การจัดสรรเงินลงทุนในพอร์ตการลงทุนด้วยวิธี Deep Deterministic Policy Gradients

Deep Deterministic Policy Gradients for Portfolio management

ณฐพงษ์ เมืองไพศาล[1] และ สมพร ปั่นโภชา[2]

[1] สาขาวิศวกรรมการเงิน คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยหอการค้าไทย, 1910531201011@live4.utcc.ac.th

[2] สาขาวิศวกรรมการเงิน คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยหอการค้าไทย, somporn_pun@utcc.ac.th

---

บทคัดย่อ

การศึกษาครั้งนี้มีวัตถุประสงค์เพื่อนำอัลกอริทึม Deep Deterministic Policy Gradients มาประยุกต์ใช้กับการหาสัดส่วนที่เหมาะสมสำหรับการลงทุนของพอร์ตโฟลิโอที่มีการลงทุนใน SET50 โดยทำการเปรียบเทียบผลกับพอร์ตโฟลิโอที่มีการลงทุนในหลักทรัพย์โดยเฉลี่ยเท่ากันทุกหลักทรัพย์ด้วยผลตอบแทนรายปี ความเสี่ยงของพอร์ตโฟลิโอ และ อัตราส่วน Sharpe โดยใช้ข้อมูลราคาเปิด ราคาปิด ราคาสูง และ ราคาต่ำ รายวันตั้งแต่วันที่ 4 มกราคม 2553 ถึงวันที่ 31 ธันวาคม 2563 ผลที่ได้พบว่า พอร์ตโฟลิโอที่ใช้ อัลกอริทึม Deep Deterministic Policy Gradients มีความสามารถสร้างผลลัพธ์ที่ดีกว่าในทุกด้าน เช่น Learning rate 0.001 ผลตอบแทนรายปีอยู่ที่ 0.0018 ความเสี่ยง 0.2363 Sharpe ratio 0.01 เปรียบเทียบกับพอร์ตโฟลิโอที่มีการลงทุนในหลักทรัพย์โดยเฉลี่ยเท่ากันทุกหลักทรัพย์ ผลตอบแทนรายปีอยู่ที่ -0.0278 ความเสี่ยง 0.2289 Sharpe ratio -0.12

คำสำคัญ: Deep Deterministic Policy Gradients, Machine learning, Portfolio management

ABSTRACT

The purpose of this study is to adapt Deep Deterministic Policy Gradients with Asset Allocation in SET50 Portfolio. Then measure annual return, standard deviation and Sharpe ratio of portfolio data from 4 January 2010 to 31 December 2020 compare with Equal weight portfolio that we use as benchmark. Base on this study results show Deep Deterministic Policy Gradient portfolio can outperform benchmark. At Learning rate 0.001 Annual return 0.0018 Volatility 0.2363 Sharpe ratio 0.01 compared to Equal weight portfolio Annual return -0.0278 Volatility 0.2289 Sharpe ratio -0.12

Keywords: Deep Deterministic Policy Gradients, Machine learning, Portfolio management

1. บทนำ

ในปัจจุบันบัญชีซื้อขายหลักทรัพย์ของประเทศไทยมีอยู่ประมาณ 3.32 ล้านบัญชี โดยมีอัตราการเพิ่มขึ้นประมาณ 200,000 ถึง 300,000 บัญชีต่อปี มีนักลงทุนรวม 1.45 ล้านราย จากในอดีตที่ผ่านมาการหาสัดส่วนการลงทุนที่เหมาะสมมีการศึกษาวิจัย ปรับปรุงและพัฒนามาอย่างต่อเนื่อง ส่งผลให้นักวิจัยสามารถสร้างแบบจำลองสำหรับการหาสัดส่วนการลงทุนที่เหมาะสมโดยการเพิ่มความซับซ้อนในการหาคำตอบที่ต้องการได้แม่นยำมากยิ่งขึ้น ในปัจจุบันมีการนำ Machine learning นั้นมาประยุกต์ใช้ในแก้ปัญหาในหลากหลายสาขา ซึ่งในงานวิจัยนี้ได้เลือกใช้วิธีการ Deep

Reinforcement Learning มาประยุกต์เพื่อหาคำตอบของการหาสัดส่วนที่เหมาะสมในการลงทุนซึ่งเป็นปัญหาหลักในการบริหารพอร์ตโฟลิโอ โดยใช้อัลกอริทึม Deep Reinforcement Learning มารวมกับการเรียนรู้เชิงลึก (Deep learning) ซึ่งเป็นรูปแบบหนึ่งของ Machine learning โดยใช้เครือข่ายประสาทเทียมเพื่อเปลี่ยนชุดของข้อมูลที่ป้อนให้กับเครือข่าย ให้เป็นผลลัพธ์จากการปรับน้ำหนักในแต่ละ Node ของเครือข่ายเพื่อเพิ่มความแม่นยำของผลลัพธ์ที่ได้จากการคาดการณ์ของเครือข่ายในแต่ละครั้ง เพื่อใช้จัดการกับปัญหาข้อมูลตั้งต้นที่มีความซับซ้อน

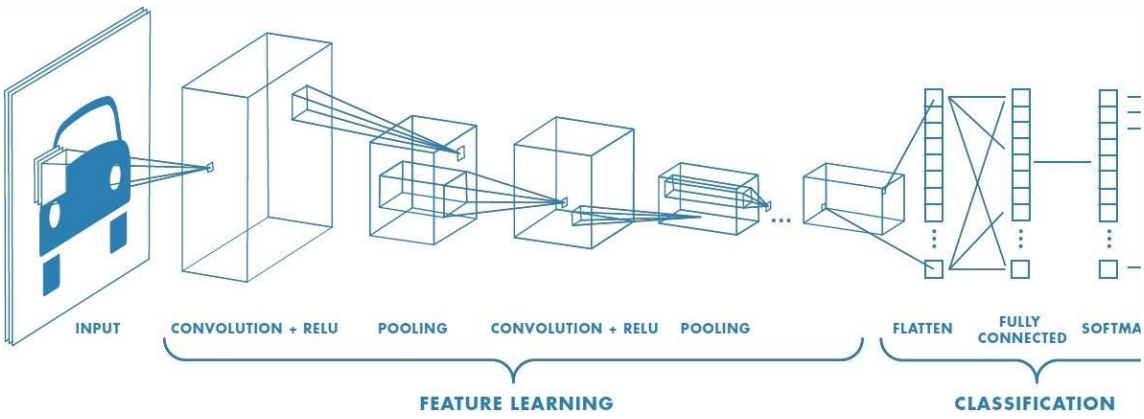
1.1 Convolutional neural network (CNN)

เป็นระบบโครงข่ายประสาทเทียมที่มีลักษณะคล้ายระบบโครงข่ายประสาทเทียมพื้นฐาน แต่มีลักษณะเด่นคือ CNN จะทำงานได้ดีกว่ากับ Input ที่เป็นประเภทรูปภาพ เนื่องจากความสามารถในการสกัดเอา Feature หรือลักษณะเด่นต่างๆออกมาจากรูปภาพเพื่อใช้ในการเป็น Input ให้แก่ CNN โดยภายในระบบ CNN จะประกอบไปด้วย 3 Layer ได้แก่ Convolutional layer, Pooling layer และ Fully connected layer

1.1.1 Convolutional layer เป็น Layer ที่ทำหน้าที่ในการสกัด Feature ออกมาจากรูปภาพที่ใช้เป็น input เพื่อนำมาสร้างเป็น Feature map โดยในการสกัด Feature นั้นทำโดยการแบ่งรูปภาพออกเป็นส่วนๆ แต่ละส่วนจะถูกเรียกว่า Cell จากนั้นนำแต่ละ Cell มาผ่านกระบวนการ Convolution กับ Filter เพื่อให้ได้เป็น Feature ออกมาโดย Filter ที่ใช้นั้นขึ้นอยู่กับความต้องการ ของผู้ใช้ว่าต้อง Feature ใดจากรูปภาพ

1.1.2 Pooling layer เป็น Layer ที่ทำหน้าที่ในการปรับขนาดและปริมาณของข้อมูลตัวอย่าง (Sample) ให้ลดลงก่อนนำส่งเข้าสู่ Layer ถัดไปเพื่อให้สามารถวิเคราะห์และเก็บรายละเอียดของภาพได้อย่างครบถ้วนโดยที่ไม่สูญเสียข้อมูล ยิ่งไปกว่านั้นกระบวนการนี้ยังช่วยลดโอกาสเกิดเหตุการณ์ Overfitting ได้อีกด้วย ในการ Pooling นั้นจะมีกระบวนการที่คล้ายกับกระบวนการสร้าง Feature maps คือการแบ่ง Feature map ออกเป็น Cell จากนั้นนำ Cell ไปผ่านกระบวนการ Pooling โดยการทำ Convolution กับ Filter อีกครั้งเหมือนกับ Convolutional layer

1.1.3 Fully connect layer ประกอบด้วยระบบ Multilayer perceptron (MLP) ในการประมวลผลข้อมูลที่ได้มาจาก 2 layer ก่อนหน้านี้เพื่อสังเคราะห์และทำการแยกแยะรูปภาพออกเป็นหมวดหมู่

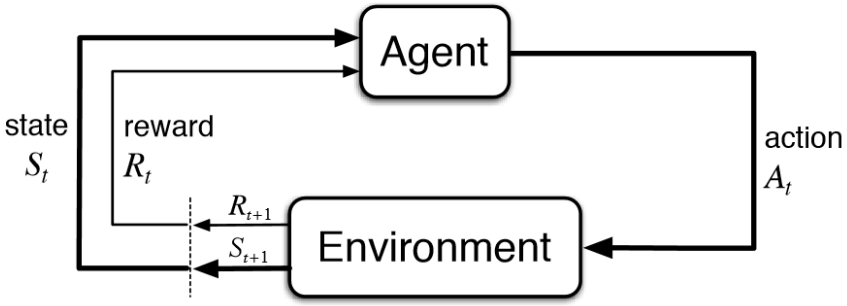


รูปที่1.1 รูปภาพแสดงการทางานของ CNN

(สืบค้นจาก https://www.mathworks.com/solutions/deep-learning/convolutional-neural- network.html)

1.2 การเรียนรู้แบบเสริมกำลัง (Reinforcement learning)

การเรียนรู้แบบเสริมกำลังเป็นการเรียนรู้แบบมีผู้สอนและไม่มีผู้สอน การเรียนรู้แบบเสริมกำลังมีลักษณะเด่นคือ ผู้เล่น (Agent) และสิ่งแวดล้อม(Environment) ซึ่งผู้เล่นจะทำการเลือก การกระทำ (Action) โดยที่การกระทำใดๆจะได้รับผลจากสิ่งแวดล้อม และสิ่งแวดล้อมจะให้ค่า สถานะ (State) และรางวัล (Reward) ซึ่งเป็นสิ่งบ่งบอกว่าการกระทำนั้นส่งผลที่ดีหรือไม่เป็นวงจรอย่างต่อเนื่องซึ่งแสดงในรูปที่ 1.1 โดยที่จุดประสงค์ของการเรียนรู้ของการเรียนรู้แบบเสริมกำลังนั้น คือการหาค่ารางวัลที่สูงที่สุด (Maximize reward)

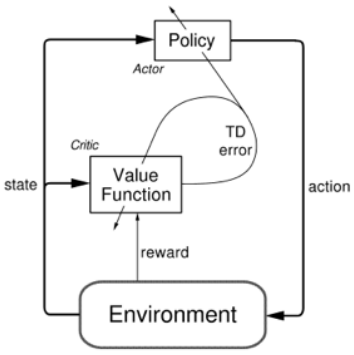


รูปที่ 1.2 วงจรการทำงานของ Reinforcement learning

ที่มา: https://towardsdatascience.com/reinforcement-learning-101-e24b50e1d292

1.3 Actor-Critic

Actor-Critic เป็นขั้นตอนวิธีการที่ใช้ในการเรียนรู้ ซึ่งสามารถแยกออกเป็น 2 ส่วน คือฟังก์ชันมูลค่าที่เรียกว่า Critic และฟังก์ชันนโยบายที่เรียกว่า Actor ดังแสดงในรูปที่ 1.3



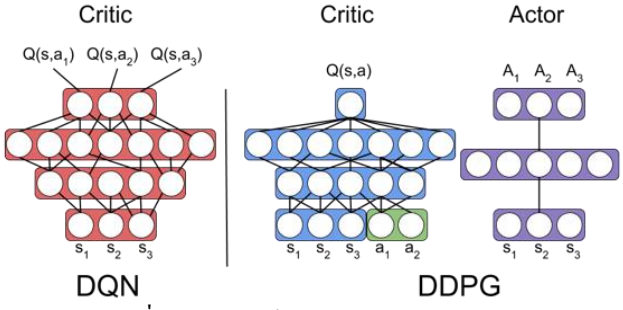
รูปที่ 1.3 ขั้นตอนวิธี Actor-Critic

ที่มา: https://towardsdatascience.com/reinforcement-learning-w-keras-openai-actor-critic-models-f084612cfd69

จากรูปที่ 1.3 แสดงให้เห็นว่าเมื่อ Actor ได้รับสถานะปัจจุบัน (state) จากสิ่งแวดล้อม จะให้ค่าที่เรียกว่า การกระทำ (action) ส่วน Critic นั้นจะใช้ค่าความคาดเคลื่อนจากการคาดการณ์รางวัล (reward) ที่ได้จากการกระทำของ Actor มาทำการปรับนโยบายของ Actor เพื่อให้ Actor สามารถเพิ่มรางวัลจากการทำตามนโยบายที่กำหนด

1.4 Deep Deterministic Policy Gradient (DDPG)

Deep Deterministic Policy Gradient (DDPG) เป็นหนึ่งในแขนงของ Reinforcement learning technique ที่มีการผสมผสานระหว่าง Q-learning และ Policy gradients โดยมี Actor network และ Critic network เป็นส่วนประกอบหลัก ดังแสดงในรูป 1.4



รูปที่ 1.4 โครงสร้างของอัลกอริทึม DDPG

ที่มา: https://medium.com/intro-to-artificial-intelligence/deep-deterministic-policy-gradient-ddpg-an-off-policy-reinforcement-learning-algorithm-38ca8698131b

จากรูปที่ 1.4 แสดงถึงความแตกต่างของอัลกอริทึมระหว่าง DQN และ DDPG โดยที่ DQN นั้นจะให้ค่า Q (s,a) ในแต่ละเอาท์พุต node ส่วน DDPG นั้นจะใช้ Actor network ในการหาเอาท์พุตค่าการกระทำ(action) และใช้ Critic network ในการหาค่า Q(s,a) ซึ่งเป็นค่าประมาณของ action ที่ได้จาก state

Algorithm 1 DDPG algorithm

Randomly initialize critic network $Q(s, a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ.
Initialize target network Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$
Initialize replay buffer R
for episode = 1, M **do**
  Initialize a random process $\mathcal{N}$ for action exploration
  Receive initial observation state s_1
  for t = 1, T **do**
    Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$ according to the current policy and exploration noise
    Execute action a_t and observe reward r_t and observe new state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in R
    Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from R
    Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
    Update critic by minimizing the loss: $L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
    Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_i}$$

    Update the target networks:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$
$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

  end for
end for

รูปที่ 1.5 Pseudocode ของอัลกอริทึม DDPG

ที่มา: https://towardsdatascience.com/deep-deterministic-policy-gradients-explained-2d94655a9b7b

จากรูปที่ 1.5 แสดงให้เห็นถึงอัลกอริทึมของ DDPG ในรูปแบบของ Pseudocode เพื่ออธิบายภาพรวมของการทำงานของอัลกอริทึมดังกล่าว

2. วัตถุประสงค์การวิจัย

1. เพื่อศึกษาอัลกอริทึม Deep Deterministic Policy Gradients และนำมาประยุกต์ใช้กับการหาสัดส่วนที่เหมาะสมสำหรับการลงทุนของพอร์ตโฟลิโอที่มีการลงทุนใน SET50
2. เพื่อใช้ Machine Learning เป็นเครื่องมือช่วยในการตัดสินใจสำหรับนักลงทุนโดยพิจารณาจากความเสี่ยงและผลตอบแทนที่คาดหวัง

3. ระเบียบวิธีวิจัย

ในการศึกษาวิจัยการบริหารพอร์ตโฟลิโอด้วยการเรียนรู้แบบเสริมกำลัง DDPG งานวิจัยนี้ได้ทำการสร้างแบบจำลองพอร์ตโฟลิโอ รวมถึงทำการสร้างสิ่งแวดล้อมด้วยจุดประสงค์เพื่อให้แบบจำลองมีการเรียนรู้ และทำการทดสอบพารามิเตอร์เพื่อหาว่า พารามิเตอร์ตัวใดที่ส่งผลต่อการแบบจำลองในการจัดสรรการลงทุน โดยการเปรียบเทียบกับพอร์ตโฟลิโอที่มีการลงทุนแบบกระจายในทุกหลักทรัพย์ในอัตราส่วนที่เท่ากันเพื่อทดสอบประสิทธิภาพของแบบจำลอง ซึ่งมีขั้นตอนในการศึกษาดังนี้

3.1 การเก็บรวบรวมข้อมูล

การเก็บรวบรวมข้อมูลจาก Setsmart โดยใช้ข้อมูลหลักทรัพย์ทั้งสิ้น 30 บริษัท ด้วยการนำข้อมูลตั้งแต่วันที่ 4 มกราคม 2553 ถึง 30 ธันวาคม 2563 มาใช้เป็นข้อมูลตั้งต้น มีจำนวนทั้งสิ้น 2,683 วัน โดยข้อมูลที่ถูกนำมาใช้ได้แก่

3.1.1 วัน (Date)

3.1.2 ราคาเปิด (Open price)

3.1.3 ราคาปิด (Close price)

3.1.4 ราคาสูงสุดในวัน (High price)

3.1.5 ราคาต่ำสุดในวัน (Low price)

โดยแบ่งชุดข้อมูลออกเป็นชุดข้อมูลสำหรับเรียนรู้ร้อยละ 60 ของข้อมูลทั้งหมด คิดเป็นจำนวน 1,608 วัน ซึ่งทำการแบ่งเป็นข้อมูลที่ใช้สำหรับตรวจสอบจำนวน 536 วัน และแบ่งเป็นข้อมูลสำหรับทดสอบจำนวน 536 วัน

ในการเตรียมข้อมูลตั้งต้นสำหรับทดสอบกับแบบจำลองในครั้งนี้ได้ทำการหาผลลัพธ์เพื่อปรับข้อมูลนำเข้าตั้งต้น ด้วยการนำข้อมูลตั้งต้นในแต่ละวันมาทำการหารด้วยราคาปิดแต่ละวัน

3.2 การสร้างแบบจำลอง

3.2.1 พอร์ตโฟลิโอที่ใช้ในการทดลอง

พอร์ตโฟลิโอนั้นประกอบด้วยหลักทรัพย์ที่มีความเสี่ยง m ตัวและหลักทรัพย์ไม่มีความเสี่ยงหนึ่งหลักทรัพย์ m = 30 โดยให้น้ำหนักของแต่ละหลักทรัพย์ที่มีความเสี่ยงแสดงด้วย $[w_1, w_2, \dots, w_m]$ และหลักทรัพย์ไม่มีความเสี่ยง $[w_c]$ ดังนั้นน้ำหนักของพอร์ตโฟลิโอ ณ เวลา t ใดๆ จะอยู่ในรูปสมการที่ 3.1

$$w_t = [w_c, w_1, w_2, \dots, w_m] \quad (3.1)$$

ณ เวลา t ผลตอบแทนจากหลักทรัพย์ต่อวันจะอยู่ในรูป y_t โดยประกอบด้วย อัตราดอกเบี้ยไร้ความเสี่ยงและผลตอบแทนของหลักทรัพย์ i เมื่อเวลา t ใช้อัตราส่วนของราคาเปิด ณ เวลา t กับเวลา t-1 อยู่ในรูป สมการที่ 3.2

โดยที่ค่า interest rate เป็นค่าคงที่

$$y_t = \left[\text{interest rate}, \frac{v_{1,t}^{(Open)}}{v_{1,t-1}^{(Open)}} - 1, \ldots, \frac{v_{m,t}^{(Open)}}{v_{m,t-1}^{(Open)}} - 1\right] \quad (3.2)$$

transaction cost คือขนาดของน้ำหนักที่เปลี่ยนไปของแต่ละหลักทรัพย์ ณ เวลา t คูณกับมูลค่า พอร์ตโฟลิโอ ณ เวลา t - 1 คูณกับ transaction fee โดยให้ค่า transaction fee เป็นค่าคงที่ ดังสมการที่ 3.3

$$\text{transaction cost} = p_{t-1}.||(w_{t-1} - w_t)||.\text{transaction fee} \quad (3.3)$$

มูลค่าของพอร์ตโฟลิโอที่เวลา t (p_t) สามารถหาได้จาก ผลคูณของมูลค่าของพอร์ตโฟลิโอเวลา t - 1 (p_{t-1}) กับน้ำหนักของหลักทรัพย์ในพอร์ตโฟลิโอ ณ เวลา t ลบด้วย transaction cost และปรับมูลค่าพอร์ตโฟลิโอด้วย ผลตอบแทนต่อวัน ดัง สมการที่ 3.4

$$p_t = (p_{t-1}.w_t - \text{transaction cost}).(y_t + 1) \quad (3.4)$$

น้ำหนักของหลักทรัพย์ หลังปรับพอร์ตโฟลิโอด้วยค่า transaction cost และผลตอบแทนรายวัน$(w'_{i,t})$ จะเท่ากับมูลค่าของหลักทรัพย์แต่ละตัวหารด้วยมูลค่าของพอร์ตโฟลิโอ ดัง สมการที่ 3.5

$$w'_{i,t} = p_{i,t}/p_t \quad (3.5)$$

3.2.2 พอร์ตโฟลิโอที่มีการลงทุนแบบกระจายในทุกหลักทรัพย์ในอัตราส่วนที่เท่ากัน

จากสมการที่ 1 เมื่อหลักทรัพย์ทั้งหมดมีหนึ่งหลักทรัพย์ไม่มีความเสี่ยงและ 30 หลักทรัพย์ที่ทีความเสี่ยง

$$w_t = \left[\frac{1}{m+1}, \frac{1}{m+1}, \frac{1}{m+1}, \ldots, \frac{1}{m+1}\right] \quad (3.6)$$

3.3 นำแบบจำลอง DDPG ไปทดสอบกับข้อมูลโดยทำการฝึกแบบจำลอง จำนวน 100 episode เพื่อให้แบบจำลองมีการเรียนรู้ข้อมูล หลังจากนั้นจะนำแบบจำลองที่ได้ไปทำการทดสอบกับข้อมูลที่แบบจำลองไม่รู้จัก จำนวน 536 วัน โดยได้ทำการออกแบบโครงสร้างของโมเดลตามตารางที่ 3.1 และ 3.2

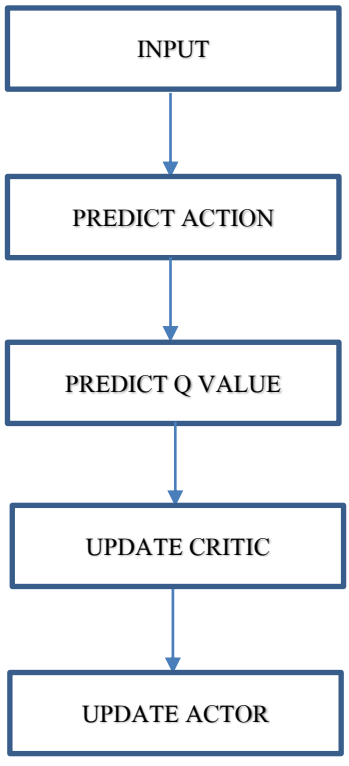
ตารางที่ 3.1 โครงสร้าง Actor ที่นำมาใช้

Layer	Shape	input	Activation
Input	[Batch size,L,M,N]		
CONV2D 1	[Batch size,L,M,2]	input	TANH
CONV2D 2	[Batch size,1,M,20]	CONV2D 1	TANH
CONCATENATE	[Batch size,1,M+1,1]	CONV2D 2,W_{t-1}	TANH
CONV2D 3	[Batch size,1,M+1,1]	CONV2D 2	TANH
FLATEN		CONV2D 3	

Layer	Shape	input	Activation
LAYER NORM		FLATEN	
OUTPUT	[Batch size,M+1]	LAYER NORM	SOFTMAX

ตารางที่ 3.2 โครงสร้าง Critic ที่นำมาใช้

Layer	Shape	input	Activation
Input	[Batch size,L,M,N]		
CONV2D 1	[Batch size,L,M,2]	input	TANH
CONV2D 2	[Batch size,1,M,20]	CONV2D 1	TANH
CONCATENATE	[Batch size,1,M+1,1]	CONV2D 2,W_{t-1}	TANH
CONV2D 3	[Batch size,1,M+1,1]	CONV2D 2	TANH
FLATEN		CONV2D 3	
Dense1	[Batch size,M+1]	FLATEN	LINEAR
OUTPUT	[Batch size,M+1]	FLATEN	LINEAR



รูปที่ 3.1 Flow chart การทำงานของ Model

จากรูปที่ 3.1 แสดงการทำงานของโมเดลในรูปแบบของ Flow chart ตั้งแต่การรับข้อมูลเข้าโมเดลเพื่อพยากรณ์การกระทำที่จะเกิดขึ้นเพื่อทำการปรับปรุงน้ำหนักของเครือข่ายของ Actor และ Critic

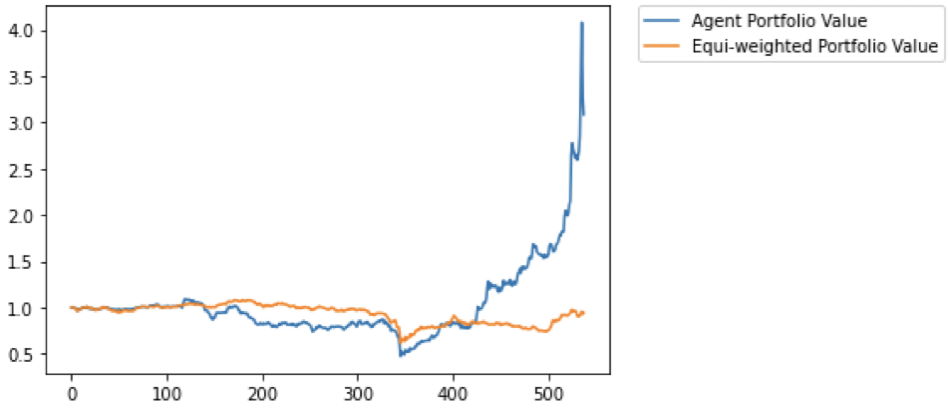
3.4 นำผลลัพธ์ที่ได้จากแบบจำลอง DDPG มาเปรียบเทียบกับพอร์ตโฟลิโอที่มีการลงทุนแบบกระจายในทุกหลักทรัพย์ในอัตราส่วนที่เท่ากัน (Benchmark) เพื่อวัดประสิทธิภาพของแบบจำลอง

4. ผลการวิจัย

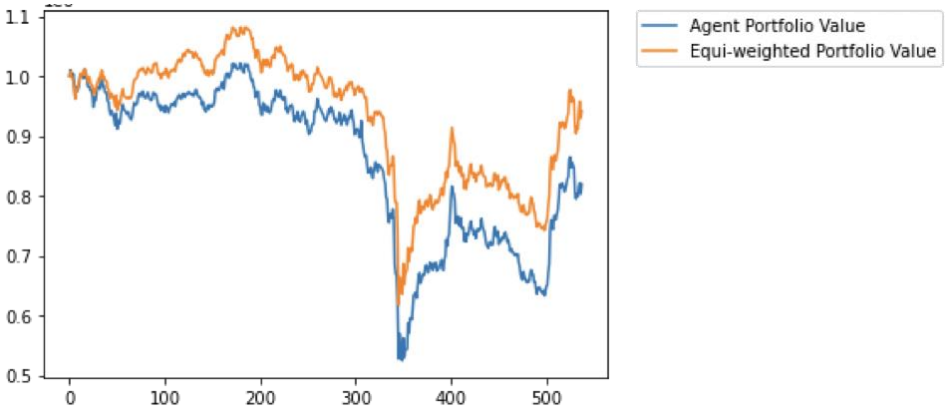
แบบจำลองที่มีการใช้โครงข่ายประสาทเทียมประเภท Convolution จากงานวิจัยอื่นที่ได้ทำการศึกษานั้นมีการนำแบบจำลอง DDPG ไปใช้กับการจัดการพอร์ตโฟลิโอที่มีการลงทุนใน Cryptocurrency Chinese stock market และ US stock market จากการศึกษาพบว่าให้ผลที่น่าพอใจ แต่ไม่เหมาะสมกับการนำมาใช้ในกรณีนี้ เนื่องจากเมื่อผ่านโครงข่ายประสาทเทียมชั้นที่ 3 เอาท์พุตที่ได้มีค่าที่น้อยมากรวมถึงการใช้ ฟังก์ชันกระตุ้นแบบ Relu นั้นทำให้ค่าที่ได้มีความแตกต่างกันอย่างไม่มีนัยสำคัญ ซึ่งก่อให้เกิดปัญหาเมื่อชั้นสุดท้ายของโครงข่ายใช้ฟังก์ชันกระตุ้นแบบ Softmax เอาท์พุตจากโครงข่ายจึงไม่มีการเปลี่ยนแปลงอย่างมีนัยสำคัญ นั่นก็คือ แม้ว่าอินพุตจะมีเปลี่ยนแปลงรูปแบบไปอย่างไรก็ไม่ทำให้น้ำหนักมีการเปลี่ยนแปลงตามทำให้แบบจำลองไม่สามารถเปลี่ยนสัดส่วนการลงทุนตามเวลาได้ ดังนั้นในงานวิจัยนี้ได้เสนอทางแก้ไขด้วยการเพิ่มชั้นของโครงข่ายประสาทเทียมก่อนหน้าชั้นสุดท้ายเพื่อทำการปรับค่ามาตรฐานของเอาท์พุตชั้นที่ 3 พร้อมทั้งเปลี่ยนฟังก์ชันกระตุ้นของโครงข่ายเพื่อให้ความสามารถในการจัดลำดับของ Convolution สามารถใช้งานได้ หลังจากทำการปรับปรุงโครงสร้างของโครงข่ายประสาทเทียมแล้วนั้นพบว่าแบบจำลองสามารถที่จะเรียนรู้และทำการปรับเปลี่ยนสัดส่วนการลงทุนตามเวลาได้

จากการศึกษาประสิทธิภาพการจัดสรรเงินลงทุนของแบบจำลอง Deep Deterministic Policy Gradient โดยใช้โครงข่ายประสาทเทียมประเภท Convolution ด้วยข้อมูลราคาเปิด ราคาปิด ราคาสูง และราคาต่ำ ของหลักทรัพย์ที่มีการซื้อขายใน SET50 ย้อนหลังเป็นเวลาทั้งสิ้น 10 ปี ตั้งแต่วันที่ 4 มกราคม 2553 ถึงวันที่ 31 ธันวาคม 2563 รวมทั้งสิ้น 2,683 วันซึ่งได้ทำการแบ่งข้อมูลออกเป็น 3 ชุดได้แก่ 1,608 วันสำหรับสร้างแบบจำลอง 536 วันสำหรับสอบทานแบบจำลอง และอีก 536วัน สำหรับทดสอบประสิทธิภาพของแบบจำลองโดยใช้พอร์ตโฟลิโอที่มีการลงทุนทุกหลักทรัพย์ในน้ำหนักที่เท่ากันเป็นตัวเปรียบเทียบพบว่า ถ้าใช้จำนวนวันของการเทรด 538 วันมูลค่าพอร์ตโฟลิโอจาก Machine learning

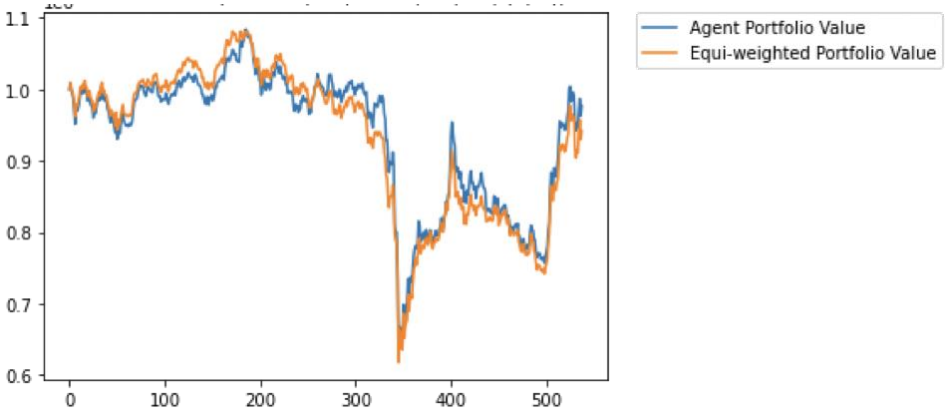
จากการที่ได้ทำการทดลองปรับโมเดลโดยกำหนดให้ Learning rate Actor = 0.005, 0.003 และ 0.001 และให้ Critic เป็นค่าคงที่ ผลการทดลองที่ได้ดังแสดงตามรูปด้านล่าง



รูปที่ 4.1 ผลการทดลองที่ **Learning rate Actor = 0.005 Critic = 0.001**



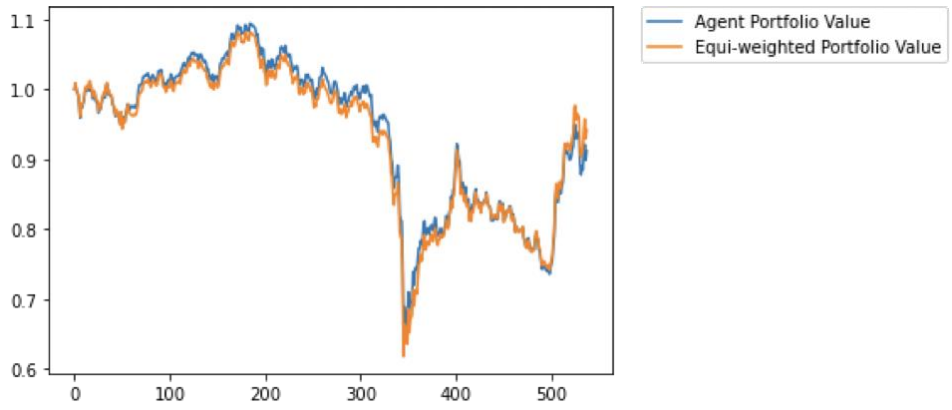
รูปที่ 4.2 ผลการทดลองที่ **Learning rate Actor = 0.003 Critic = 0.001**



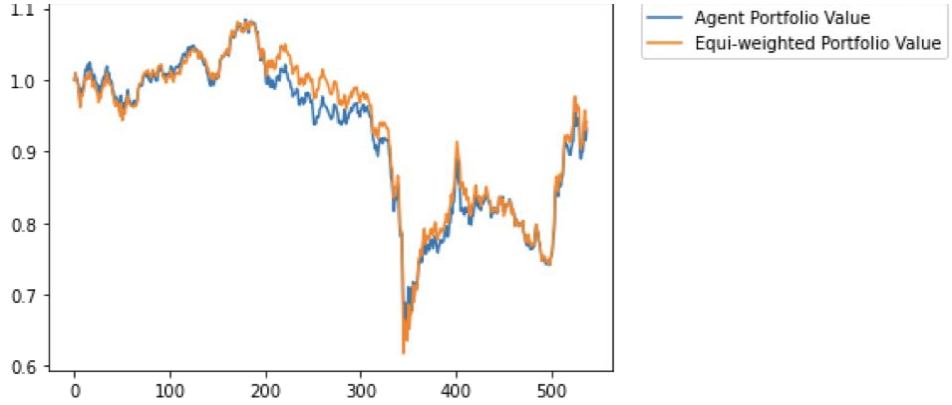
รูปที่ 4.3 ผลการทดลองที่ **Learning rate Actor = 0.001 Critic = 0.001**

และจากการทดสอบแบบจำลองพบว่า ที่ Learning rate Actor = 0.001 Critic = 0.001 ตามรูปที่ 4.3 ให้ผลที่น่าพอใจเนื่องจากพอร์ตที่ทำการจัดด้วยแบบจำลองให้ผลตอบแทนที่ดีกว่าพอร์ตที่ทำการลงทุนเท่ากันตลอดช่วงเวลา ดังนั้น จึงทำการทดสอบเพิ่มเติมด้วยการเปลี่ยนชนิดของข้อมูลอินพุตทั้งหมด 4 ชุด ได้แก่ ชุดที่1 ราคาเปิด

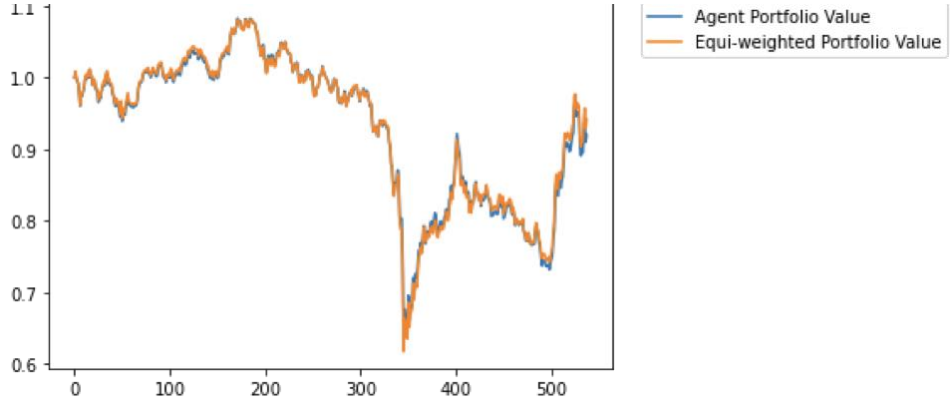
ราคาปิด ราคาสูง ราคาต่ำ, ชุดที่ 2 ราคาเปิด ราคาสูง, ชุดที่ 3 ราคาปิด ราคาสูง และชุดที่ 4 ราคาสูง ที่ค่า Learning rate 0.001 ซึ่งได้ผลการทดสอบดังรูปที่ 4.4, 4.5 และ 4.6 จากนั้นได้ทำการสรุปผลดังตารางที่ 4.1



รูปที่ 4.4 ผลการทดลองที่ **Learning rate Actor = 0.001 Critic = 0.001 ตัวแปรราคาเปิด ราคาปิด**



รูปที่ 4.5 ผลการทดลองที่ **Learning rate Actor = 0.001 Critic = 0.001 ตัวแปรราคาเปิด ราคาสูง**



รูปที่ 4.6 ผลการทดลองที่ **Learning rate Actor = 0.001 Critic = 0.001 ตัวแปรราคาเปิด**

สมการที่ 4.1 ใช้ในการคำนวนหา Annual return ของ พอร์ตโฟลิโอ

$$R_p = \left(\frac{P_t}{P_0}\right)^{total\ day/252} - 1 \qquad (4.1)$$

สมการที่ 4.2 ใช้ในการคำนวนหา Volatility ของ พอร์ตโฟลิโอ

$$\sigma_p = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} w_i w_j \sigma_{ij}} \quad (4.2)$$

สมการที่ 4.3 ใช้ในการคำนวนหา Sharpe ration ของ พอร์ตโฟลิโอ

$$S_p = \frac{R_p}{\sigma_p} \quad (4.3)$$

ตารางที่ 4.1 สรุปผลการทดลอง

	Portfolio Value	$E(R_p)$	σ_p	S_p
UCRP	941670	-0.0278	0.2289	-0.12
LR 0.005	2775708	0.6132	0.4881	1.26
LR 0.003	864416	-0.0660	0.2530	-0.26
LR 0.001	1003861	0.0018	0.2363	0.01
LR 0.001 ตัวแปรราคาเปิด ราคาสูง	955007	-0.0213	0.2169	-0.10
LR 0.001 ราคาเปิด ราคาปิด	949693	-0.0239	0.2379	-0.10
LR 0.001 ราคาเปิด	965853	-0.0161	0.2369	-0.07

จากตารางที่ 4.1 ผลจากการจัดสรรเงินลงทุนผ่านแบบจำลองจากพารามิเตอร์ที่กำหนดด้วยข้อมูลจำนวน 536 วัน พบว่าแบบจำลองให้ผลลัพธ์มูลค่าวันสุดท้ายของพอร์ตโฟลิโอที่ 2775708.0, 864416.0, 1003861.0 ที่ Learning rate 0.005 ,0.003 ,0.001 ตามลำดับ เมื่อเปรียบเทียบกับพอร์ตโฟลิโอที่มีการลงทุนในทุกหลักทรัพย์เท่ากันตลอดช่วงเวลาพบว่ามูลค่าสุดท้ายอยู่ที่ 941670 Sharpe ratio -0.12 โดยที่แบบจำลองมีความสามารถในการจัดสรรเงินลงทุนในระดับที่ใกล้เคียงกับตัวเปรียบเทียบโดยมีค่า Sharpe ratioที่ 1.26, -0.26, 0.01 ตามลำดับ

ในการทดสอบเปลี่ยนแปลงชนิดของตัวแปรเพื่อดูการผลเปลี่ยนแปลงของมูลค่าสุดท้ายโดยทำการเปลี่ยนชนิดของอินพุตทั้งหมด 4 ชุด ได้แก่ ชุดที่1 ราคาเปิด ราคาสูง ชุดที่2 ราคาเปิด ราคาปิด ชุดที่3 ราคาสูง ชุดที่4 ราคาเปิด ราคาปิด ราคาสูง ราคาต่ำ มูลค่าสุดท้ายอยู่ที่ 949693.0, 955007.0,965853.0, 1003861.0 ตามลำดับ ซึ่งผลที่ออกมานั้นเมื่อเปรียบเทียบกับพอร์ตโฟลิโอที่มีการลงทุนในทุกหลักทรัพย์เท่ากันตลอดช่วงเวลาซึ่งมีมูลค่าวันสุดท้ายอยู่ที่ 902741 พบว่าแบบจำลองที่ใช้มีความสามารถในการจัดสรรเงินลงทุนในระดับที่ดีกว่าทั้งสิ้น โดยมีค่า Sharpe ratio อยู่ที่ -0.01, -0.01, -0.07 และ0.01 ตามลำดับ

5. บทสรุปและข้อเสนอแนะ

จากการศึกษาพบว่าแบบจำลอง Machine learning ที่ใช้อัลกอริทึมแบบ DDPG ที่สร้างขึ้นมานั้น จากการที่ได้ทำการทดลองปรับโมเดลโดยในส่วนของค่า Learning rate Actor และตัวแปรอินพุต แบบจำลองดังกล่าวมีความสามารถในการจัดสรรเงินลงทุนที่ดีกว่าการลงทุนแบบเท่ากันตลอดช่วงเวลา ซึ่งเห็นได้จากผลลัพธ์ของพอร์ตโฟลิโอที่ให้ผลตอบแทนที่สูงกว่าตัวเปรียบเทียบ และมีค่า Sharpe ratioที่สูงกว่า ซึ่งแสดงให้เห็นถึงผลตอบแทนที่สูงกว่าต่อความเสี่ยงหนึ่งหน่วย จึงสามารถสรุปได้ว่า แบบจำลองดังกล่าวมีความสามารถในการเรียนรู้ลักษณะและสร้างความสัมพันธ์ระหว่างข้อมูลอินพุตกับค่าน้ำหนักของโครงข่าย และสามารถนำแบบจำลองมาประยุกต์ใช้กับการหาสัดส่วนที่เหมาะสมสำหรับการลงทุนของพอร์ตโฟลิโอได้

ข้อเสนอแนะสำหรับการการศึกษาครั้งนี้ เนื่องจากแบบจำลองที่ได้ทำการวิจัยนั้นมีความสามารถในการเรียนรู้ข้อมูลเพื่อสร้างความสัมพันธ์ระหว่างข้อมูลอินพุตกับค่าน้ำหนักของโครงข่าย จึงมีจุดที่สามารถนำไปศึกษาเพิ่มเติมได้ เช่น การกำหนดตัวฟังก์ชันรางวัลใหม่โดยใช้ Sharpe ratio ที่สูงที่สุด หรือการเพิ่มชนิดของอินพุต เช่น ข้อมูลปัจจัยทางเศรษฐกิจ ข้อบ่งชี้ทางเทคนิคของหลักทรัพย์ เป็นต้น

เอกสารอ้างอิง

สุมิตรา ตั้งสมวรพงษ์ และบุษบา คงปัญญากุล. (2563). *ส่องตลาดหุ้นไทย ผ่านโครงสร้างผู้ถือหุ้น*. สืบค้นจาก https://www.set.or.th/dat/vdoArticle/attachFile/AttachFile_1606964362157.pdf

ราชฤทธา รัตนวรกานต์. (2562). *การหาค่าเหมาะสมที่สุดของการเรียนรู้เชิงลึกโดยใช้อัลกอริทึมเชิงวิวัฒนาการ*. (วิทยานิพนธ์ปริญญามหาบัณฑิต, มหาวิทยาลัยศิลปากร).

Sharpe, W.F. (1994). The Sharpe Ratio. *The journal of portfolio management Fall 1994, 21*(1), 49-58. doi: https://doi.org/10.3905/jpm.1994.409501

Jiang, Z., Xu, D., & Liang, J. (2017). *A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem*. Retrieved from https://arxiv.org/abs/1706.10059

Liang, Z., Chen, H., Zhu, J., Jiang, K., & Li, Y. (2018). *Adversarial Deep Reinforcement Learning in Portfolio Management.* Retrieved from https://arxiv.org/abs/1808.09940

Huang, G., Zhou X., & Song, Q. (2020). *Deep Reinforcement Learning for Portfolio Management Based on The Empirical Study of Chinese Stock Market.* Retrieved from https://arxiv.org/abs/2012.13773

Hieu, L. T. (2020). Deep Reinforcement Learning for Stock Portfolio Optimization. *International Journal of Modeling and Optimization,* 10(5), 139-144. doi: 10.7763/IJMO.2020.V10.761