IS5006 INTELLIGENT SYSTEM DEPLOYMENT

PROJECT WRITE-UP

Prepared by:

Group 7

Name	Matriculation Number
Ge Xiaomeng	A0112747H
Goh Khai Hong	A0212197H
Shen Siyuan	A0112489B
Xue Bin	A0039717X
Yong Chee Xian Matthew	A0090988J
Zou Yang	A0070179A

Semester 2, AY2020/2021

NATIONAL UNIVERSITY OF SINGAPORE

Table of Content

1. Executive Summary

Cryptocurrency trading strategies play a critical role in Algorithmic trading. However, it is challenging to design a consistent and profitable strategy in a complex and dynamic cryptocurrency market. In this project, we propose an ensemble strategy scheme that automatically learns cryptocurrency trading strategies by maximizing investment return. We train 2 machine learning agents and use 3 other algorithmic agents to obtain an ensemble trading strategy using 5 different algorithms: Bollinger Band, Bollinger Band Trend, Double Duel Recurrent Q-learning agent, Deep evolution and Sentiment Analysis. The ensemble strategy inherits and integrates the best features of the three algorithms, thereby robustly adjusting to different market conditions. In order to avoid the large memory consumption in training networks with continuous action space, we employ a load-on-demand approach for processing very large data. We test our algorithms on the Bitcoin cryptocurrency which have adequate liquidity.

Below summarizes our learning journey in the creation of the MAS model. We will first introduce the solution architect of our best MAS model, with elaboration on the roles of the various agents. Next, we implore into the strength and limitation of the MAS model. Lastly, we address any improvements we could make on the model, if given more time and resources.

2. Multi-Agent System(MAS) Model

We are tasked with designing and implementing a system for Algorithmic trading. This is a multi-agent and case-based reasoning system for algorithmic trading (multithreading Flask server with multiple agents). This system allows users to get recommendations on whether to buy/sell/hold a cryptocurrency in the HitBTC exchange. A MAS with multiple agents interacting with each other is proposed as the solution to the complex and decentralized problem.

The goals of our system are:

- to make recommendation(s) on whether to buy or sell cryptocurrencies
- to take action(s) based on the recommendation(s) etc.

The design of our MAS model is as below.

2.1. Model Solution Architecture

The high level architecture is depicted by the flowchart below.

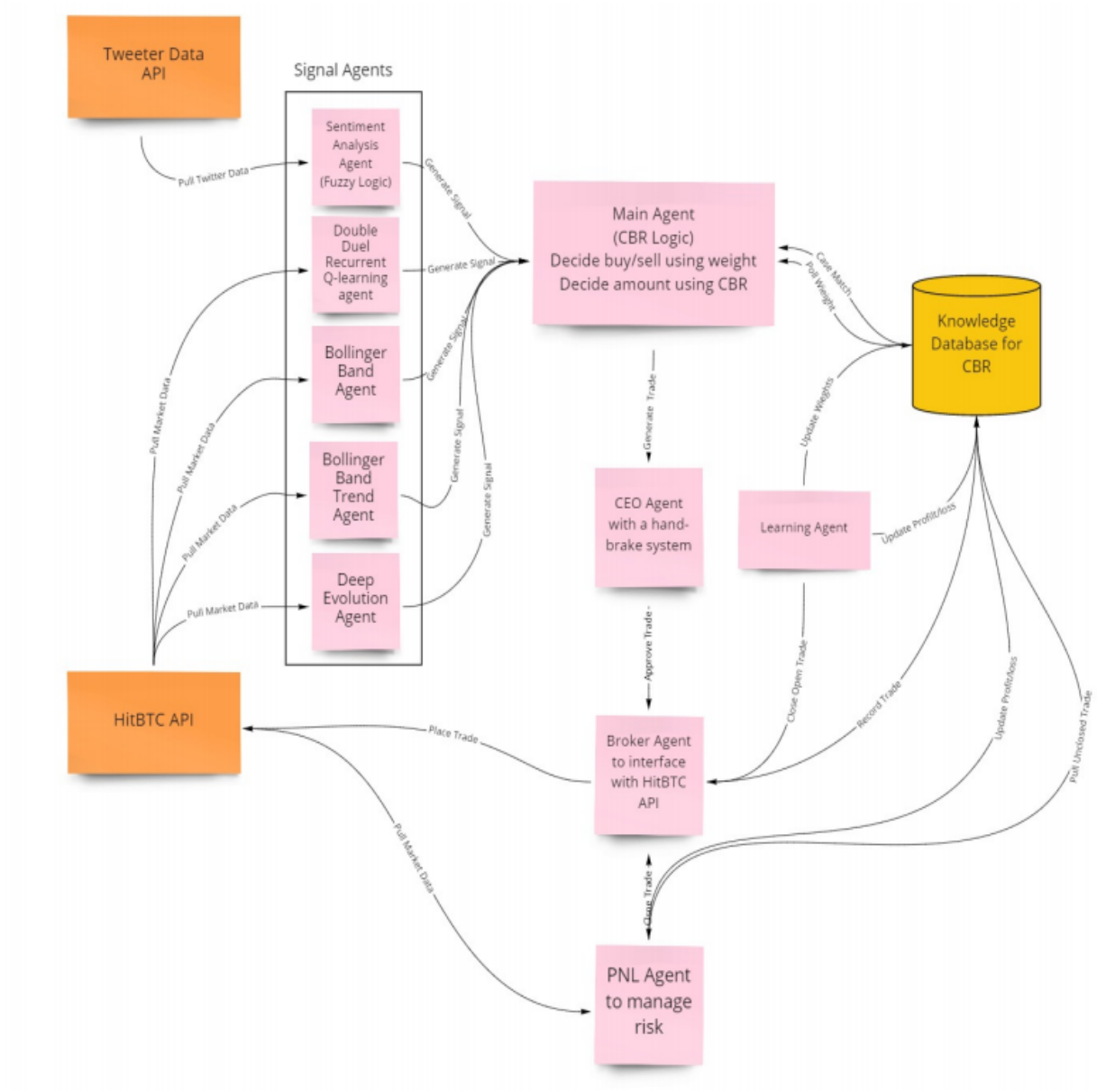

Figure 2.1. Flowchart of MAS model

2.2. Agents

Our MAS model comprises 6 core agents and a knowledge database. The 6 agents are Signal agents, Decider agent, CEO agent, Broker agent, Profit and Loss(PnL) agent and Learning agent.

2.2.1. Signal agents

Signal agents gather information from the market periodically, capturing the market information from different perspectives, both quantitatively and qualitatively. For our model, we have set the period at 1 minute.

As a bonus part of this project, we have added in Bollinger Band agent, Bollinger Band Trend agent to monitor market conditions as well as Deep Evolution agent learning agents and Double Duel Recurrent Q-Learning agent for learning.

1) **Bollinger Band agent:**

Bollinger Bands(BBand) are usually used to gauge market's overbought/oversold conditions. It relies on the concept of mean reversion of the price. This assumes that when a price deviates substantially from its average, it will regress back to its mean. This agent generates a sell signal when the price falls below the lower band and a buy signal when the price rises above the high band. Otherwise, it generates a neutral signal.

The formula used to compute BBand (Mitchell, 2020) is:
lower = mean(20) – 2*std(20))
upper = mean(20) + 2*std(20)

2) **Bollinger Band Trend agent:**

The Bollinger Band Trend(BBTrend) indicator is usually used in conjunction with the BBand indicator. It computes signals for both strength and direction of price, offering an alternative perspective. This agent generates a buy signal when the BBTrend is above 0, a sell signal when BBTrend is below 0.

The formula used to compute BBTrend (Investopedia, 2021) is:
lower = abs(lowerBB(20) – lowerBB(50))
upper = abs(upperBB(20) – upperBB(50))
BBTrend = (lower - upper) ÷ middleBB(20)

3) **Deep Evolution agent:**

Deep evolution is a gradient-free approach to reinforcement learning (Snow, 2019,11). Using genetic algorithms one can find the parameters that best defines a good performing agent. It starts with a certain number of predefined agents (i.e. potential trading strategies) with randomly initialised parameters. The algorithm selects the top performing decile of agents and adds a bit of Gaussian noise to the parameters so that in the next iteration the agent gets to explore the neighbouring space to identify even better performing strategies.

4) **Double Duel Recurrent Q-Learning agent:**

This agent is an online action-value function learning with an exploration policy. It takes into account a series of interactions: action, observe, maximise, adjust policy, and repeatedly iterate again and again

The formula used for the Double Duel Recurrent Q-Learning agent (Snow, 2019, 12) is:

Take some action a_i and observe $(s_i, a_{,i}, s_i', r_i)$

$$y_i = r(s_i, a_i) + \gamma \max_{a'} Q_\theta(s_i', a_i')$$

$$\emptyset \leftarrow \emptyset - \alpha \frac{dQ_\emptyset}{d\emptyset}(s_i, a_i)(Q_\theta(s_i, a_i) - y_i)$$

Then explore with the epsilon-greedy policy:

$$\pi(a_t|a_t) = \begin{cases} 1 - \epsilon \; if \; a_t = argmax_{a_t} Q_\theta(s_t, a_t) \\ \epsilon/(|A| - 1) \; otherwise \end{cases}$$

5) Fuzzy Logic agent:

The Fuzzy Logic agent will work with 2 other agents (Google API agent and Tweepy agent) to generate 'recommendation score' based on sentiment score and sentiment magnitude obtained from Google API agent. The generated recommendation score and the percentages of the labels (positive, negative and neutral) will be used to generate signal -1 (sell signal), 0 (no action) or 1(buy signal) based on the predefined thresholds. The thresholds were fixed based on the results from our simulation.

6) Google API agent:

The Google API agent will act as a "middle man" between our system and Google Cloud Platform (e.g. via Google Sheets and Google Drive APIs). The Google Natural Language API will be used to perform sentiment analysis on the pre-processed tweet and generate sentiment score, sentiment magnitude and the label (positive, negative or neutral). Additionally, textblob library is used to perform sentiment analysis and calculate the subjectivity and polarity of our tweets. Further, it is also used to create, write or append data to Google Sheets in a timely manner.

7) Tweepy agent:

The tweepy agent is used to extract tweet data from Twitter using Twitter API. The agent can be used to pre-process and clean up the extracted tweet data by removing mentions, hashtags, retweets and urls before sentiment analysis on the data.

2.2.2. Decider agent

The Decider agent gathers information from signal agents periodically and makes the final decision: stand(buy/sell/neutral) and trading quantity. It connects to the knowledge database which stores two critical information sets in our model: namely agent weights and trading history.

Agent weights indicate how much decision power each agent has over the final stand to buy/sell an instrument. Quantity is computed with case based reasoning using the trading history. Every time a sequence of signals are retrieved, the agent queries the knowledge database for all historical trade with the same signal sequence. Average profitability is computed based on those historical trades. If the average profitability exceeds a predefined threshold, we double the quantity to trade, otherwise we use a predefined default quantity.

2.2.3. CEO agent

The CEO agent acts more like a reviewer to all the decisions from the decider agent. It approves/rejects decisions based on a set of heuristics (for e.g. Do not trade when volume is low). The CEO agent checks the balance of the account. If there is sufficient balance, it places an order, otherwise it aborts the order. For every order, it also computes the take profit and stop loss level. Lastly, it records the order into the trade book in the knowledge database once it's placed.

2.2.4. Broker agent

The broker agent handles all information exchange between the system and the exchange. It integrates with the exchange and exposes below 9 methods:

- Get_ohlcv_data
- Get_balance
- Get_ticker_price
- Place_market_buy_order
- Place_market_sell_order
- Place_limit_buy_order
- Place_limit_sell_order
- Get_order_status
- Cancel_order

2.2.5. Profit and Loss (PnL) agent

The PnL agent manages risk. We do not want one trade to blow up our entire portfolio. The profit/loss agent polls the tradebook frequently to check and update order status. If an order is still open, it updates the order status as "open"; If an order is filled, it updates the order status as "closed"; If an order is cancelled, it updates the order status as "cancelled". It also checks if any take profit/stop loss level is violated. If so, it simply closes the trade to secure profit/reduce loss.

2.2.6. Learning agent

The learning agent updates itself periodically according to agents' performance(based on profitability). Trading history captures all the activities in the past. Additionally, it provides near real time reporting for users to take a glimpse of the system.

The learning agent runs only after each trading cycle. The cycle period can be configured. At the end of each cycle, it cancels all open orders and closes all open trades (trades with empty close prices and profit/loss). It then computes the long profitability and short profitability of each agent. The weight is computed as (long profitability/short profitability)-1. If the weight is below 0. It'll be adjusted to 0.

2.2.7. Knowledge database

The knowledge database stores two sets of information - agent weights and pertinent past cases. It also saves them periodically into disk and google sheets.

Agent weights can be accessed at https://drive.google.com/file/d/1zQ0HVSiGXFzzT2Bzdj1ZR66_zckFt3ng8jc1O9syjdU/edit.

Pertinent past cases can be accessed at https://drive.google.com/file/d/1uiy8oFF4xEd6TtJT46KB8KA863J_iAVwFBHyP1UV2xs/edit

3. Incorporating learnings from class

The design of our MAS model heavily relies on our learnings from the lectures, assignments and guest speakers.

3.1. Forward and backward chaining

Equipped with the principles for forward and backward chaining, our MAS model is able to operate optimally by considering the time taken for various tasks, optimising the decision making process for each agent. Adding relevant heuristics at critical processes allows us to prevent potential major faults in the model.

3.2. Fuzzy logic

Our secondary research shows that cryptocurrency reacts positively/negatively to certain twitter accounts. The use of fuzzy logic on sentiment analysis provides a relevant signal for our model to act upon.

3.3. Case Based Reasoning

Incorporating CBR allows our model to learn from past trades and helps augment the decision making processes.

3.4. Insights from guest speakers

With their insights, we were able to make tweaks that improved the model's performance. One change we made was to have the CEO agent execute limit orders instead of market orders. This allows us to obtain a more favourable price for the trades.

3.5. Importance of documentation

Throughout the course of this final project, we have documented our thought processes and experimentations in GitHub.

3.6. Strengths

As we design the MAS model, we make sure to cover the end-to-end processes. This has allowed us to create a system that is production ready.

3.7. Limitations

1) Thin Volume of Demo Digital Exchange

The Demo hitbtc exchange has very thin volume. The price spread for BTC/USDT is usually a few hundreds. This makes it very difficult to perform real time simulation as orders take a while to fill. With risk management in place, we should practically never trade. However, for the purpose of this project, we've removed the constraint of volume check for the sake of simulation.

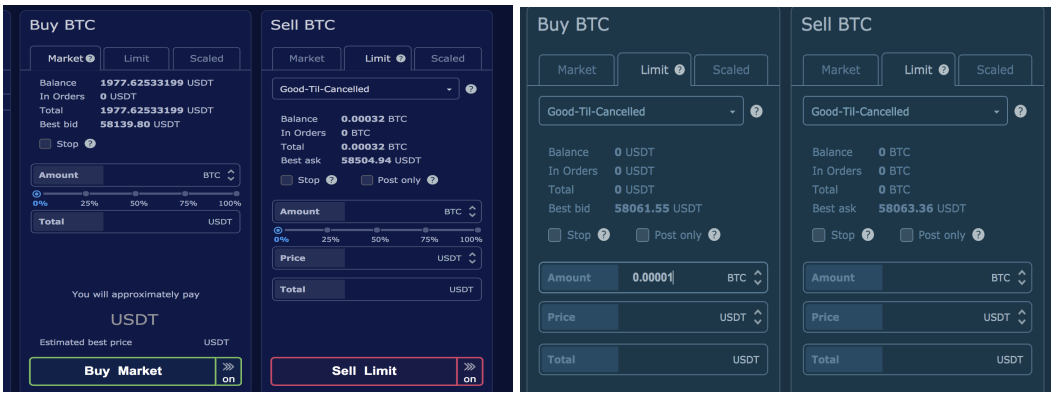

Figure 3.7.1. Demo vs Live Bid/Ask Price

2) Not real time

The profit/loss agent is not real time(currently running every minute). In case there is a huge fluctuation of price within the minute, the risk management logic may not kick in timely.

3) Not full time

The system may not be online 24/7. When the system goes down, all the open trades will be subjected to the force of the market without risk control.

4. Bonus

In our MAS model, we have incorporated more novel techniques such as BBand and BBtrend. Additionally, we have explored the use of machine learning to generate meaningful signals to aid the decision agent.

5. Improvements

An improvement we could have made, if given more time and resources, is to incorporate more advanced CBR techniques. From our research, CBR techniques such as Dynamic Adaptive Ensemble(DAE) CBR (Chun & Park, 2005) and Regression CBR(RCBR) (Chun & Park, 2006) have been shown to perform better than conventional CBR, using composite neighbours.

6. Conclusion

In conclusion, we have researched about MAS and CBR as well as trading strategies for cryptocurrency arbitrage and built a production-ready MAS model. Our model is made up by 6 core agents and a knowledge database, and is able to not only make recommendations on whether to buy or sell cryptocurrencies but also to take actions based on the recommendations. For ease of interpretation, we have also enabled excel outputs of trading record, PnL information and graph representations. We have also tested our algorithms. In addition, we have examined our model from the benefit and risk perspective. From this project, we see great potential in application of MAS in algo trading and become more vigilant on the risks embedded.

7. References

Chun, S.-H., & Park, Y.-J. (2005). Dynamic adaptive ensemble case-based reasoning: application to stock market prediction. *Expert Systems with Applications*, *28*, 435-443.

Chun, S.-H., & Park, Y.-J. (2006). A new hybrid data mining technique using a regression case based reasoning: Application to financial forecasting. *Expert Systems with Applications*, *31*, 329-336.

Investopedia. (2021, March 16). *What are the best indicators to use in conjunction with Bollinger Bands?* Investopedia. Retrieved March 25, 2021, from https://www.investopedia.com/ask/answers/121014/what-are-best-indicators-use-conjunction-bollinger-bands.asp#:~:text=Using%20the%20%25b%20Indicator,the%20upper%20and%20lower%20bands.&text=This%20is%20helpful%20for%20traders,determine%20divergences%20and%20

Mitchell, C. (2020, November 16). *Using Bollinger Bands to Gauge Trends*. Investopedia. Retrieved March 26, 2021, from https://www.investopedia.com/trading/using-bollinger-bands-to-gauge-trends/

Snow, D. (2019, July 16). Machine Learning in Asset Management. *Elsevier*.