MSE 448 Final Presentation

Aman Sawhney, Yang Fan, Chris Lazarus

June 2, 2021

Outline

Overview

- (Refresher) HFT as an MDP
- Data
- Strategy Overview
- Simulation Assumptions
- Technical Features and Feature Engineering
- Learned Model Trading Results
- GAN Results
- Next Steps

High Frequency Trading as an MDP

Since HFT strategies rely on taking and providing liquidity when it is appropriate, we make the modeling assumption that order book dynamics are a markov process. Hence we may formulate a HFT strategy as a Markov Descion Process in the following manner:

- We will assume discrete time intervals which will be determined by our time-scale, T.
- For each time step $0 \leq t \leq T$ we have
  - $s_t := (O_t, q_t)$ where O_t is the order book history at time t over a look back period steps and q_t is the amount of the asset the agent currently holds.
  - $a_t \in \{T, P, N\}$ where T is the act of taking liquidity, P is the act of providing liquidity, and N is the act of doing nothing.
  - r_{t+1} is an appropriate reward function.

This framework generalizes to a multidimensional asset space.

Reinforcement Learning

There are two main approaches to solving RL problems: value-based methods (ie. Q-learning) and policy search methods (ie. policy gradient).

- Deep Q-Learning (DQN) - minimizes MSBE, off-policy, sample efficient, generally good for discrete and low dimensional action and state spaces
- Proximal Policy Optimization (PPO) - maximizes expected return, on-policy, sample inefficient, generally good for continuous action and state spaces

We tried DQN and PPO:

- DQN - showed good performance
- PPO - abandoned due to low performance and high computational cost

Data

We pulled top of the book data from MayStreet aggerated by second, from 9:30AM to 11:30AM for the first 5 months of 2021 for the 5 S&P 500 stocks with the highest beta.
This amounted to a massive data set with well over 100 million rows.

Strategy Overview

We assume

- The starting account balance of our agent is the cash value of 6000 shares at the opening price of a given security
- The agent is able to trade with two times leverage
- All entered positions must be exited after a two minute holding time
- At any given second the agent is able to buy the minimum of 100 shares of the ask size and sell the minimum of 100 shares and the bid size.
- The agent must always buy at the ask and sell at the bid price.
- The agent must maintain a net worth greater than 0. I.E. the value of its positions plus the cash held as balance must be greater than 0. Otherwise, trading must end.
- The agent can trade from 9:32 am to 10:02 am.

Strategy Overview (count.)

Given these assumptions, our agent must optimize buy, sell, and hold actions to maximize the following reward function:

$$r_t = 1_{t<T} \alpha * return_t + 1_{t=T} \beta * R_t$$

Where t is current second, T is the final time step and a_t is the action at time t, $return_t$ is the two minute return of a_{t-120}, R_T is the overall return, and α and β are hyperparmeters .

Technical Features and Feature Engineering

- The order book data: bid/ask price, size, number of providers; adjusted volume
- Technical indicators: SMA, EMA, RSI, ROC, TRIX, PPO, PVO, AROON, DPO, MACD, SRI
- Re-normalize some of the indicators against the first value encountered in the beginning of each episode, to increase performance when feeded into NN
- Maxmial Fourier modes are mostly 0 over short horizons, and requires huge prepossessing time.
- Custom NN structures as feature extractors, with every obs as a F by L matrix, where F is the number of features and L is the amount of history we allow the agent to look back.
  - Large MLP networks
  - LSTM + MLP (1D LSTM running over the L dimension)
  - Transformer + MLP (with the same obs matrix feeding into encoder and decoder)

Sanity Check with Zigzag patterns

- We create a counterfactual order book with oscillating linear patterns ranging from 10 to 20, with 0 gap between the bid and ask price.
- Implemented technical indicators and normalization as specified before.
- Adapted holding periods (5s) for to match the period of oscillation (20s).
- Comparison between strategies with/without forced liquidations.
- Trained different models with similar amount of computing cost.

Sanity Check with Zigzag patterns

MLP without forced liquidation
Define rewards for every step

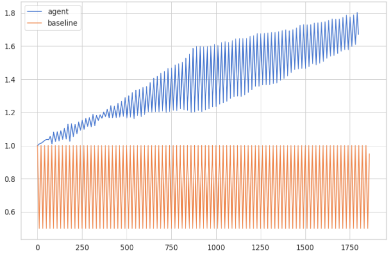


Figure: Portfolio Value

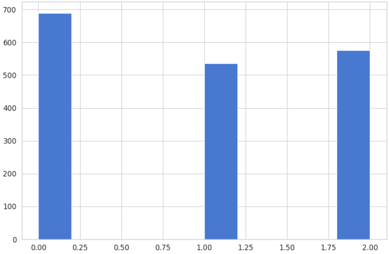


Figure: Actions

Actions = 0: short, 1: buy, 2: hold

Sanity Check with Zigzag patterns

MLP with forced liquidation
Define rewards for every step

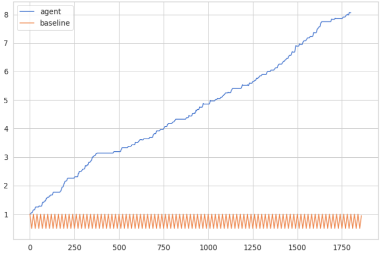


Figure: Portfolio Value

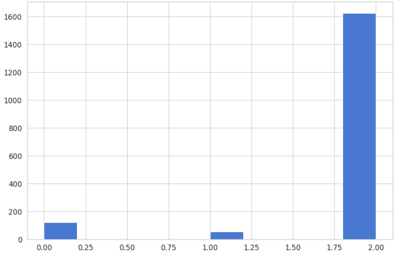


Figure: Actions

Actions = 0: short, 1: buy, 2: hold

Sanity Check with Zigzag patterns

LSTM + MLP with forced liquidation
Define rewards for every step

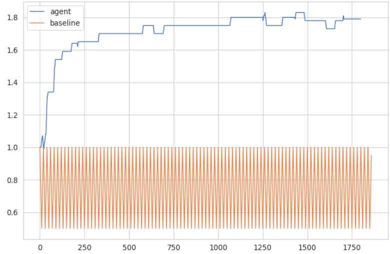


Figure: Portfolio Value

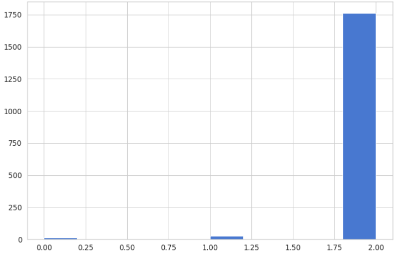


Figure: Actions

Actions = 0: short, 1: buy, 2: hold

Sanity Check with Zigzag patterns

Transformer + MLP with forced liquidation
Define rewards for every step

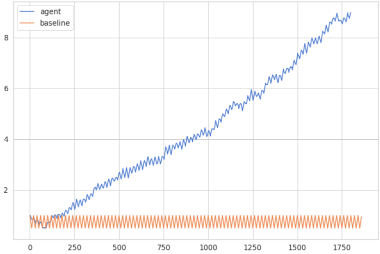


Figure: Portfolio Value

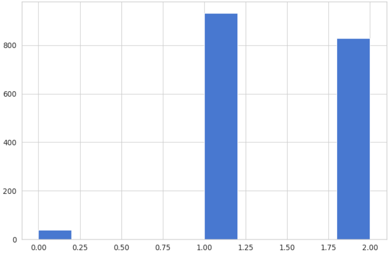


Figure: Actions

Actions = 0: short, 1: buy, 2: hold

Sanity Check with Zigzag patterns

MLP without forced liquidation
Define rewards for only the terminal step as the total return

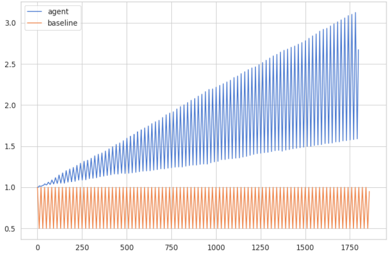


Figure: Portfolio Value

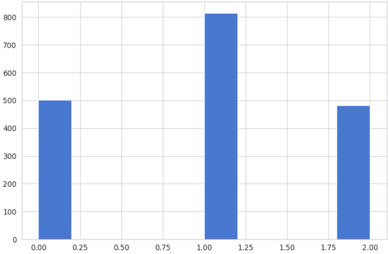


Figure: Actions

Actions = 0: short, 1: buy, 2: hold

Sanity Check with Zigzag patterns

MLP with forced liquidation
Define rewards for only the terminal step as the total return

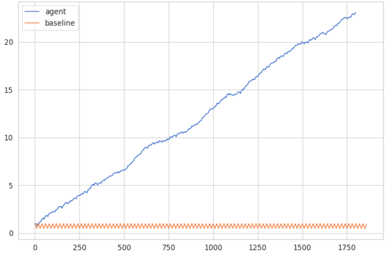


Figure: Portfolio Value

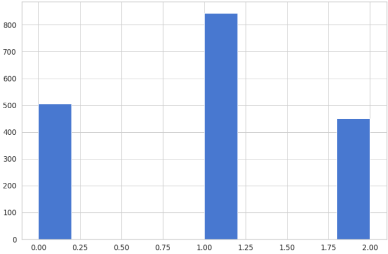


Figure: Actions

Actions = 0: short, 1: buy, 2: hold

Sanity Check with Zigzag patterns

LSTM + MLP with forced liquidation
Define rewards for only the terminal step as the total return

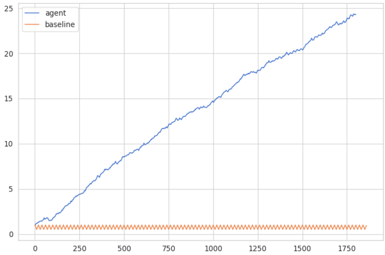


Figure: Portfolio Value

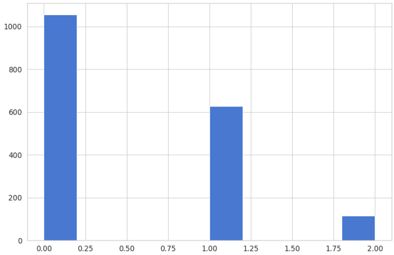


Figure: Actions

Actions = 0: short, 1: buy, 2: hold

Sanity Check with Zigzag patterns

Transformer + MLP with forced liquidation
Define rewards for only the terminal step as the total return

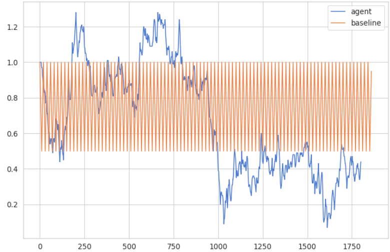


Figure: Portfolio Value

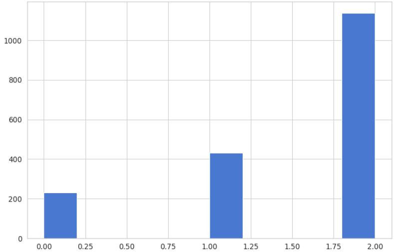


Figure: Actions

Actions = 0: short, 1: buy, 2: hold

Sanity Check with Zigzag patterns

- MLP might be sufficient as compared to more complicated networks, with the added benefit of being more consistent. We can always add non-linearity in the feature engineering step.
- Forced liquidation after a short period of time helps the agent.
- There is room to try different reward functions, might be the key to the agent's performance.

Simulation Method

To train our RL agent we built an environment using Open AI's gym interface. Each episode of training loads a random day and random symbol's data from a train or test time dataframe. Compared to other RL projects, such as FinRl and other RL environments, our environment has the following advantage:

- Homogeneous Trading Horizons: By sampling from the same time of day, as compared to randomly sampling along a stock path, our episodes contain very similar market micro structure
- Data Density: Most other academic projects use daily data. Since we use data aggregated on the second level, the data is far more dense.
- Realism: In the real world, high frequency trading models should be adapt at providing sustainable profit across a variety of symbols. Hence the variety of symbols allows train a more realistic model.

Learned Model Trading Results

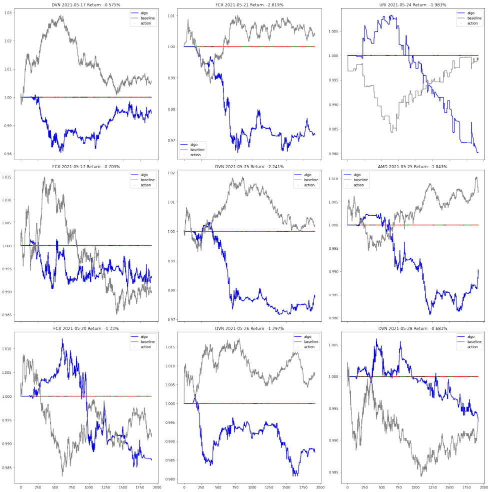


Figure: Model Results Using Multiple Symbols

Learned Model Trading Results (cont.)

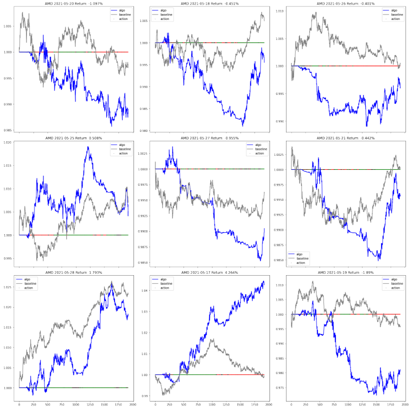


Figure: Model Results Using Only AMD

If we take the previous return sequences, we obtain a sharpe ratio for the 1 second time period of 0.00229. Annualized, assuming you can only trade half an hour a day, that is a sharpe ratio of 1.54.

Learned Model Trading Results (cont.)

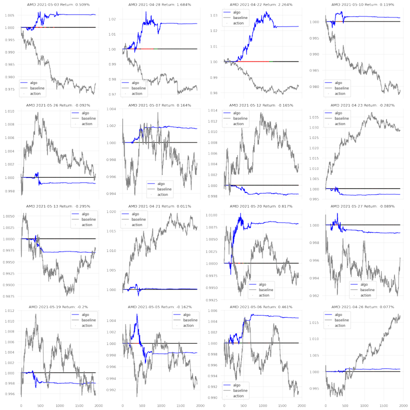
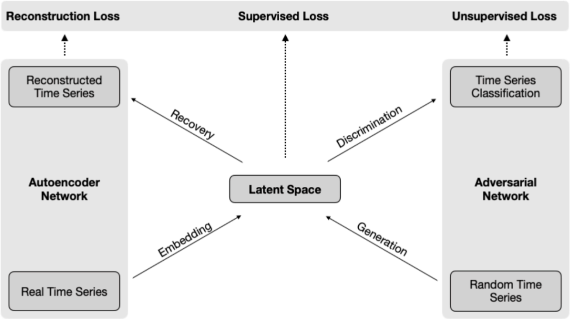


Figure: Model Results Using Only AMD Using A Larger Model

If we take the previous return sequences, we obtain a sharpe ratio for the 1 second time period of 0.0131. Annualized, assuming you can only trade half an hour a day, that is a sharpe ratio of 8.82.

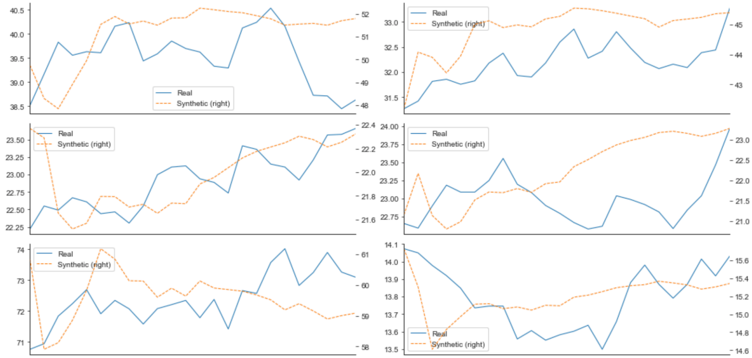
Data Augmentation using GANs [WKKK20]



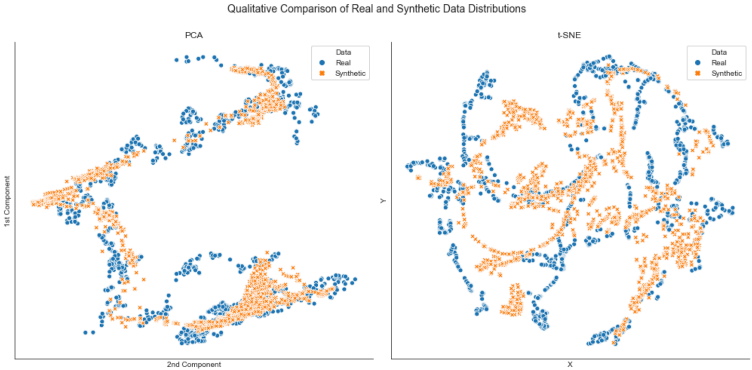
- Autoencoder: embedding and recovery networks - stacked RNN and a feedforward network
- Adversarial Network sequence generator and sequence discriminator components - RNN as generator and a bidirectional RNN with a feedforward output layer for the discriminator

Source: [Jan20]

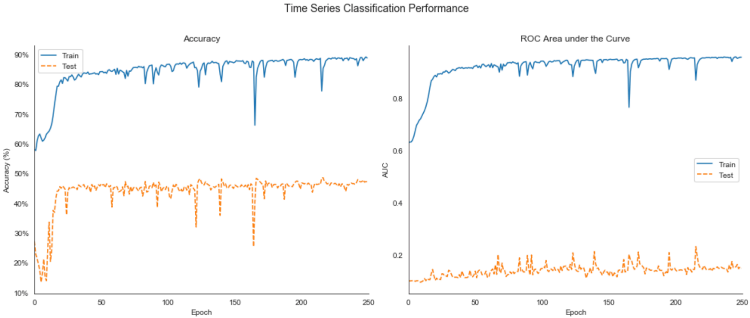
GAN Results - 1



GAN Results - 2



GAN Results - 3



Next Steps

- Use the synthetic data to train the agent
- Analyze and compare the effect of reward function choice
- Finalize hyperparameter tuning
- Train more specialist traders.

References I

Stefan Jansen, *https://github.com/stefan-jansen/synthetic-data-for-finance*, Deep RL Workshop, NeurIPS 2020 (2020).

Magnus Wiese, Robert Knobloch, Ralf Korn, and Peter Kretschmer, *Quant gans: deep generation of financial time series*, Quantitative Finance **20** (2020), no. 9, 1419â 1440.