TRADING WITH NEURAL NETWORKS: A LONG-SHORT APPROACH

Tomás Barbosa Romeiro
University College London
Student ID: 24022849
tomas.romeiro.24@ucl.ac.uk
COMP0162 - Advanced Machine Learning in Finance

March 18, 2025

1 Introduction

Predicting daily S&P 500 returns is particularly difficult due to non-linear dependencies and high-frequency noise inherent in market data. Recent machine learning advances that capture temporal and cross-sectional relationships show promise. This study draws on Fischer and Krauss [TF18], who demonstrated that LSTM networks can effectively forecast daily S&P 500 returns and generate profitable long–short strategies. We extend their work by applying similar and additional models, most notably CNNs, to a more recent, volatile period.

Following Fischer and Krauss's methodology, our dataset is divided into study periods with separate training and test sets. Stocks are selected daily based on the probability of outperforming the cross-sectional median, and long–short portfolios are constructed accordingly. While we replicate the LSTM architecture, we also integrate CNNs, whose effectiveness in forecasting was shown by Di Persio and Honchar [LDP16] and Du, Fernández-Reyes, and Barucca [BDB20], to enable direct performance comparison.

This report contributes to existing literature by evaluating the predictive performance of Logistic Regression, LSTM, and CNN models on S&P 500 returns and by analysing their potential to enhance investment strategies. We also compare key portfolio metrics to assess the robustness and economic relevance of these machine learning-based approaches

2 Methodology

Our methodology comprises five main stages, building on Fischer and Krauss [TF18] for data preparation and application of the LSTM model. First, we divide our dataset into rolling study periods, each containing a training set (in-sample) followed by a test set (out-of-sample). Second, we define our feature space and binary classification labels. Third, we implement an LSTM network, adhering as closely as possible to the aforementioned authors for comparability. Fourth, we introduce both a CNN and a wavelet-enhanced CNN (Wavelet-CNN), inspired by the aforementioned wavelet-based studies. Additionally, to benchmark our deep learning models against a more traditional approach, we implement a logistic regression model as a baseline. Finally, we formulate a daily long–short trading strategy that ranks stocks by each model's predicted probabilities that goes long the top 10 stocks ranked according to the predicted probability of over performing our target (cross-section of forecasted median next day return) and short the bottom 10 ranked ones. This section attempts to summarise the approach of replicated for our study period in terms of data preparation, sequence construction and implementation of the LSTM and Logistic Regression models. This is followed by an overview of CNN and Wavelet-CNN models and the specific architecture used in this report.

2.1 Study period, data preparation and training and test sets

Following Fischer and Krauss (2017) [TF18], we construct our dataset and perform predictions using rolling study periods, each composed of a training set for model estimation and a test set for out-of-sample predictions. Each period consists of a 750-day training period (approximately three years) followed by a 250-day trading period (approximately one year). Our dataset, covering daily stock returns of the constituents of the S&P 500, spans from early January 2014 to early December 2024, allowing for the creation of eight overlapping study periods. Table 1 displays descriptive statistics of the data set.

Industry	No. Stocks	Mean Return	Standard Deviation	Skewness	Kurtosis
Communication Services	20.8	0.58	5.54	-0.05	1.58
Consumer Discretionary	59.2	0.71	6.37	-0.63	4.80
Consumer Staples	36.9	0.55	3.63	-0.07	0.17
Energy	24.2	0.61	9.79	0.43	5.46
Financials	68.8	0.93	5.51	-0.53	1.88
Health Care	57.8	0.87	4.53	-0.17	-0.20
Industrials	69.2	1.01	5.26	-0.29	1.21
Information Technology	52.1	1.46	5.62	-0.28	0.22
Materials	24.5	0.81	5.61	-0.15	0.12
Real Estate	28.3	0.54	5.18	-0.46	2.00
Utilities	27.6	0.72	4.37	-0.40	0.17
All Industries	469.3	0.80	5.77	-0.05	6.32

Table 1: Average monthly summary statistics for S&P 500 constituents from January 2014 to December 2024. These are defined as per the Global Industry Classification Standards Code and are computed on equal-weighted portfolios, formed on a monthly basis using the reconstruction of the historical S&P components. Monthly returns and standard deviations are denoted in percent.

For each stock i, we construct sequences of length L = 240 trading days based on standardised returns. Each sequence is used to predict the stock's return direction one day ahead. Stocks available at the last day of the training period of each study period constitute the set of stocks n_i, typically close to 500 due to index membership changes. Stocks lacking sufficient historical price data are excluded for the specific dates, thus slightly reducing the actual number of sequences used in practice. The rolling approach follows Fischer and Krauss (2017) [TF18], allowing for robust training and validation, and ensuring comparability of results with previous literature.

Our goal is to forecast the next-day cross-sectional median-relative stock returns using historical daily returns. For each stock $i \in \{1, \ldots, N\}$ we define its daily return at day t as:

$$r_{i,t} = \frac{P_{i,t}}{P_{i,t-1}} - 1 \tag{1}$$

where $P_{i,t}$ denotes the dividend and split-adjusted closing price for stock i on day t.

Due to the required sequence length for feature generation, the first 240 days of each study period are reserved exclusively for constructing input features, as they are necessary to produce the first valid prediction on day 241.

For each stock, we then standardise the return sequences to predict a binary cross-sectional target defined as:

$$y_{i,t+1} = \begin{cases} 1, & r_{i,t+1} \geq \text{median}(r_{j,t+1}), \quad \forall j \in \text{stocks available at day } t+1 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

Sequences are standardised by subtracting the training-period mean return μ and dividing by its standard deviation σ:

$$\tilde{r}i, t = \frac{ri, t - \mu}{\sigma} \tag{3}$$

where μ and σ are computed exclusively from the training period so that look-ahead bias is avoided.

Each sequence covers approximately one trading year. These sequences are generated using a rolling-window approach, ensuring that for each stock, the earliest available sequence contains returns from day t-239 to t, the next sequence shifts forward by one day (covering t-238 to t+1), and so on. This overlapping structure provides a large number of training samples while preserving the temporal dependencies inherent in financial time series. The dataset is then split into in-sample sequences used for training and out-of-sample sequences for model evaluation and trading. This is illustrated in Figure 1 below.

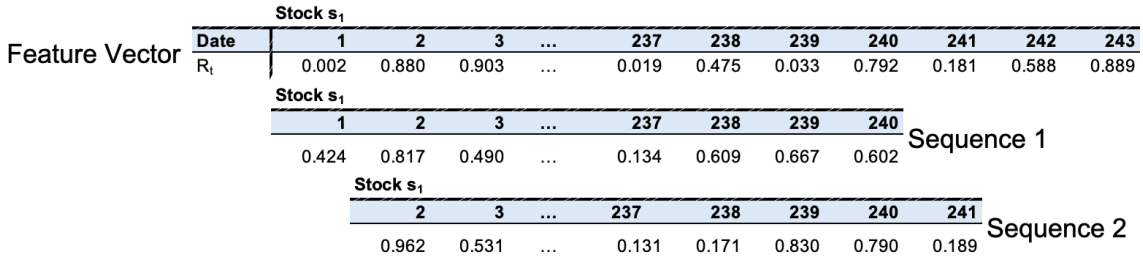

Figure 1: Illustration of the input sequences generated for the LTSM and CNN networks, analogous to the illustration in the original article.

Over the 1,000-day study period, given the chosen sequence length of 240 days and with approximately 500 stocks in the S&P 500, this results in a theoretical total of around 380,000 sequences. However, in practice, the actual number of sequences is slightly lower due to data quality filtering, which removes stocks with missing or unreliable data, as well as changes in the S&P 500's composition over time, where stocks are added or removed from the index.

2.2 Logistic Regression (baseline model)

We used Logistic regression as the baseline model used as a benchmark to the LSTM and CNNs. It predicts the probability that a given stock return on day t+1 will be above or below the median cross-sectional return, using lagged returns as input features (to mimic, as best as possible, the sequence construction methodology used for neural networks). The logistic regression model is formulated as:

$$p(y_{i,t+1} = 1 | x_{i,t}) = \frac{1}{1 + e^{-\left(\beta_0 + \sum_{l \in \mathcal{L}} \beta_l x_{i,t}^{(l)}\right)}} \tag{4}$$

where:

- $y_{i,t+1}$ is the binary target, $x_{i,t}^{(m)} = r_{i,t-m}$ the return at lag m.
- Lagged returns include $m \in \{1, 2, \ldots, 20\} \cup \{40, 60, \ldots, 240\}$
- Parameters are estimated by maximising likelihood (cross-entropy loss).

2.3 Long Short-Term Memory (LSTM) model

Long Short-Term Memory (LSTM) networks, introduced by Hochreiter and Schmidhuber [SH18], model temporal dependencies across sequential data. Each LSTM cell at time step t processes inputs as follows:

$$\begin{aligned} f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) & i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) & \tilde{c}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c) \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t & h_t &= o_t \odot \tanh(c_t) \end{aligned} \tag{5}$$

where:

- W_f, W_i, W_o, U_f, U_i, U_o are weight matrices and b_f, b_i, b_o, are bias vectors.

- x_t, is the input at time step t, h_t the hidden state, and c_t the cell state.
- f_t, is the forget gate, responsible for the choice of information to remove from memory.
- i_t, is the input gate, responsible for the choice of information to add to memory.
- o_t, is the output gate, responsible for the choice of information to use as output.

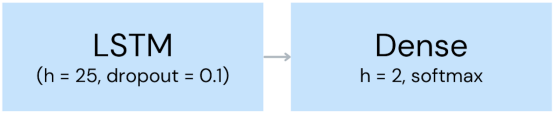

Figure 2: LSTM model architecture.

We use sequences of length $T = 240$ days, standardised by training-period returns statistics. The architecture proposed by Fischer and Krauss [TF18] is replicated and is as follows:

- An input layer accepting sequences of length L.
- A LSTM layer with 32 units, applying a dropout of 10%, and returning the full sequence for further processing.
- A *Dense* layer, activated by *softmax*, which outputs the binary probability prediction for the next-day return, where the loss function for training is sparse categorical cross-entropy:

$$\hat{y}_{i,t+1} = \text{softmax}(Wh_t + b) \tag{6}$$

The LSTM model is trained separately for each study period (with an in-sample cross validation of 20% of the training set), using a binary cross-entropy loss and *Adam* optimiser. An early stopping criterion based on validation loss (10 epochs with patience = 5) to avoid over-fitting. Ideally we would attempt a higher number of epochs, but due to computational and time constraints 10 was the number chosen. However, we have observed that the early stop occurred when running the model for a small number of periods. This is might be due to a combination of the noise associated with financial data of this type and the fairly simple architecture used.

2.4 Convolutional Neural Network models

Convolutional Neural Networks, introduced by LeCun and Bengio [LB97], extracts localised temporal features via 1-dimensional convolution filters, capturing shorter-range patterns within the time series:

$$h_t^{(j)} = f\left(\sum_{m=0}^{M-1} W_m^{(j)} x_{t-m} + b^{(j)}\right), \quad j = 1, \ldots, J \tag{7}$$

- Convolutional layers capture short-term patterns in returns, followed by pooling and dense layers to summarise sequence-level features.
- Model input is identical to the LSTM (sequences of length 240 standardised returns).

The sequences are fed as inputs into the CNN. The network structure comprises:

- An initial convolutional block with 32 filters (kernel size 3), activated by *ReLU*, followed by *Max Pooling*.
- A second convolutional block with 16 filters (kernel size 3), also followed by *Max Pooling*.
- A global average pooling layer condenses the extracted features into a fixed-size vector.
- Finally, a *Dense* layer, with dropout regularisation, preceding a *softmax*-activated output layer predicting the binary target (positive/negative returns).

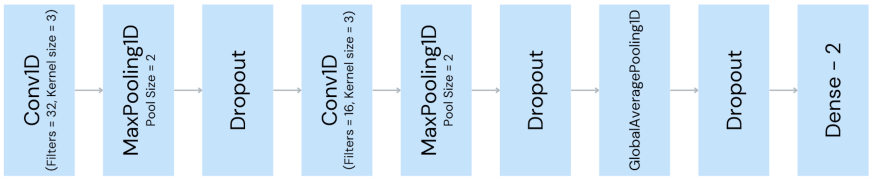

Figure 3: CNN Model architecture.

The CNN model is trained in the same manner as the LSTM model in terms of training validation split, number of epochs, patience, optimise and loss function.

2.4.1 Wavelet-CNN

Our Wavelet-CNN architecture, first proposed by Donoho and Johnstone [DJ94] extends the standard convolutional neural network by applying a wavelet transform to each return sequence before CNN processing. Our approach draws from the work of Di Persio and Honchar [LDP16] and Du, Reyes and Barucca [BDB20], who demonstrated that wavelet-based frequency decomposition can enhance predictive accuracy in financial forecasting, particularly by isolating relevant market signals from noise.

Financial time series are notoriously noisy, exhibiting high-frequency fluctuations that can obscure meaningful longer-term patterns. The discrete wavelet transform (DWT) decomposes a raw signal into approximation (low-frequency) and detail (high-frequency) components at various scales, capturing both short-term and longer-scale dependencies. By effectively separating noise from structured market movements, wavelets can produce clearer input representations for the CNN.

Instead of feeding raw 1D sequences, the DWT decomposes each sequence into multiple sub-bands, which are then padded or upsampled to form a 2D tensor. Given an input return sequence $\mathbf{x} = [x_1, x_2, \ldots, x_n]$, we obtain a set of wavelet coefficients:

$$W_\psi(j,k) = \frac{1}{\sqrt{2^j}} \sum_t x_t \psi\left(\frac{t - 2^j k}{2^j}\right) \tag{8}$$

where j and k represent scale and translation parameters, respectively, and $\psi(\cdot)$ denotes the mother wavelet function. The wavelet transform provides explicit time-frequency information, allowing the CNN to detect features that may be missed by standard time-domain convolutions. The resulting tensors are passed into a CNN architecture similar to our standard CNN, except the input layer is now 2D.

2.5 Data and software

The dataset comprises daily adjusted closing prices for S&P 500 constituents (3 January 2014–5 December 2025) from Compustat via WRDS, with historical constituent changes scraped from Wikipedia (since WRDS and free databases lack this information), and S&P 500 index prices from Yahoo Finance. Each stock's return series begins at its index entry (or 3 January 2014) and ends upon removal or at the final date; prices are cum-dividend and fully adjusted for corporate actions, and stocks with missing data are excluded.

Experiments were conducted in Python using Pandas, NumPy, and Scikit-learn for data management and logistic regression, and TensorFlow's Keras for LSTM and CNN models. Jupyter Notebooks were used for execution, with visualizations via Matplotlib and Seaborn and performance evaluation using Scikit-learn's metrics.

3 Results and Discussion

We first examine the classification quality of each model, standard metrics such as accuracy, precision, recall, and specificity, in order to understand how well the daily outperformance labels are predicted. Next, we assess the quantitative performance metrics, including mean daily return, Sharpe ratio, and drawdowns, to see how classification accuracy translates into actual profitability. Finally, we display

cumulative return plots to illustrate how each model's strategy would fare over the study period, providing an overall comparison of their practicality in a real trading context.

3.1 Classification metrics

The confusion matrix converts our binary classification, where a predicted probability above 50% indicates outperformance of the cross-sectional mean (which is sensible as stock returns are fundamentally random at short time frames), into counts of correct versus incorrect predictions in our daily long–short setting. We denote true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). From these, we compute:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN}, \qquad \text{Precision} = \frac{TP}{TP + FP},$$

$$\text{Recall} = \frac{TP}{TP + FN}, \qquad \text{Specificity} = \frac{TN}{TN + FP}, \qquad \text{F}_1 = \frac{2\,(\text{Precision})(\text{Recall})}{\text{Precision} + \text{Recall}}.$$

	Accuracy	Precision	Recall	Specificity	F1 Score
Logistic Regression	0.499	0.499	0.512	0.485	0.506
LSTM	0.501	0.501	0.476	0.523	0.489
CNN	0.500	0.501	0.500	0.501	0.501
Wavelet-CNN	0.500	0.500	0.460	0.540	0.479

Table 2: Evaluation metrics across models.

Table 2 shows all models have similar performance (around 50%), which is expected in a noisy, near-efficient market where predicting daily outperformance is nearly random. However, even small differences in stock rankings, especially for the top and bottom 10, can lead to markedly different portfolio constructions and trading outcomes. Overall metrics may mask variations in accuracy for these key subsets, ultimately driving profitability.

Figure 4 shows that predicted probabilities reveal differences in model confidence that overall metrics do not capture. The LSTM assigns higher probabilities for long trades (0.53–0.58) and lower for shorts, indicating strong conviction, while the CNN remains near the decision boundary for both, suggesting near-random selection. Logistic regression and wavelet-CNN are relatively neutral (longs 0.52–0.54, shorts 0.45–0.47), with the wavelet-CNN slightly more variable on the short side. Thus, despite similar overall metrics, these confidence differences could lead to varying trading outcomes.

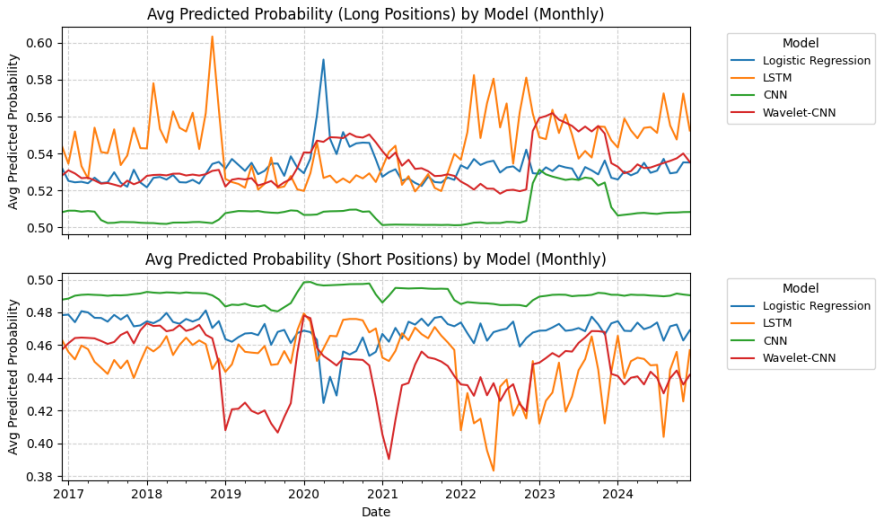

Figure 4: Monthly average predicted probability that a stock will outperform the cross sectional median return, for trades executed by each model.

3.2 Performance analysis of the trading strategy

The trading strategy employed in this study is a daily long–short approach based on cross-sectional stock return predictions. Each trading day, the model ranks stocks according to their predicted probability of outperforming the median return of all stocks in the universe. The top 10 stocks with the highest predicted probabilities are selected for long positions, while the 10 stocks with the lowest predicted probabilities are shorted. Positions are held for one day before rebalancing, ensuring that the portfolio dynamically adjusts to new predictions each trading session.

We assess the strategy by first evaluating key financial metrics: daily return distributions, risk, and annualized risk-adjusted performance (e.g. Sharpe and Sortino ratios). Next, we plot cumulative returns over the study period to gauge consistency and relative model performance under varying market conditions. Finally, we break down results by industry to determine whether profitability is driven by specific sectors or broadly distributed. A comprehensive breakdown of the performance results of the strategy for every model is displayed in Fig. 5 below.

	Logistic Regression	LSTM	CNN	Wavelet-CNN
Mean Return (Long) %	0.057	0.079	0.024	0.013
Mean Return (Short) %	-0.066	-0.013	-0.058	-0.044
Mean Return (Net) %	-0.009	0.066	-0.035	-0.031
Standard Error %	0.039	0.033	0.032	0.031
T-Statistic	-0.238	2.023	-1.072	-1.001
Min %	-16.439	-9.864	-10.349	-13.156
Q1 %	-0.796	-0.725	-0.738	-0.759
Median %	-0.008	0.009	0.02	-0.038
Q3 %	0.771	0.819	0.756	0.724
Max %	13.819	8.604	7.683	7.996
Share > 0 %	49.65	50.4	50.9	48.6
Std Dev %	1.757	1.468	1.447	1.403

	Logistic Regression	LSTM	CNN	Wavelet-CNN
Skewness	-0.227	0.381	-0.635	-0.379
Kurtosis	10.622	4.923	4.507	7.173
1% VaR	-5.027	-3.893	-4.375	-3.709
1% CVaR	-7.105	-4.967	-5.874	-5.457
5% VaR	-2.529	-2.191	-2.468	-2.185
5% CVaR	-4.17	-3.161	-3.656	-3.304
Max Drawdown %	-66.386	-34.417	-68.559	-76.46
Return % p.a.	-2.353	16.739	-8.738	-7.916
Std Dev % p.a.	27.898	23.307	22.97	22.276
Downside Dev % p.a.	21.193	15.115	17.865	16.256
Sharpe p.a.	-0.084	0.718	-0.38	-0.355
Sortino p.a.	-0.111	1.107	-0.489	-0.487

Figure 5: Performance characteristics of the portfolio for all models, including daily return / risk characteristics and annualized risk-return metrics.

The performance summary table highlights significant differences across models in terms of returns, risk, and overall efficiency. The LSTM model stands out as the strongest performer, achieving the highest net return (6.6%) and annualised return (16.74%), while also maintaining a relatively moderate risk profile compared to other models. Below is a more detailed review.

3.2.1 Return Metrics

The LSTM model dominates, achieving a mean net daily return of 0.066% and an annualised return of 16.74%, substantially outperforming all other models. In contrast, both the CNN and Wavelet-CNN models deliver negative annualised returns of -8.74% and -7.92%, respectively, while Logistic Regression falls in between with an annualised return of -2.35%.

3.2.2 Risk and Volatility

Standard deviation (risk) is relatively similar across models, but the LSTM model uniquely achieves higher returns per unit of risk with a positive annualised return profile. Maximum drawdowns are steep for all models; however, the LSTM's drawdown of -34.42% is substantially lower than those of the CNN (-68.56%), Wavelet-CNN (-76.46%), and Logistic Regression (-66.39%). Additionally, VaR and CVaR metrics indicate that the LSTM exhibits lower tail risk compared to the others, although all models perform fairly similarly on these measures.

3.2.3 Risk-Adjusted Performance

The LSTM model achieves 0.72, the only positive Sharpe ratio, indicating that it generates a reasonable return per unit of risk. In contrast, Logistic Regression (-0.08), CNN (-0.38), and Wavelet-CNN (-0.36) all perform worse than a risk-free asset. In terms of Sortino Ratio, the LSTM again stands out (1.11), meaning it delivers positive excess return despite downside risk, whereas all other models have negative Sortino ratios.

Considering the analysis above, the LSTM significantly outperforms the other models in both absolute and risk-adjusted returns. However, it still experiences substantial drawdowns, which could be problematic in a real-world trading strategy. Due to the significant outperformance, we will focus on the LSTM in more detail in the following section.

3.3 Review of the LSTM based Strategy Performance

In this section, we delve deeper into the behaviour and performance of the LSTM-based strategy, comparing its cumulative returns to those of the S&P 500 over the full study period. Fig. 6 below highlights the extent to which the LSTM model navigates major market fluctuations, shedding light on its drawdown patterns and overall risk–return efficiency relative to the benchmark and other models.

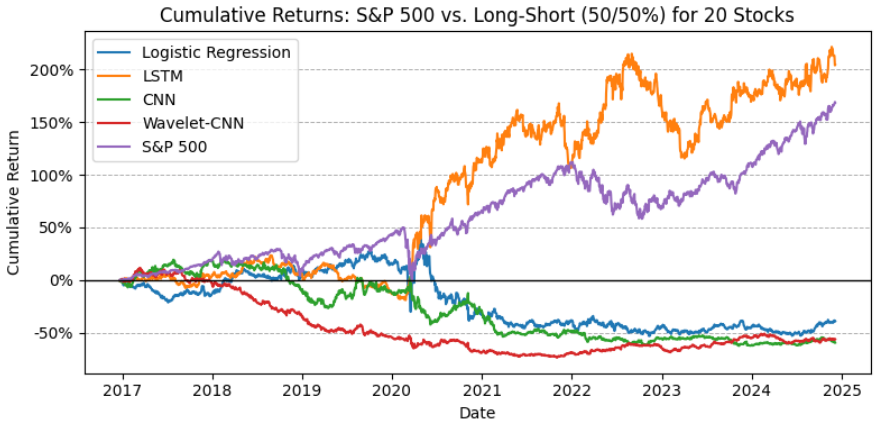

Figure 6: Comparison of the cumulative returns of the Long-Short, daily strategy for the different models and the S&P 500, across the entire study period.

While the S&P 500 suffered steep drawdowns during the COVID-19 crash (2020) and the 2022 bear market, the LSTM strategy experienced milder dips, recovered sooner, and eventually surpassed the index. Notably, although the LSTM outperformed in 2022, it briefly reversed gains before the bear market, possibly due to wrongly anticipating a counter-trend. In contrast, other models, especially the Wavelet-CNN, struggled and eventually locked in an approximate -50% drawdown each. This suggests that the LSTM more effectively identifies short-term patterns and rotates out of underperforming stocks during volatile periods.

Table 3 shows the trade volume distribution across sectors, providing essential context before an analysis of industry specific cumulative returns can be made. These must be interpreted relative to each industry's portfolio weight. For instance, although the cumulative returns in the Energy sector were over 1,300% at one point in 2023, its overall impact is limited by its small allocation (around 4-5% of total trades). This underscores the need to assess both absolute performance and relative exposure when evaluating final returns.

Industry	Share of Longs (%)	Share of Shorts (%)
Communication Services	6.05	6.16
Consumer Discretionary	18.96	20.12
Consumer Staples	6.27	5.63
Energy	9.24	9.57
Financials	9.61	8.14
Health Care	11.81	11.81
Industrials	10.43	9.80
Information Technology	14.12	16.63
Materials	6.51	6.51
Real Estate	4.13	3.36
Utilities	2.81	2.25
Total	100.00	100.00

Table 3: Industry representation in total number of trades, per position type.

Figure 7 shows that the LSTM strategy's outperformance is driven by key sector trends. Notably, the Energy sector is an extreme outlier and the primary contributor from 2020 onward, coinciding with COVID–19 market dislocations that caused sharp declines in energy demand and asset prices, followed by a robust recovery. Additionally, Industrials and Real Estate (perhaps due to the low interest rate environment affecting demand for housing [GO21]) delivered some gains in 2020 and 2021, further boosting cumulative returns.

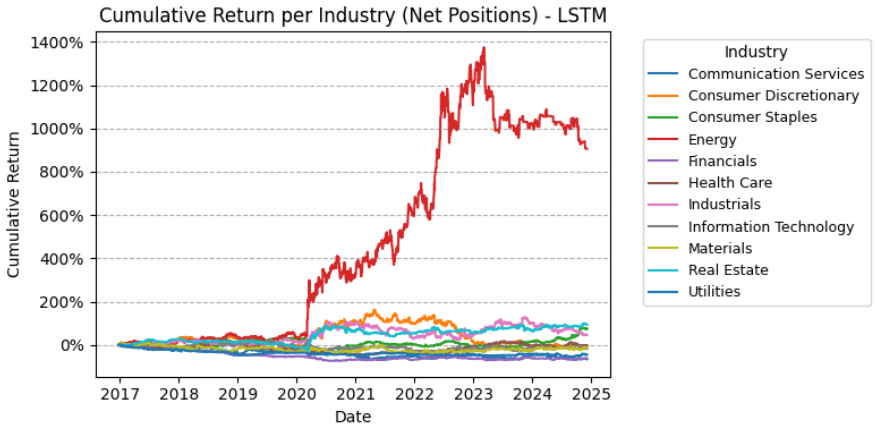

Figure 7: Industry-level breakdown of cumulative net returns of the Long-Short LSTM across the study period.

When compared to the industry-wide S&P 500 performance (see Figure 8), it is evident that the LSTM model capitalised on counter-trend opportunities. The strategy's focus on Energy coincided with its market downturn and subsequent rebound, suggesting that the model effectively identified and exploited temporary mispricing. Interestingly, while the Information Technology sector was the best-performing industry in the S&P 500 over the past five years, the LSTM model did not benefit from this over performance, reinforcing its counter-trend tendencies.

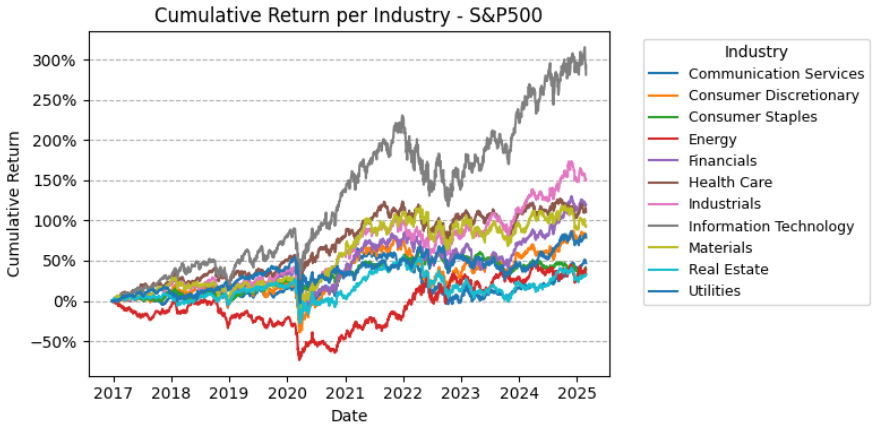

Figure 8: Industry-level breakdown of cumulative S&P500 returns across the study period.

A more granular analysis of the strategy's position types provides additional clarity. On the long side (Figure 9), the LSTM model found returns in Information Technology alongside the index (but below we will see it also tried the opposite side, unsuccessfully) and on Energy and Industrials.

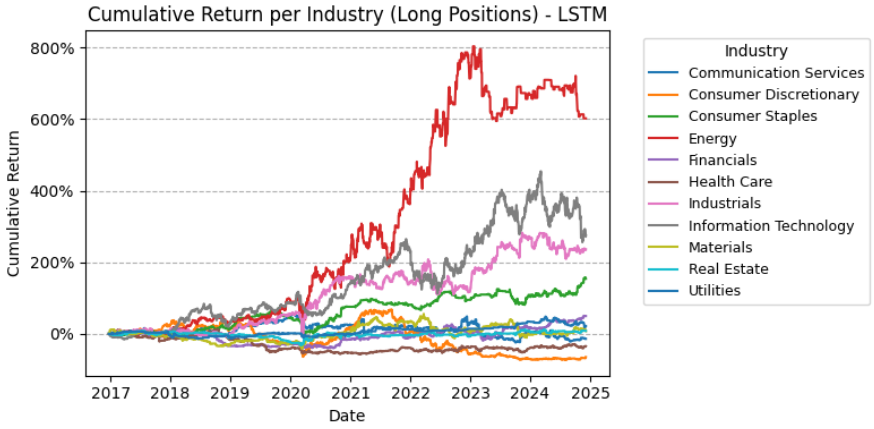

Figure 9: Long-side industry-level breakdown of cumulative returns of the Long-Short LSTM across the study period.

Conversely, the short positions (Figure 10) reveal a different dynamic. The strategy generated significant profits in early 2020 by shorting Energy stocks, capitalising on the collapse in oil prices. However, a portion of these gains was eroded over time as the sector rebounded. The Real Estate sector also contributed significantly to short-side performance, with sustained gains that mirrored the overall trend of distress in commercial property markets due to the COVID-19 pandemic [CP22]. In contrast, shorting Information Technology and Financials stocks resulted in substantial drawdowns, aligning with the long-term bull market in the former, which accelerated due the commercialisation of AI related products in late 2022 and the bounce back in the later, as the banking sector's earnings in late 2023 and 2024 started to reflect the rise in net income margins associated with the increases in overall interest rates [AGH24]. In conclusion, while the LSTM strategy outperformed the market and anticipated reversals, much of its success was driven by a highly volatile sector.

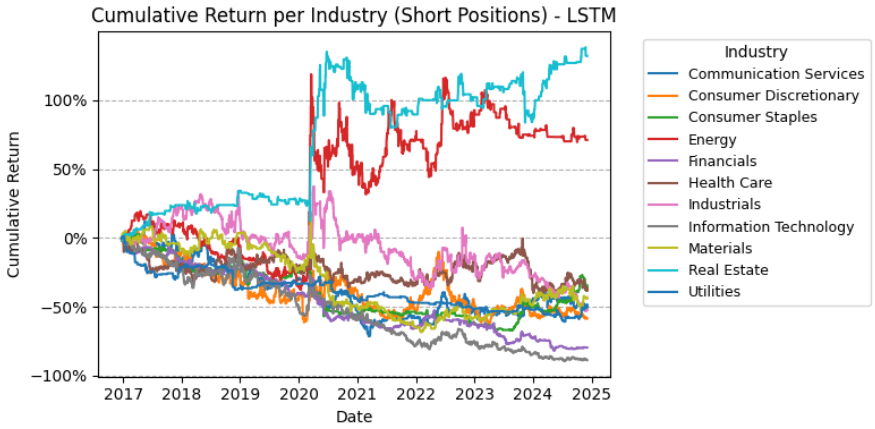

Figure 10: Short-side industry-level breakdown of cumulative returns of the Long-Short LSTM across the study period.

3.4 Final Remarks

While the LSTM results are promising, they must be viewed in context with several methodological limitations. The chosen architectures, though effective in some financial tasks, may be too simplistic to capture complex market dynamics; more advanced layer structures, especially for non-LSTM models, might boost performance but would demand greater computational resources and careful optimization. Additionally, due to time and resource constraints, we did not perform exhaustive hyper-parameter tuning, which may have limited the models' potential given the high dimensionality and noise in financial data.

In real-world trading, frequent retraining is vital to adapt to evolving market conditions. Although our study periods balance historical context with generalizability, long training windows can "cement" models in outdated trends, a phenomenon known as concept drift, since market regimes can shift within months. Training on data from multiple regimes may dilute critical signals and cause models to miss sharp shifts, while fixed multi-year windows risk over-fitting to past trends. Shorter or rolling windows can better capture recent dynamics and improve responsiveness to sudden anomalies such as earnings surprises or policy changes.

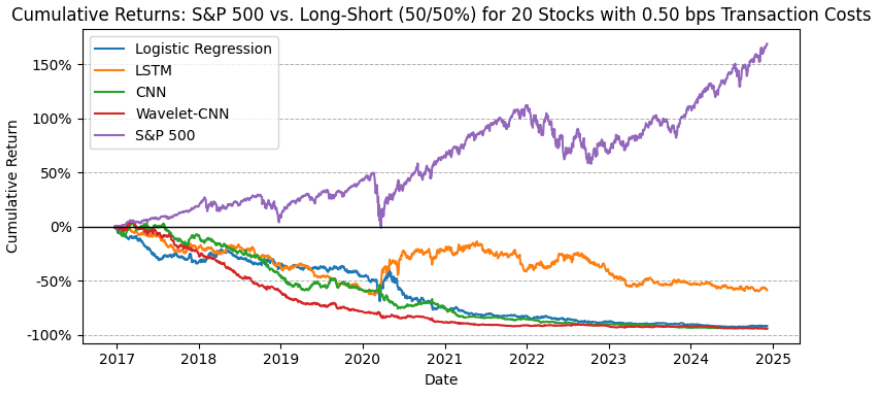

Figure 11: Cumulative returns with transaction costs of 0.5 bps

Finally, returns were calculated based on closing prices, which may not capture real-world dynamics like bid-ask spreads, liquidity, and transaction costs. These factors could significantly erode returns, especially in strategies that rebalance often as Figure 11 shows. Adopting lower-frequency rebalancing, integrating transaction costs or turnover constraints, and using execution-aware trading can help bridge this gap.

References

[AGH24] Marukel Nunez Maxwell Max Flötotto Oskar Skau Amit Garg, Marti Riba and Matic Hudournik. The state of retail banking: Profitability and growth in the era of digital and ai. Technical report, McKinsey & Company, 2024.

[BDB20] D. Fernandez-Reyes B. Du and P. Barucca. Image processing tools for financial time series classification. 2020.

[CP22] James Chong and G. Michael Phillips. Covid-19 losses to the real estate market: An equity analysis. *Finance Research Letters*, 45, 2022.

[DJ94] David L Donoho and Jain M Johnstone. Ideal spatial adaptation by wavelet shrinkage. *biometrika*, 81:425–455, 1994.

[GO21] Judit Montoriol Garriga and Pedro Alvarez Ondina. The impact of the pandemic on international housing markets: is there a risk of overheating? Technical report, CaixaBank, 2021.

[LB97] Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time-series. 1997.

[LDP16] Oleksandr Honchar Luca Di Persio. Artificial neural networks architectures for stock price prediction: comparisons and applications. *International Journal of Circuits, Systems and Signal Processing*, 10, 2016.

[SH18] Jürgen Schmidhuber Sepp Hochreiter. Long short-term memory. *Neural computation*, 9:1735–1780, 2018.

[TF18] Christopher Kraus Thomas Fischer. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270:654–669, 2018.