# Summary of Replication

## 1. Introduction

- This report aims to replicate the methods and findings of the paper *'Deep Learning Statistical Arbitrage'* .
- The paper provides a comprehensive machine learning framework for constructing arbitrage portfolios using a convolutional transformer and deep learning techniques.
- The goal of this replication is to verify the methods on a smaller dataset and assess the model's performance on a selection of 10 stocks using the Fama-French factor model.

## 2. Methodology

### 2.1 Data

- For this replication, 10 stocks from the S&P 500 were selected using data obtained from *yfinance* spanning from 1982 to 2020.

  (The original paper used data from CRSP)

- Daily returns were calculated and adjusted by risk-free rates, alongside the Fama-French 3-factor data obtained from the Kenneth French Data Library.

### 2.2 Steps

- The methods of the original paper were applied, including the calculation of residuals from the Fama-French factor model and the use of a rolling window approach to generate arbitrage portfolios.
- Different models (including CNNTransformer, OUFFN, FourierFFN) was then employed to extract trading signals, followed by an analysis of returns with and without transaction costs.

#### 2.2.1 Data collection and residuals generation

- **Daily Data**: Collected daily adjusted closing prices.
- **Returns Calculation**: Converted closing prices to daily returns.
- **Excess Returns**: Adjusted returns by subtracting the risk-free rate to obtain excess returns.
- **Residuals Calculation**: Regressed the daily stock returns against the Fama-French factors. The residuals (unexplained returns) were then used as inputs for machine learning models to generate arbitrage signals.

#### 2.2.2 Model Training and Testing

- **Three different models** were implemented to extract signals from the residuals, and construct trading strategies:

  *OUFFN (Ornstein-Uhlenbeck Feedforward Network), CNNTransformer, and FourierFFN.*

- **Dataset**: 1000 days were used for training, while the remaining data was used for testing.
- **Conditions**: Models were optimized using two objectives: **Sharpe ratio** and **Mean-Variance**, with **cost** or **no cost**
- **Visualization**: Various plots, including *cumulative returns*, *turnover*, and *short positions*, were generated to visualize performance.

# 3. Results

The replication successfully produced trading signals and arbitrage portfolios, and the results were consistent with the original paper, although some deviations were observed, likely due to the smaller dataset used.

For this replication, Two sets of codes were used for training and testing. The first followed the original paper's approach, while the second incorporated **excess returns** to improve risk-adjusted ratios.

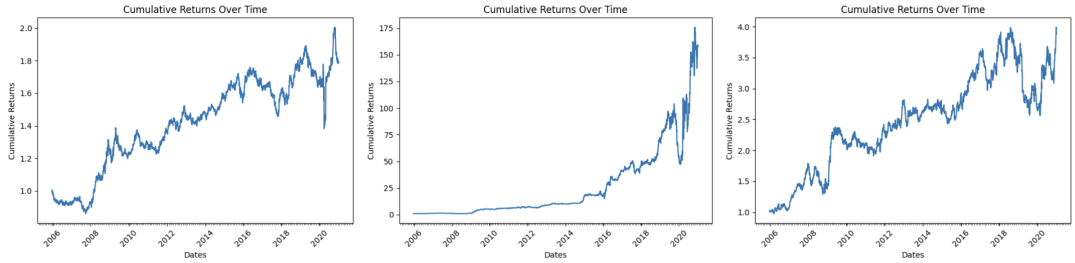
Besides, most models performed not very well in tests with cost conditions. As well, several models performed less effectively with mean-var objective.

## 3.1 Normal models performance:

The models generally performed well without costs, particularly in optimizing the Sharpe ratio.

- With no cost, factor 3, mean-var objective.
    - left: OUFFN, mid: FourierFFN, right: CNNTransformer


- With no cost, factor 3, sharp objective.
    - left: OUFFN, mid: FourierFFN, right: CNNTransformer



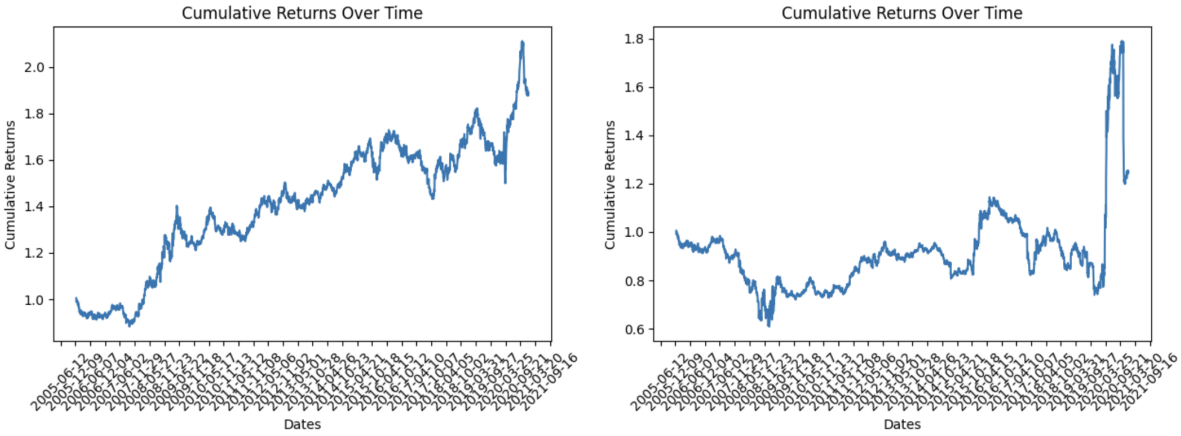

- With cost, factor 3, mean-var objective.
    - left: OUFFN, mid: FourierFFN, right: CNNTransformer


- With cost, factor 3, sharp objective.
    - left: OUFFN, mid: FourierFFN, right: CNNTransformer

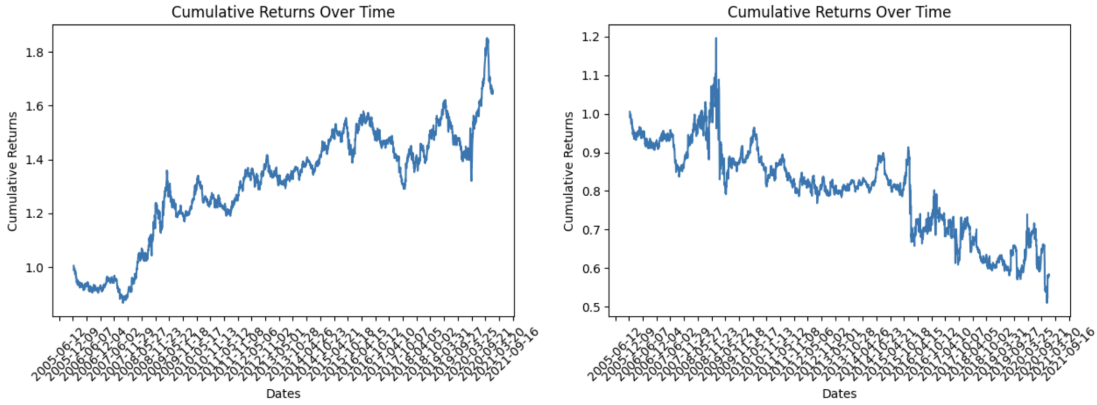

## 3.2 Additional models performance:

In this test,  the models only performed well when using OUFFN model to extract the signals, likely because of its different parametric structure.

- With no cost, factor 3, sharp objective.
    - left: OUFFN, right: FourierFFN,




- With cost, factor 3, sharp objective.
    - left: OUFFN, right: FourierFFN




- With cost, factor 1,mean-var objective.
    - left: OUFFN, mid: FourierFFN, right: CNNTransformer