table 3.1.

The normalization narrows the range between maximum and minimum in the state space to reduce computational complexity.

2.4 Action

The *action set* $\mathcal{A}$ is defined as Equation (2.6):

$$\begin{aligned} \mathcal{A}(\boldsymbol{pos} = \boldsymbol{SQUARE}) &= \{WAIT, SHORT, LONG\} \\ \mathcal{A}(\boldsymbol{pos} = \boldsymbol{ohterwise}) &= \{WAIT, LIQUIDATE\}. \end{aligned} \tag{2.6}$$

Equation (2.6) corresponds to the position transition as Figure 1.6.

The (2.7) shows each *action element* A_t belongs to the action set. Each action element A_t means taken action at time t.

$$A_t \in \mathcal{A}(\boldsymbol{pos}) \tag{2.7}$$

2.5 Action Modification

Despite Equation 2.6, the agent may take wrong action. For example, he may liquidate wrongly even when he is in the square. This is because the agent must learn the position transition of Figure 1.6 although the transition is deterministic.

Therefore, the RL system is implemented to replace a wrong action with the *wait* action as below.

- In Square: the agent wrongly *liquidates* $\Longrightarrow$ *wait* action
- In Short of Long position: the agent wrongly takes *long* or *short* position $\Longrightarrow$ *wait* action

In addition, the RL system forces the agent to liquidate his position forcefully when one episode finishes. If the system does not, the agent can keep waiting to avoid losses in any case even if he takes a long or short position.

2.6 Episode

The final time step T of one episode is defined as Equation (2.8)

$$T = 1200. \tag{2.8}$$

One *episode* is defined as the (2.9)

$$\boldsymbol{S_0}, A_0, \quad R_1, \ \boldsymbol{S_1}, A_1, \quad R_2, \ \boldsymbol{S_2}, A_2, \quad ..., \quad R_{T-1}, \ \boldsymbol{S_{T-1}}, A_{T-1}, \quad R_T, \ \boldsymbol{S_T}, \tag{2.9}$$

One *step* is defined as Equation (2.10)

$$step = \begin{cases} \boldsymbol{S_t}, A_t \quad (t = 0) \\ R_t, \ \boldsymbol{S_t}, A_t \ (otherwise) \end{cases}, \tag{2.10}$$

2.7 P/L and Reward

Equation (2.11) means profit and loss (P/L) which corresponds to Section 1.2.1:

$$profit = \begin{cases} (cu_t - pos\_rate) \times 10000 \ (pos = LONG \ \wedge \ a = LIQUIDATE) \\ (pos\_rate - cu_t) \times 10000 \ (pos = SHORT \ \wedge \ a = LIQUIDATE) \\ 0 \ (otherwise) \end{cases}, \tag{2.11}$$

where $\times 10000$ is leverage to amplify P/L. The negative profit means losses.

Note that the cu_t is the current exchange rate of time t, not $t+1$ of cover deal in Section 1.2.1. It is simplification in order to make implementation easy.

In this research, the *reward* is defined as same as the profit. Note that the leverage enables the agent to learn because the normalization in Section 2.3 makes the reward without the leverage close to zero.

2.8 DQN

As agent of RL, this research utilizes DQN [6] where the policy is Boltzmann Q Policy, or soft-max policy.

The neural network is constructed with Keras as below:

Listing 2.1: Neural network structure with Keras

```
Layer (type)                 Output Shape              Param #
=================================================================
flatten_1 (Flatten)          (None, 9)                 0
_________________________________________________________________
dense_1 (Dense)              (None, 16)                160
_________________________________________________________________
activation_1 (Activation)    (None, 16)                0
_________________________________________________________________
dense_2 (Dense)              (None, 16)                272
_________________________________________________________________
activation_2 (Activation)    (None, 16)                0
_________________________________________________________________
dense_3 (Dense)              (None, 16)                272
_________________________________________________________________
activation_3 (Activation)    (None, 16)                0
_________________________________________________________________
dense_4 (Dense)              (None, 4)                 68
_________________________________________________________________
activation_4 (Activation)    (None, 4)                 0
=================================================================
Total params: 772
Trainable params: 772
Non-trainable params: 0
```

In Listing 2.1, the flatten_1 layer is the input layer which takes the state elements as Equation 2.1. The activation_1, activation_2, and activation_3 layers utilize ReLU (Rectified Linear Unit) activation [9], and the activation_4 layer uses linear activation [10]. The dense_1, dense_2, dense_3 and dense_4 layers are densely-connected (fully-connected) neural network layers [11]. The activation_4 corresponds to the output layer on the right side of Figure 1.8 to decide to take an action as WAIT, SHORT, LONG or LIQUIDATE.

Chapter 3

Experiment

The agent is trained based on Section 2 by 50,000 steps trading on the testing dataset which mostly equals 47 episodes. Each training episode consists of same data. After that, the agent is tested by trading of one episode on nine testing datasets. Note that each testing dataset make each environment interact with the agent exactly the same every episode. This is why testing consists of one episode.

To confirm the effect of MA as metrics, the training and nine testing vary the number of MA from zero to five. For example, when the number of MA is four which is denoted as MA4, the MA elements are changed as Equation 3.1. When the number of MA is three which is denoted as MA3, the MA elements are changed as Equation 3.2, and so on.

$$\begin{aligned} ma1_t &= \tfrac{1}{120} \sum_{i=t-120}^{t} p_i, \\ ma2_t &= \tfrac{1}{80} \sum_{i=t-80}^{t} p_i, \\ ma3_t &= \tfrac{1}{50} \sum_{i=t-50}^{t} p_i, \\ ma4_t &= \tfrac{1}{30} \sum_{i=t-30}^{t} p_i, \\ ma5_t &= 0 \end{aligned} \tag{3.1}$$

$$\begin{aligned} ma1_t &= \tfrac{1}{120} \sum_{i=t-120}^{t} p_i, \\ ma2_t &= \tfrac{1}{80} \sum_{i=t-80}^{t} p_i, \\ ma3_t &= \tfrac{1}{50} \sum_{i=t-50}^{t} p_i, \\ ma4_t &= 0, \\ ma5_t &= 0 \end{aligned} \tag{3.2}$$

3.1 Dataset

The dataset consists of the USD/JPY (US Dollar vs. Japanese Yen) exchange rate where the candlestick range is five minutes. It is comprised of the training dataset and nine testing datasets as Table 3.1 shows. Note that the trained/tested data are not the entire ones but 1200 ones that correspond to Equation 2.8, which is listed as Trained/Tested Data Period column on Table 3.1.

Table 3.1: Dataset of training and testing

Dataset Name	Period (Day.Month.Year, GMT)	Number of Data	Trained/Tested Data Period
Train	01.01.2004 - 09.01.2004	1908	01.01.2004 - 07.01.2004
Test 1	11.01.2004 - 20.01.2004	2040	11.01.2004 - 16.01.2004
Test 2	01.02.2004 - 10.02.2004	2040	01.02.2004 - 06.02.2004
Test 3	20.06.2004 - 30.06.2004	2340	20.06.2004 - 25.06.2004
Test 4	02.01.2005 - 10.01.2005	1752	02.01.2005 - 07.01.2005
Test 5	01.01.2007 - 10.01.2007	2302	01.01.2007 - 05.01.2007
Test 6	01.01.2009 - 09.01.2009	1992	01.01.2009 - 07.01.2009
Test 7	02.01.2011 - 10.01.2011	1752	02.01.2011 - 07.01.2011
Test 8	01.01.2014 - 10.01.2014	2016	01.01.2014 - 08.01.2014
Test 9	01.01.2019 - 10.01.2019	2040	01.01.2019 - 08.01.2019

3.2 Evaluation Method

The evaluation method consists of two parts: accumulated reward and waiting ratio.

The first evaluation method is the *accumulated reward* which is defined as total rewards per one episode. It means the performance of agent's Forex trading. Note that the accumulated reward is equivalent to total P/L per one episode due to the definition of Section 2.7.

The second evaluation method is the *waiting ratio* defined as Equation 3.3

$$waiting\ ratio = \frac{the\ number\ of\ waiting\ in\ one\ episode}{the\ number\ of\ total\ actions\ in\ one\ episode}. \tag{3.3}$$

As mentioned in Section 1.6, when a trader cannot be sure the direction of the exchange rate, he should wait. The waiting ratio shows whether RL realizes this strategy. The further the period of testing dataset is from the training period, the more the waiting ratio should increase.

3.3 Details

For details about source code and dataset, see the GitHub repository [12].

Chapter 4

Result

4.1 Accumulated Reward

Figure 4.1 shows the accumulated reward of each MA for each training episode. It suggests that the agent cannot learn how to make a profit because even the second half of the episodes show mostly negative rewards, as well as Table 4.1 shows. In addition, the number of MA seems to have little impact on the reward in the training.

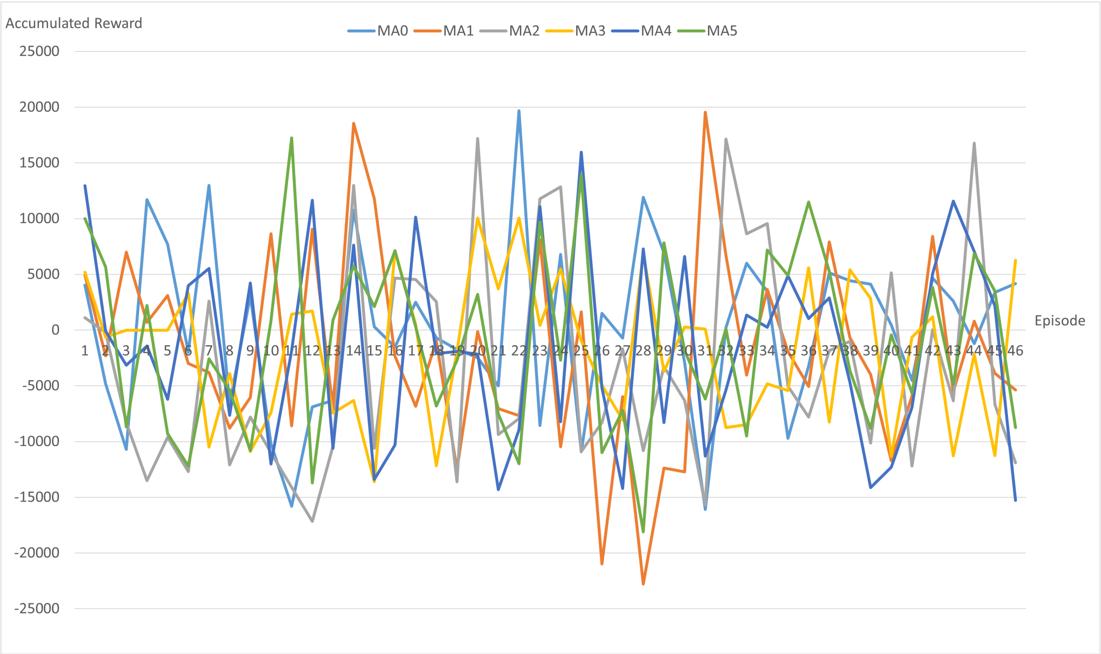

Figure 4.1: Accumulated reward in training for the number of moving average (MA)

Table 4.1: The average of accumulated reward in the last ten trainings for each MA

MA0	MA1	MA2	MA3	MA4	MA5
954	-2748	-2169	-2785	-2123	-1205

The same is true for Figure 4.2. Most rewards of the tests are negative, and the figure suggests that the number of MA could not improve the trading performance since each MA shows similar accumulated rewards. If the number of MA had some sort of impact on the performance, each MA would show the different accumulated reward.

As a result, it is concluded that MA may be invalid metrics for DQL of Forex trading.

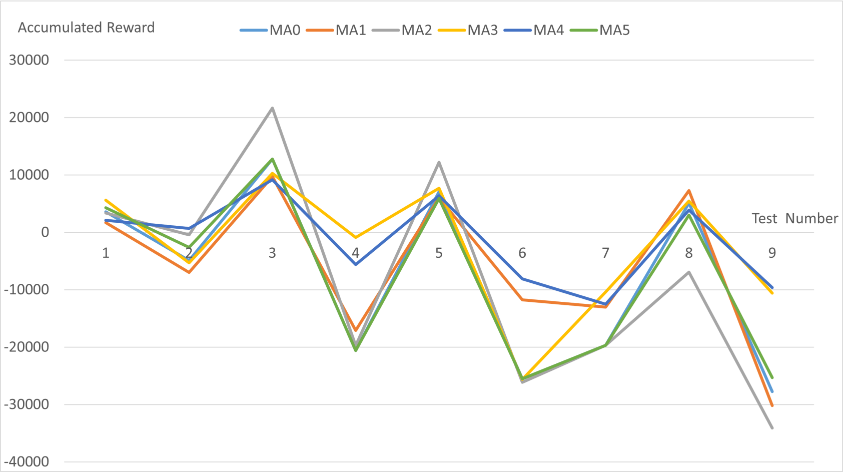

Figure 4.2: Accumulated reward in testing for the number of MA

4.2 Waiting Ratio

Figure 4.3 is the chart of the waiting ratio for each MA in the training episodes. It suggests that the waiting ratios of any number of MA tend to converge to the range of 70% to 75% as well as Figure 4.4 shows.

From Figure 4.5 to Figure 4.10, these figures indicate that the range of 70% to 75% is the boundary whether the loss absolutely occurs or not: the right sides of the range in the figures show that most data points are negative accumulated rewards.

RL is presumed to help to avoid losses in Forex trading. As explained in the previous section, it was difficult for the agent to get profit with MA metrics, therefore the agent seemingly focuses on avoiding losses.

Figure 4.11 and 4.12 are consistent with the expectation of Section 3.2: the further the period of testing dataset is from the training period, the more the waiting ratio increases. The waiting can prevent losses when the agent cannot expect the future exchange rate.

In conclusion, RL itself is considered to be useful for avoiding losses in Forex trading.

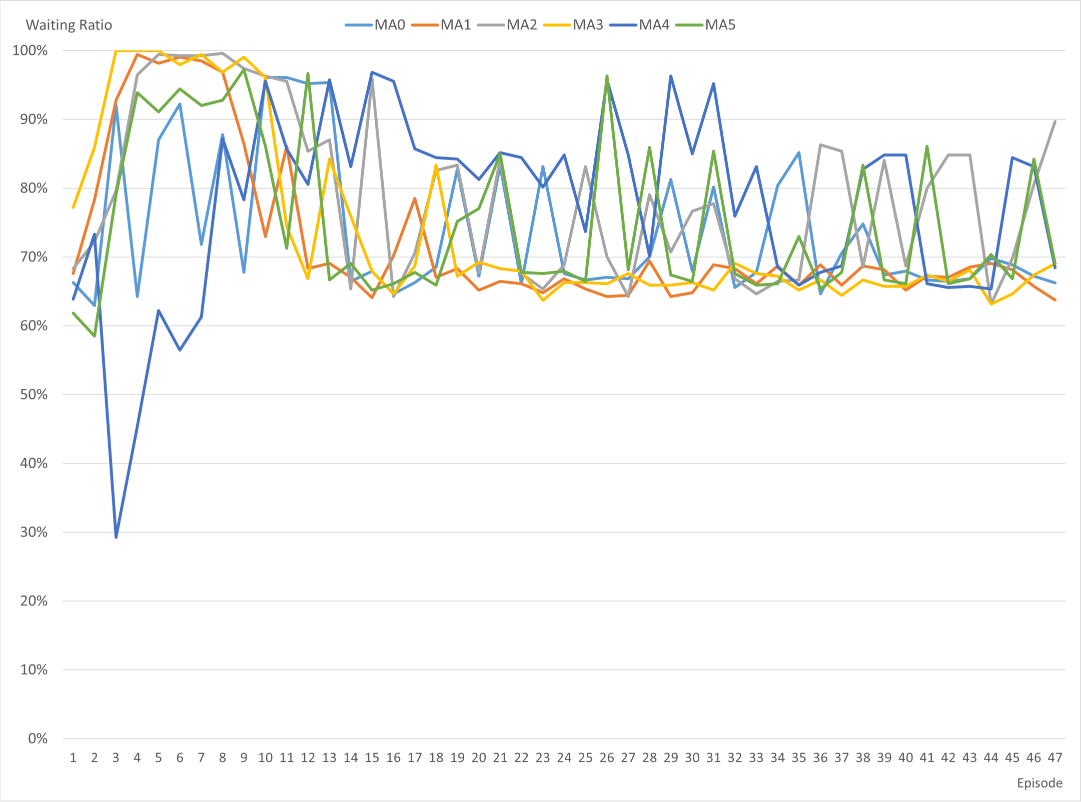

Figure 4.3: Waiting ratio in training for the number of MA

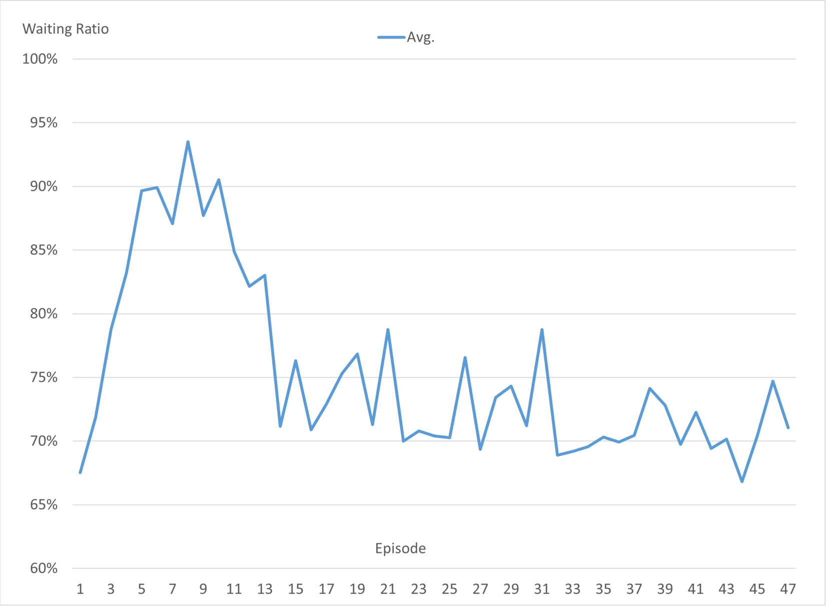

Figure 4.4: Average of waiting ratio in testing for the number of MA

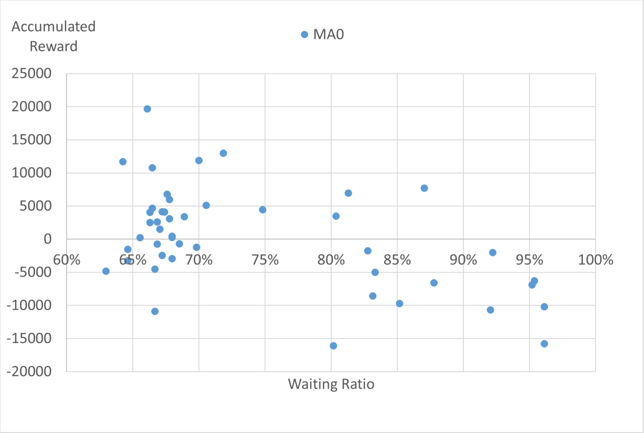

Figure 4.5: Scatter plot between waiting ratio and accumulated reward in training of MA0

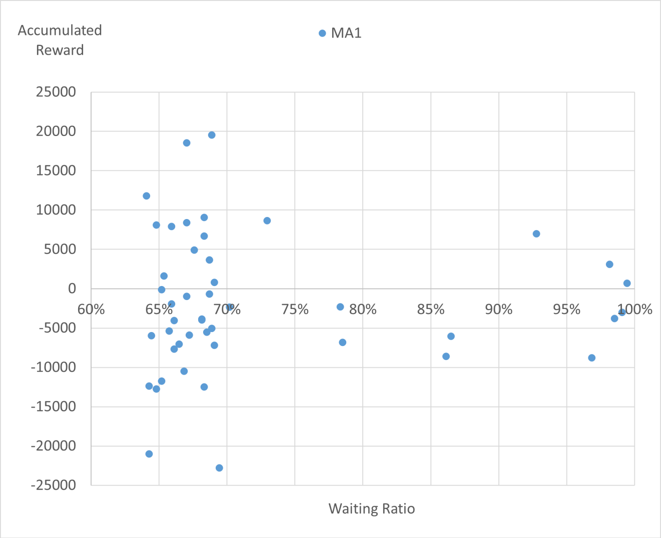

Figure 4.6: Scatter plot between waiting ratio and accumulated reward in training of MA1

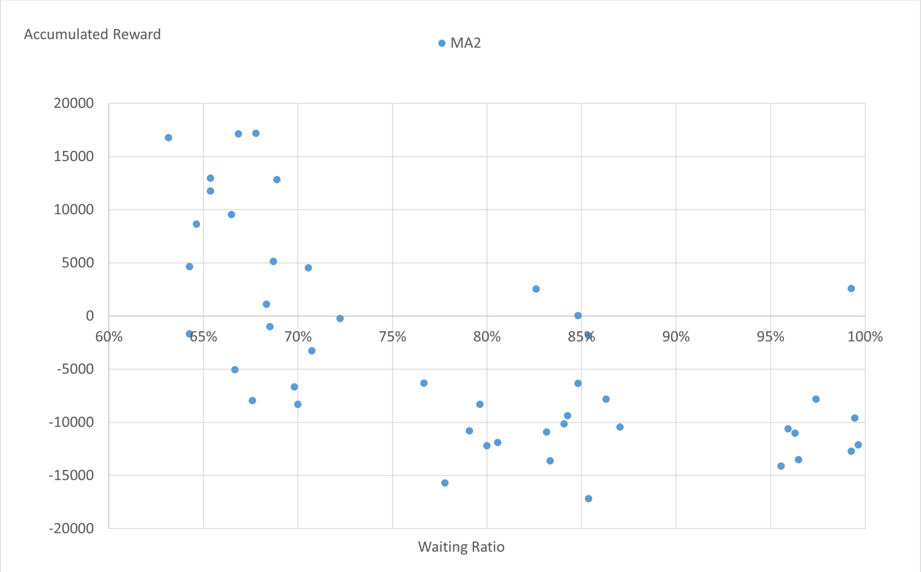

Figure 4.7: Scatter plot between waiting ratio and accumulated reward in training of MA2

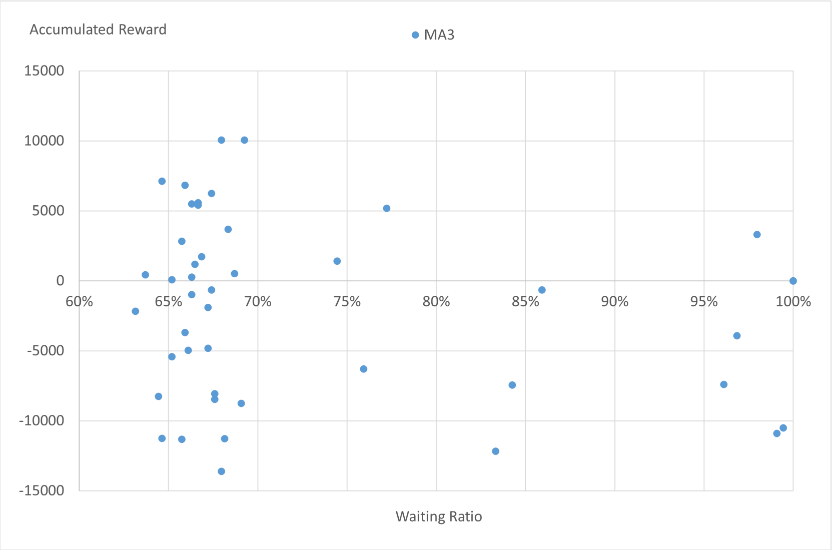

Figure 4.8: Scatter plot between waiting ratio and accumulated reward in training of MA3

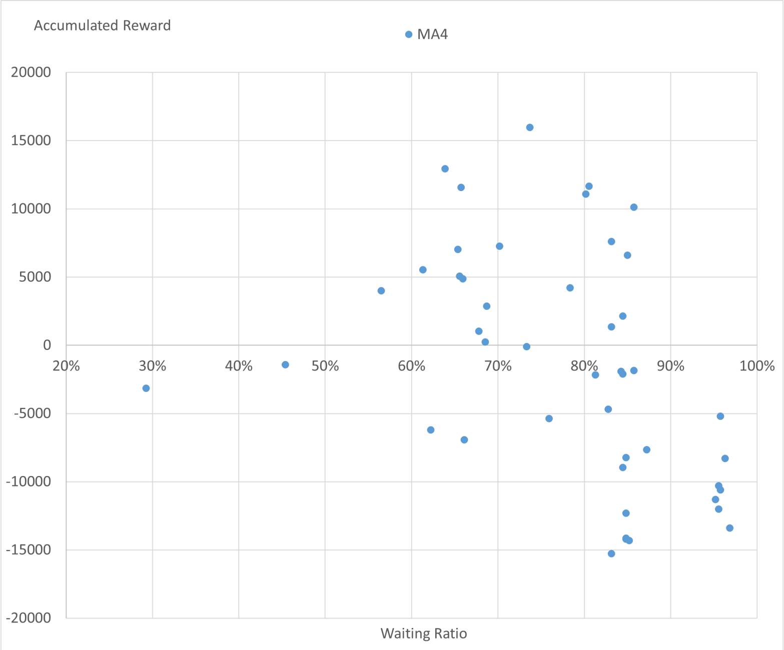

Figure 4.9: Scatter plot between waiting ratio and accumulated reward in training of MA4

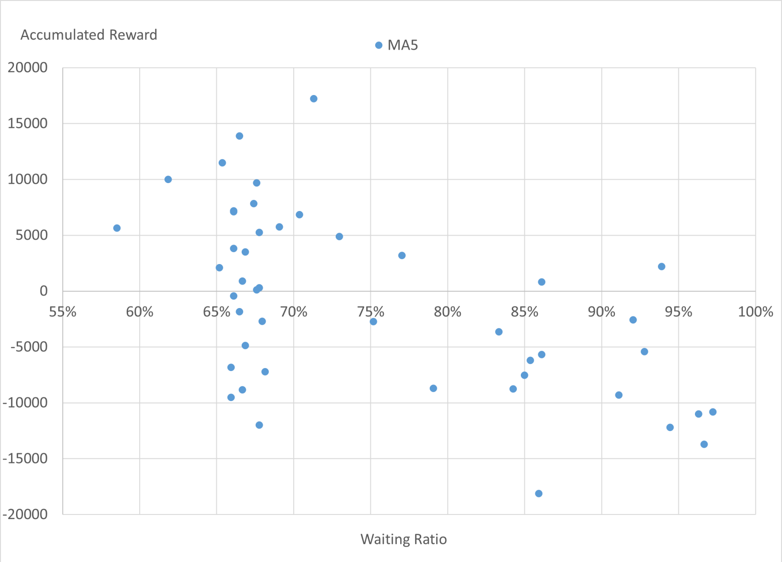

Figure 4.10: Scatter plot between waiting ratio and accumulated reward in training of MA5

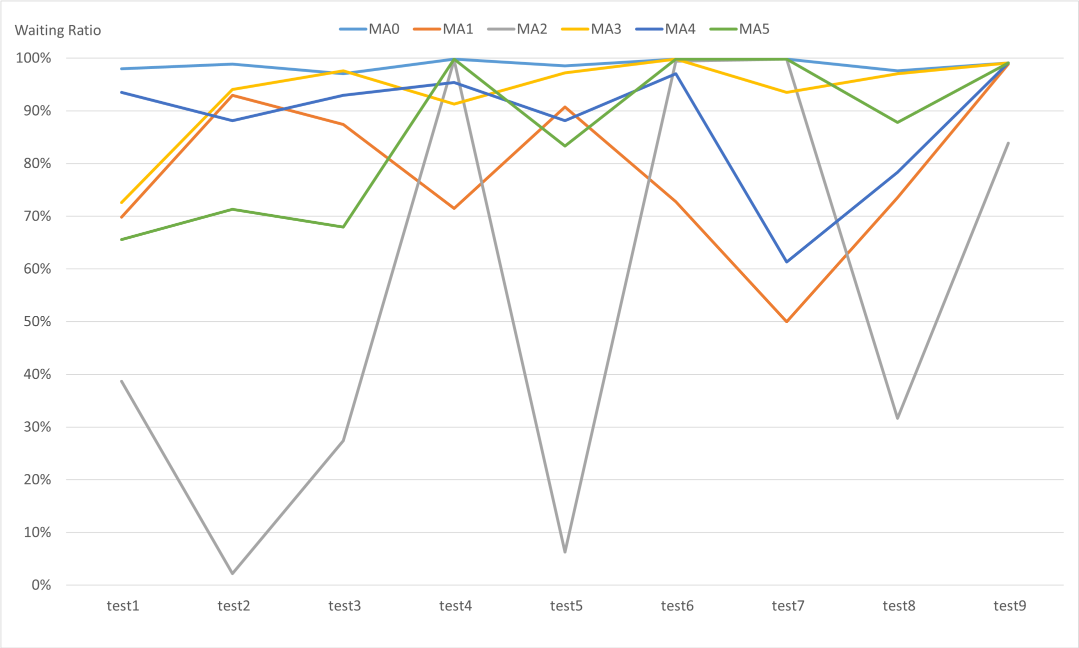

Figure 4.11: Waiting ratio in testing for the number of MA

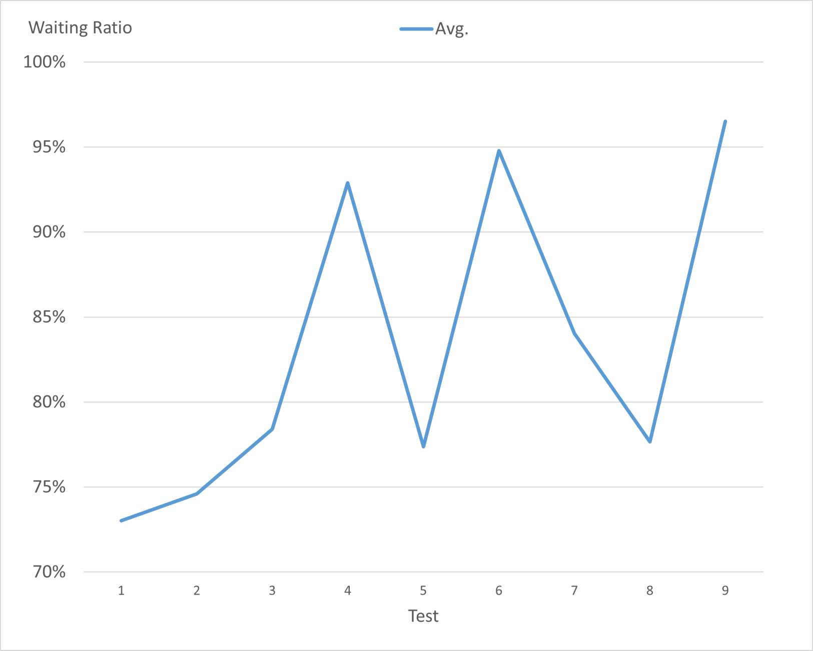

Figure 4.12: Average of waiting ratio in testing

Chapter 5

Conclusion

According to the result of the accumulation reward, the experiments could not prove the importance of multiple MAs. We concluded that MA may be invalid metrics for DQL of Forex trading. However, it is difficult to assert that MA is invalid metrics as state element because professional financial analysts still utilize it. Other causes of this research may prevent the performance. For example, hyperparameter such as step may be inappropriate or the way of utilizing MA may be too simple to perform for Forex trading.

On the other hand, the result of the waiting ratio suggested that RL itself can be useful to avoid losses.

The biggest problem was that the agent failed to learn to get profit in both training and testing. Unfortunately, my research method was not useful for an algorithmic trading system.

Further investigations are needed to make this DQN method practical. Firstly, we have to decide whether MA is invalid metrics for Forex trading with the experiments to vary hyperparameters, the way of utilizing MA, and the type of DQN. After that, if it turns out that MA is invalid, we have to consider other metrics. For example, oscillator, Fibonacci retracement, relative strength index (RSI), or Bollinger Band can be the candidate of it [13]. In addition, we may need to consider combining DQL with the price prediction model such as using convolutional neural network (CNN) [14].

Secondly, as Section 2.5 suggested, the agent with DQN is supposed to skip learning the position transition since the transition is deterministic. Therefore, we will have to find the way to realize it while DQN focuses on a stochastic environment.

Lastly, we have to verify whether RL actually avoids losses in the trading. Section 4.2 suggested the RL usefulness, but it was not enough to prove it. To validate the evidence, we must also identify which statistics need to be analyzed.

References

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[2] J. Zhang, Y. Liu, K. Zhou, G. Li, Z. Xiao, B. Cheng, J. Xing, Y. Wang, T. Cheng, L. Liu *et al.*, “An end-to-end automatic cloud database tuning system using deep reinforcement learning,” in *Proceedings of the 2019 International Conference on Management of Data*, 2019, pp. 415–432.

[3] 新見朋広, “本邦外国為替証拠金（FX）取引の最近の動向,” 日銀レビュー, no. 2016-J-9, pp. 1–7, 2016.

[4] 牧野貴樹, 澁谷長史, 白川真一, 浅田稔, 麻生英樹, 荒井幸代, 飯間等, 伊藤真, 大倉和博, 黒江康明 *et al.*, これからの強化学習. 森北出版, 2016.

[5] C. J. C. H. Watkins, “Learning from delayed rewards,” 1989.

[6] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing atari with deep reinforcement learning,” *arXiv preprint arXiv:1312.5602*, 2013.

[7] A. Bahrammirzaee, “A comparative survey of artificial intelligence applications in finance: artificial neural networks, expert system and hybrid intelligent systems,” *Neural Computing and Applications*, vol. 19, no. 8, pp. 1165–1195, 2010.

[8] “Moving Average Strategies for Forex Trading,” https://www.investopedia.com/ask/answers/122314/how-do-i-use-moving-average-ma-create-forex-trading-strategy.asp, accessed: 2021-2-20.

[9] “tf.keras.activations.relu,” https://www.tensorflow.org/api_docs/python/tf/keras/activations/relu, accessed: 2021-2-20.

[10] “tf.keras.activations.linear,” https://www.tensorflow.org/api_docs/python/tf/keras/activations/linear, accessed: 2021-2-20.

[11] “tf.keras.layers.Dense,” https://www.tensorflow.org/api_docs/python/tf/keras/layers/Dense, accessed: 2021-2-20.

[12] “Master Thesis Resources,” https://github.com/s1230038/masterThesis, accessed: 2021-2-20.

[13] “6 Types of Technical Analysis Every Forex Trader Should Learn,” https://www.valutrades.com/en/blog/6-types-of-technical-analysis-every-forex-trader-should-learn, accessed: 2021-2-20.

[14] A. Suchaimanacharoen, T. Kasetkasem, S. Marukatat, I. Kumazawa, and P. Chavalit, “Empowered PG in Forex Trading,” *17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, ECTI-CON 2020*, pp. 316–319, 2020.