Overview Report: Transformer Agent for Trading Strategy

Introduction

This report gives a summary of how to put into action and assess a Transformer-based agent for trading strategy. The given task was to "Implement and Fine-Tune Transformer-Based Model."

The ipynb notebook that has been submitted, together with this report, builds upon the provided submission files.

It has the dataset prep and PPO agent training and evaluation code already given. My submission is made using this foundation.

Here are the outlines for the Task given and impelemented :

Data Preprocessing and Feature Engineering

1. Dataset Preparation:

The dataset contains characteristics of the market such as 'Close', 'Volume', 'High', 'Low' and 'Open'. It also includes various technical indicators:

- **Momentum Indicators:** RSI, MACD, Stochastic Oscillator
- **Volume Indicators:** On-Balance Volume (OBV)
- **Volatility Indicators:** Bollinger Bands (BB), Average True Range (ATR)
- **Trend Indicators:** Average Directional Index (ADX), Commodity Channel Index (CCI)
- **Other Indicators:** Log returns (DLR), Time Weighted Average Price (TWAP), Volume Weighted Average Price (VWAP)

**These features were already given with the PPO implementation. **

2. Target Generation:

I have used a strategy rooted in moving averages to set specific goals during the stage of model training.

Targets are created based on a combination of moving averages and RSI:

- **Buy Signal:** Short-term MA is higher than long-term MA and RSI is below a certain level (30).
- **Sell Signal:** If the short-term MA falls below the long-term MA and RSI rises above a certain level (70).
- **Hold Signal:** In all other cases.

Transformer Model Implementation

1. Model Architecture:

The model applies a Transformer design, which is very effective in capturing the order of elements and managing time-series information. The architecture has these parts:

- **Embedding Layer:** Transforms input features into a higher-dimensional space.
- **Transformer Layers:** Multiple layers of Transformer blocks that process the sequence data.
- **Fully Connected Layer:** Outputs the probabilities of the three actions (buy, sell, hold).
- **Batch Normalization and Dropout:** Enhance training stability and reduce overfitting.

2. Training Setup:

- **Loss Function:** Cross-Entropy Loss to handle multi-class classification.
- **Optimizer:** AdamW optimizer for better handling of sparse gradients.
- **Learning Rate Scheduler:** Drops the learning rate by a factor of 10 after every 30 epochs to adjust it on a regular basis. I chose an initial learning rate of 0.01.
- **Gradient Clipping:** Prevents the exploding gradient problem during backpropagation.

Trading Environment with Blotter

The code was created using the provided, already existing codebase for Blotter. It was intended specifically for PPO agent.

1. Environment Design:

The environment imitates trading by performing actions (buy, sell, hold) according to model forecasts. It keeps track of how much money is in the portfolio, how many shares are being held and calculates rewards.

2. Reward Calculation:

Rewards are calculated based on:

- **Slippage:** Difference between expected and actual price.
- **Transaction Cost:** Proportional to trade volume and market volatility.
- **Time Penalty:** Based on the delay between decision and execution.

3. Running the Simulation:

- The environment is reset at the start of each episode.
- The model predicts actions based on current state observations.
- Perform actions, calculate rewards and move the environment to the next state.
- The simulation continues until all data points are processed.

4. Performance Metrics:

The model is assessed by looking at total rewards and how much the trading strategy earns overall.

Results and Evaluation

1. Training Performance:

We watch the training process by checking loss and accuracy measurements across various epochs. The model shows good equilibrium in forecasting buy/sell signals with accuracy while reducing false positives.

The Accuracy finally achieved on evaluation was around **98.25%**

2. Trading Blotter:

The trading blotter is a logbook that keeps track of every trade made within the simulation. It includes time stamps, prices for buying or selling stocks, how many shares were traded and the reward given. This detailed record assists in studying the model's choices and their economic

effects.

3. Simulation Outcome:

The last portfolio value and total rewards give understanding about the model's performance in a practical trading situation.

Conclusion

Please find the attached ipynb notebook for referencing the code.