# Feature Engineering in RL for Algorithmic Trading

## Investigating the Effects of State Representation on Trading Agent Performance in the Forex Market

## 01. Background

- Reinforcement Learning (RL) has shown a promising future for creating autonomous agents that can learn to trade profitably in the forex market, a global marketplace where currencies are exchanged.
- In this AI paradigm, an agent learns optimal behaviour through trial-and-error by performing actions within a market environment and receiving feedback in the form of numerical rewards.
- Despite this potential, its adoption in the financial industry remains limited due to perceived risks associated with RL-based agents. This creates a clear gap between the technology's promise and its practical use.
- This research focuses on the agent's state representation: the specific collection of data (like market prices, technical indicators, and agent status) provided to the agent to inform its decisions. The design of this input is a critical factor for success.
- This work aims to provide insights for developing more effective, robust, and safer trading agents by systematically investigating how different state representation designs impact performance.

## 02. Research Questions

**What are the impacts of different state representation designs on the performance of an RL-based low-frequency forex trading agent?**

1. **(SQ1)** How does the inclusion or exclusion of specific features (Technical Indicators, Trading Agent Data) affect agent performance?
2. **(SQ2)** How does the quantity of information impact agent performance?


TU Delft

**Finn van Oosterhout**

N. Yorke-Smith

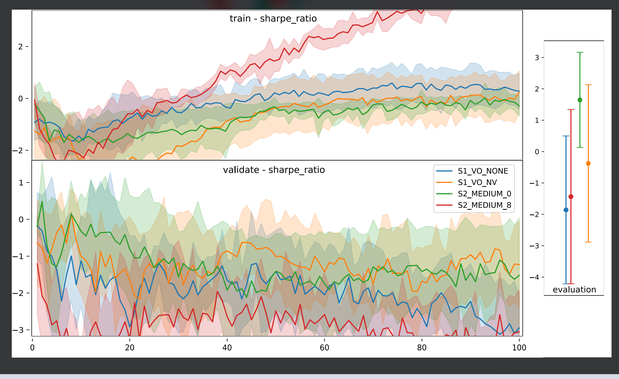
A. Papapantoleon

A. Kolarijani


## 03. Methods

- **Market Environment:** We developed a custom simulated forex environment. This environment is fed historical EUR/USD candlestick data, simulating each candle's open, high, low, and close price movements at each interaction from the agent. We model life-like conditions by incorporating the bid-ask spreads and commission fees.
- **The Features:** The environment supplies the agent observations about the state of the environment. These observations are divided into five categories: Time, Trend, Momentum, Volatility, and Agent features.
- **Two-Phase Experimentation:** We designed a systematic, two-phase protocol to test our research questions:
  - **Phase 1** investigated the impact of feature types, systematically adding or removing features in each category to see their influence.
  - **Phase 2** investigated the quantity of information by varying the historical lookback from 0 to 32 prior timesteps.
- **Rigorous Experimentation:** To increase stability and reliability of the results, every experiment was run across five different random seeds. Additionally, the dataset was split up into three sets: training, validation, and evaluation. The agents were saved at the end of every episode, which were then each evaluated against the train and validation dataset. A final model was selected based on the combined performance, which was then run on the evaluation dataset.

### Results




## 04. Results

- **A "Less is More" Approach:** For Time, Momentum and Volatility features, providing a single, well-chosen indicator outperformed using multiple indicators, which tended to introduce confounding noise. Conversely, for Trend features, providing all available indicators was beneficial. Across all categories, providing at least one feature was more beneficial than zero, showing the importance of each category.
- **The Peril of Too Much History:** Surprisingly, increasing the quantity of information, both by adding more features and by extending the historical lookback window, consistently decreased the agent's performance. The best results were achieved with little to no historical context, as more data led to the agent overfitting to the data rather than learning general strategies.
- **High Instability and Sensitivity:** Across most experiments, the agent struggled to achieve reliable profitability, showing high variance across different runs even with identical settings. Performance was extremely sensitive to initial conditions and even the order of features in the input vector.
- **Concluding:** In our context, a precise balance of features was needed, due to overfitting when presented with a larger quantity of information.

## 05. Conclusion & Future

**Designing an effective state representation is a balancing act. The complexity of the input data must be carefully matched with the learning algorithm's ability to process it without overfitting. Simply providing more data is not a guarantee of better performance.**

- **Increasing Robustness**: Future experiments must be made more robust by using additional random seeds, and developing more sophisticated model selection criteria. Overfitting should also be combatted.
- **Decreasing Noise:** Automated feature extraction techniques could also provide the agents with less noise, which should be investigated further.