User guide

This Chapter explains the components in my solution and how to execute the code.

B.1 Review of Files

The following is a list of scripts and their corresponding functionality.

- ctrader/dataCollector.algo $\rightarrow$ collects order flow imbalance features from the limit order book on the CTrader FXPro platform and appends to a local CSV file.
- ctrader/forwardTestExecute.algo $\rightarrow$ sends OFI data to the signal generating script (through POST requests) and parses the signals on the CTrader FXPro platform to execute live trades (on a demo account).
- trainers/alphaExtraction.py $\rightarrow$ trains a linear regression model (supervised learning component) to extract alphas at multiple horizons from the order flow imbalance features, then tests on unseen data.
- trainers/dqn.py $\rightarrow$ trains a deep Q network (reinforcement learning component) to learn when to buy or sell when given alphas at multiple horizons as input, then tests on unseen data.
- trainers/ddqn.py $\rightarrow$ trains a double deep Q network (reinforcement learning component) to learn when to buy or sell when given alphas at multiple horizons as input, then tests on unseen data.
- trainers/qLearning.py $\rightarrow$ uses Q learning (reinforcement learning component) to update a Q table to associate actions of buying and selling with states of alphas at multiple horizons, then tests on unseen data.
- backtesting.py $\rightarrow$ runs the backtesting combining the alpha extraction with each reinforcement learning agent on test data and returns performance metrics for each instrument.
- config.py $\rightarrow$ contains global variables used in this project.
- forwardTestSignals.py $\rightarrow$ hosts a web server and parses OFI data from the CTrader FXPro platform script (through POST requests) and sends signals for making trades on the platform using the best performing reinforcement agent for the best instrument.

- hyperparameterTuning.py → finds and returns the best configuration of hyperparameters for all models for each instrument.
- plots.py → contains methods to generate plots and tables.
- utils.py → contains utility methods, model architectures, and a reinforcement learning environment to allow iterating through historical data.

B.2 Setup

1. Download CTrader FXPro [33]
2. Copy dataCollector and forwardTestExecute into FXPro's working directory
3. Create a Python virtual environment with version 3.9.13
4. Install the Python dependencies with

   ```
   pip install -rrequirements.txt
   ```

B.3 Execution

1. Collect OFI data with the dataCollector script on the CTrader FXPro platform
2. Run the tuning script to find the best parameters for the models with:

   ```
   python -m hyperparameterTuning path_to_data_dir
   ```

   and update the config script with the best parameters for each model
3. Train the supervised learning component for alpha extraction with:

   ```
   python -m trainers.alphaExtraction path_to_ofi_data
   ```

4. Train each reinforcement learning agent (Q Learning, DQN, and DDQN) with:

   ```
   python -m trainers.qLearning path_to_ofi_data
   python -m trainers.dqn path_to_ofi_data
   python -m trainers.ddqn path_to_ofi_data
   ```

5. Perform backtesting with:

   ```
   python -m backtesting path_to_data_dir
   ```

6. Perform forward testing by hosting the web application with:

   ```
   python -m forwardTestSignals qLearning/dqn/ddqn
   ```

   and then run the forwardTestExecute script on the CTrader FXPro platform