Blockhouse: Machine Learning Engineer Work Trial Documentation

1. Introduction

1.1. Overview:

This document details the implementation, fine-tuning, and evaluation of a transformer-based model designed to generate trade recommendations. The project is part of a work trial to assess machine learning skills and proficiency in handling transformer models for financial market data.

1.2. Objective:

The objective of this task is to develop and fine-tune a transformer-based model capable of processing trade and market data to generate actionable trade recommendations (Buy, Sell, Hold). The model's performance will be evaluated against a simple trading blotter.

2. Setup and Environment

2.1. Tools and Libraries:

- Programming Language: Python
- Libraries: PyTorch, NumPy, Pandas, TA-Lib, Stable-Baselines3, Scikit-Learn, SciPy, Gym

Note: I was unable to use TA-Lib as specified in the provided PPO Baseline notebook. Instead, I utilized the 'ta' library, which offers similar functionality. Consequently, I modified the TechnicalIndicators function accordingly.

- Dataset: xnas-itch-20230703.tbbo.csv
- Provided PPO Implementation: Baseline PPO

3. Model Implementation

3.1. Transformer Model Architecture:

Transformer Encoder:

- Number of Layers: 3
- Input Features (in_features): 95
- Feedforward Network Dimensions: 256
- Number of Heads (nhead): 5

Fully Connected Layer (fc_out):

- Input Features (in_features): 95
- Output Features (out_features): 1

3.2. Transformer Components

- **Transformer Encoder (self.transformer_encoder):**
  - Multi-Head Attention (self_attn): Allows the model to focus on different parts of the input sequence, capturing dependencies between different time steps.
  - Feedforward Network (linear1, linear2): A two-layer feedforward network with an intermediate dimensionality defined by dim_feedforward.
  - Normalization (norm1, norm2): Layer normalization is applied after the self-attention and feedforward network to stabilize and speed up the training.
  - Dropout (dropout1, dropout2): Dropout layers are included to prevent overfitting by randomly dropping units during training.
- **Fully Connected Output Layer (self.fc_out)**:
- Takes the output from the last time step of the transformer encoder and transforms it into the desired output dimensionality (Generateing the next predicted price based on the processed input sequence.).

3.3. Data Preprocessing : I have selected following features which I feel provide a proper balance between price, volume, technical indicators, and key action-related information.

- **price**: Represents the price at which a trade occurred.
- **bid_px_00, ask_px_00**: The best bid and ask prices, respectively.
- **size**: Represents the size (quantity) of the trade.
- **Volume**: Trading volume within the period.
- side: Information about side of the trade.
- **ATR_1, ATR_2, ATR_5, ATR_10, ATR_20**: Average True Range, used to measure market volatility over different periods.
- **TWAP (Time-Weighted Average Price):** Average price over a specific period.
- **VWAP (Volume-Weighted Average Price):** Average price based on volume, a key indicator in trading.
- **CCI (Commodity Channel Index):** Measures the deviation of the price from its average.
- **DLR:** Possibly a custom indicator.
- **RSI (Relative Strength Index)**: Measures the speed and change of price movements.
- **MACD** : Moving Average Convergence Divergence, used to spot changes in the strength, direction, momentum, and duration of a trend.

3.4. PPO Model :

- Environment: A custom trading environment is created to simulate stock trading.
- PPO Agent: The PPO agent is trained using reinforcement learning to maximize trading performance. It interacts with the trading environment and makes decisions based on the predicted prices.
- The PPO model works in tandem with the TradeTransformer model, where the transformer provides predictions on future prices, and the PPO agent uses these predictions to make strategic trading decisions.
- This combination of deep learning (for price prediction) and reinforcement learning (for decision-making) creates a powerful framework for algorithmic trading, aiming to achieve superior market performance.

4. Fine Tuning

4.1. Hyperparamater Tuning:

The hyperparameters of the transformer model were fine-tuned using GridSearch. After conducting GridSearch, the following hyperparameters were determined to be the best:

- **Batch Size**: 128
- **Learning Rate**: 0.0001
- **Number of Heads**: 5
- **Number of Layers**: 3
- **Dimension of Feed-Forward Network**: 256

5. Evaluation

The model's performance was tested against baseline models, including PPO and DQN. The key findings are:

- **Baseline Models**:
  - **PPO Model**: Served as a reference for performance.
  - **DQN Model**: Another benchmark for comparison.
- **Transformer + PPO Model**:
  - **Effective Trade Execution**: The combination of Transformer and PPO showed strong trade execution.
  - **Competitive Portfolio Values**: This model achieved better portfolio values compared to the baseline models.

6. Examples of Generated Recommendations

Transaction Details:

1. **Step 59015: BUY**
   - **Price**: $185.31
   - **Shares**: 0.177

- **Reasoning**: Buying a moderate number of shares (0.177) at a lower price is generally sensible. The model is initiating a position with a reasonable amount of shares.

2. **Step 59021: SELL**
    - **Price**: $194.65
    - **Shares**: 0.122
    - **Reasoning**: Selling a smaller portion of shares (0.122) at this price suggests the model is taking some profits while retaining a portion of the position. This partial sell could be a strategy to lock in gains while still holding some shares in case the price continues to rise.
3. **Step 59022: SELL**
    - **Price**: $192.13
    - **Shares**: 0.374
    - **Reasoning**: Selling a larger portion of shares (0.374) at a slightly lower price than the previous sell could indicate the model is aiming to exit a larger part of the position before the price potentially drops further. This could be a strategy to reduce exposure or capitalize on a peak before a potential downturn.

Share Allocation Reasoning:

- **Buy Action:** The model buys a moderate number of shares, which seems appropriate for the initial entry point.
- **Sell Actions:**
    - **First Sell:** The smaller quantity (0.122 shares) suggests taking a portion of profits while keeping some shares to potentially benefit from further price increases.
    - **Second Sell:** The larger quantity (0.374 shares) at a slightly lower price might indicate a strategy to offload a significant portion of the position before the price declines further or to avoid holding too long in a potentially volatile market.

The model's choice to sell a portion of shares at a higher price and then sell a larger portion at a slightly lower price shows a balanced approach to managing the position. It indicates an attempt to secure profits while managing risk by reducing exposure as the price changes.

7. Future Work:

- Expand Dataset: Train on a larger dataset to improve reliability.
- Add Domain Knowledge: Use relevant features and outputs for better performance.
- Explore Encoder Transformers: Consider using encoder transformers for improved feature representation.
- Implement Safety Measures: Add fail-safes to reduce financial risk

8. Conclusion:

This project has successfully implemented and refined a transformer-based model with a PPO agent for trade recommendations. The results are promising, showing effective trade execution and decision-making, with room for further improvement.

9. Acknowledgements:

- Blockhouse Work Assessment GitHub: https://github.com/Blockhouse-Repo/Blockhouse-Work-Trial
- I drew inspiration from the notebook to gain background knowledge and familiarize myself with the implementation. I have thoroughly understood the code and am confident in my ability to explain the implementation in detail. Rest assured, my work is based on a solid understanding of the concepts and techniques demonstrated in the notebook.