28-10-2025

003 Advanced Trading Strategies: Deep Learning

MARKET MICROSTRUCTURE AND TRADING SYSTEMS

José Armando Melchor Soto
Rolando Fortanell Canedo
ITESO

Contents

Introduction

This project focuses on the development of a systematic trading strategy using deep learning models trained on engineered time series features. The objective is to design, implement, and evaluate a quantitative trading framework capable of predicting market signals — long, short, or short, or hold through the integration of multiple neural network architectures. The analysis will be conducted using a dataset containing 15 years of daily price data for the selected financial asset, allowing for the identification of long-term market behaviors and regime shifts.

The strategy will incorporate three deep learning architectures: MLP, CNN, and LSTM, each designed to capture distinct aspects of temporal and nonlinear dependencies in financial data. A comprehensive feature engineering process will be conducted, combining momentum, volatility, and volume indicators across multiple periods. All features will be normalized, and class imbalance will be addressed through weighting techniques to enhance model fairness and generalization.

MLFlow will be used for experiment tracking, ensuring systematic management of model versions, hyperparameters, and performance metrics. Additionally, data drift analysis will be performed to detect significant changes in the distribution of input variables over time, providing insight into market regime changes or evolving dynamics.

The developed models will be evaluated through a Backtesting framework that integrates realistic trading conditions, including transaction costs, position sizing, and dynamic trade management. The final evaluation will consider both predictive and financial performance metrics such as Accuracy, F1-Score, Sharpe Ratio, Sortino Ratio, Calmar Ratio, Maximum Drawdown, and Win Rate. Overall, this project aims to establish a robust and data-driven methodology for applying deep learning techniques to systematic trading in financial markets.

Asset Selection

Asset

APPLE **(AAPL)**

Economic Sector

Information Technology

Industry

Consumer Electronics

Company Overview

Apple Inc. (AAPL) is a leading multinational technology company that designs, manufactures, and markets consumer electronics, software, and digital services. Its ecosystem, including the iPhone, iPad, Mac, and App Store generates recurrent revenues and high operating margins, making AAPL a key component of global equity indices. The company's strong fundamentals, high liquidity, and predictable market behavior make it an ideal asset for deep learning–based systematic trading, providing sufficient historical depth and stable signal-to-noise patterns in price movements.

Strategy Overview

This strategy employs deep learning models with the goal of predicting trading signals (buy, sell, or hold) using historical market data. One of the main advantages of this approach is that deep learning models can automatically learn complex patterns that humans or simpler models are unable to identify or may overlook during analysis.

To improve the model's performance, different neural network architectures are used, as each one captures different aspects of market behavior. Since market movements are highly nonlinear and exhibit subtle, hard to detect patterns, a single model would not be sufficient to capture all the existing complexity. By combining multiple models, the accuracy of predictions increases. In this project, the following architectures are employed:

• **MLP (Multilayer Perceptron):** This model corresponds to a neural network composed of multiple layers of interconnected neurons. Each neuron receives information from the previous layer and transforms it through nonlinear activation functions. This allows the model to learn complex relationships between market variables. However, one of its main limitations is that it treats input data as static, making it difficult to detect temporal dependencies in time series, such as those found in financial data.

• **CNN (Convolutional Neural Network):** This model applies convolutional filters that move over the input data to detect local patterns. In the context of trading, it allows the identification of how prices and indicators change over short time intervals, detecting relevant events such as breakouts, trend reversals, or periods of high volatility. In other words, this model treats the price sequence as a structured signal, efficiently extracting spatial and temporal information.

By combining both models within the strategy, the CNN acts as a feature extractor that identifies short-term patterns, while the MLP uses those patterns to decide the optimal action buy, sell, or hold according to market context. In this way, both models complement each other to improve the accuracy and robustness of predictions.

As mentioned earlier, the use of deep learning in trading strategies offers major advantages, as it allows models to automatically learn nonlinear and complex market behaviors, as well as relationships that evolve over time. Additionally, these models can leverage large volumes of data, handling information more effectively than traditional models.

However, there are also significant limitations. Deep learning models are prone to overfitting, especially when the data contain a lot of noise or when the training set is limited. They also require high computational costs for both training and implementation, and their “black box” nature makes it difficult to interpret the model’s decisions an additional challenge in the financial domain.

Feature Engineering

For this strategy, three main types of indicators were used: momentum, volatility, and volume-based indicators. Each group serves a distinct purpose in analyzing market behavior and provides complementary information for trading decision-making.

Momentum indicators measure the speed at which an asset's price moves over time and determine whether that movement is gaining or losing strength. In other words, they help identify the acceleration or deceleration of a trend. These indicators provide insight into the market's momentum dynamics, showing whether buyers or sellers are in control. The momentum indicators used were the following:

Momentum

• RSI (Relative Strength Index): measures the speed and magnitude of recent price movements to detect potential overbought or oversold conditions

For this indicator 4 different windows are used, 7,10, 14 and 20, for example the rsi window with 7 is the smallest window of the four, it is more sensitive and reacts faster to price changes, it also gives earlier signals, the largest window is the one with 20, it is smoother, reacts slower, and is better for detecting long term trends, with the window as 14 it is more of a medium term one, and it is the most common to use, since it is not that sensitive, but not as smooth.

• Awesome Oscillator: evaluates market momentum by comparing a 34-period simple moving average with a 5-period one, helping to detect trend reversals.

For the Awesome Oscillator, window1 is shorter and window2 is longer, the difference between these two averages measures momentum shifts, smaller windows have mor sensitivity, and large windows, are smoother with fewer reversals.

• Williams %R: identifies whether an asset is overbought or oversold, with values ranging from 0 to -100. A value between 0 and -20 suggests overbought conditions, while between 80 and -100 indicates oversold conditions.

When using Williams %R, there's an specific lookback period to use or how many previous candles to use, it is typically 14 periods, but when using smaller periods the indicator reacts faster which is good for short-term trades but with larger periods the response is slower, but better for long term trades.

• ROC (Rate of Change): calculates the percentage change in price relative to a specified number of previous periods, helping to detect trends and extreme market levels.

In ROC the window placed are the number of periods used to compare the current price to the price in the past, for example a roc window of 10, compares the price now to 10 periods ago, smaller windows are mor responsive to short-term changes, and larger windows are better for trend detection but lag more.

• Stochastic Oscillator: compares the closing price with its price range over a given period, allowing observation of trend strength and potential reversal points.

In the Stochastic Oscillator the window is the number of periods to find the highest high and the lowest low which is usually 14, the smooth window refers to how smooth the %K line is, which usually is 3, with a smaller window there's faster reaction, but more noise, and a larger window is slower, but with smoother movemets, smoothing helps reduce false crossovers and improve readability

Volatility

Volatility indicators, on the other hand, measure the variation in price over a period. Their goal is to quantify how quickly prices fluctuate and to provide information about the market's level of uncertainty. High volatility indicates rapid and unpredictable movements, while low volatility reflects greater stability. These indicators help

anticipate whether the market is entering a calm or more turbulent phase. The volatility indicators used were the following:

• Average True Range (ATR): measures the market's overall volatility by considering daily price ranges. A high ATR value indicates a more volatile market, while a low value suggests more stable movements.

The ATR uses a number of window which is the number of periods used to compute the average volatility, the default ATR window is a 14 period which averages the "true range" over 14 candles, a smaller window reacts quickly to volatiliy spikes, and a larger one is smoother, and tracks long term volatility.

• Bollinger Bands: use upper and lower bands that expand or contract depending on volatility. Wider bands imply higher volatility, while narrower bands reflect stability.

With Bollinger Bands you have the bollinger window which is the lookback period for the moving average, and it defines how many candles are used to calculate de central mean, a shorter window reacts faster to price changes and a longer one smoother but is less responsive, then you have the bollinger dev which is the number of standard deviations used to set the width of the bands, when the value is higher the wider the bands are, and when the value is lower, the narrower the bands are.

• Donchian Channel: shows the highest and lowest prices reached over a specific period, helping to identify trends and possible entry or exit points.

For the Donchian Channel the window is the number of periods to find the highest high and the lowest low, if the specified window is 10 then the channel will show the high and low over the last 10 candles, a smaller value means the channel adapts faster to price changes, and a larger value is slower, which is better for long term trades.

• Keltner Channel consists of a central moving average and two lines placed above and below it. The channel's slope indicates the trend direction: upward when the channel points up, and downward when it points down.

The window in the Keltner channel is the EMA period for the channel's middle line, the Keltner ATR is used for the upper and lower bands, the higher the multiplier means the wider the channel, which has fewer breakout signals, and the lower the multiplier, the tighter the channel, more signals but more noise

Volume

Finally, volume-based indicators analyze the number of transactions executed within a specific period. These reflect the level of market interest and participation, which allows the assessment of the strength of price movements. In general, a movement accompanied by high volume is more reliable than one with low volume, as it implies greater investor participation. The volume indicators used were the following:

• On Balance Volume (OBV): measures cumulative changes in volume to anticipate price movements. If volume increases without a significant change in price, it can be interpreted as a potential signal of a strong upcoming move.

• Chaikin Money Flow (CMF): evaluates the money flow over a period of time. Its values range from -1 to 1, where values close to 1 indicate strong buying pressure, and values close to -1 indicate strong selling pressure.

The CMF window is the number of periods to average the money flow over, a smaller window means it captures short term buying/selling pressure, and a larger window is smoother, with long term capital flow trend.

• Accumulation/Distribution Index (A/D): combines price and volume to determine whether money is flowing into or out of an asset, helping to confirm the strength of a trend.

• Volume Price Trend (VPT): accumulates volume changes weighted by the percentage change in price. It indicates the direction and intensity of market movements, providing insight into the overall flow of price and volume.

Taken together, these three groups of indicators provide a more comprehensive view of market behavior, combining information about the strength of movements, price stability, and investor participation.

Normalization

Normalization and feature scaling is very important for this strategy, these are preprocessing step so the variables selected or in this case the indicators can contribute proportionally to the model, when working with different indicators each can have different numerical ranges, for example: RSI ranges from 0 to 100, ATR or OBV can take large unbounded values, CMF or Williams can ve negative, and price-based features might be in the hundreds or the thousands, if these are used like this, in can cause biased learning, so the model might interpret a higher magnitude feature as more important.

Class imbalance strategy (class weighting explanation)

Usually in trading, signal classification is imbalanced, for example, there's 80% hold, 10% long and 10% short, this imbalance can usually cause the model to predict the majority class most of the time, to fix this a strategy is applied using threshold adjustment. To do this an alpha is manually changed instead of using a default one, to classify between these three decisions. The horizon used is a 10 day period, since the data from the dataset is daily, there wouldn't be a lot of movement between for the model to identify trades if a shorter horizon is used, the alpha cannot be high because there's not a lot of drastic movement between days.

Target Variable Definition

Trading signals:

- Long (1): A long signal represents a buy, basically expecting the asset's price to increase soon, it implies that the future returns are predicted to be positive. When the model indicates a long signal, means the momentum is positive, that

it has a strong upward trend, indicators show accumulation or buying pressure, and there's a breakout above a resistance level, basically betting on price appreciation

- Hold (0): A hold signals means staying out of the market or maintaining the current position, basically what the model is telling you is that there's not a clear signal, this avoids overtrading and helps preserve capital during flat or noisy markets, usually traders hold when the market is moving with no strong trend, the volatility is low and not worth the risk, and indicators are mixed.
- Short (-1): A short position means you're selling an asset you don't own, looking to buy it back later at a lower price, when your model gives a short signal, it's basically telling you to sell or go short because the market looks bearish, traders usually place short positions when momentum turns negative, there's clear selling pressure or breakdown below support, and the market sentiment turns bearish

Threshold holding for class assignment

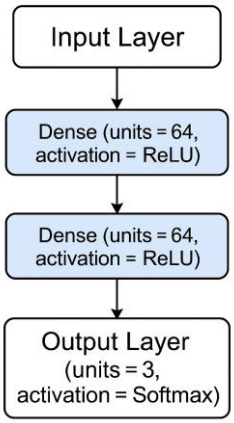
In this trading strategy, a 1% threshold is used to classify trading signals based on future price changes. If the future price is at least 1% higher than the current price, it indicates a long position, while a drop of 1% or more signals a short position. When the price change is smaller than 1% in either direction, it is considered a hold signal, suggesting no significant movement. This threshold helps filter out minor fluctuations and focuses only on meaningful market trends, ensuring trades are made when the potential return justifies the associated risk and transaction costs.

Model Architecture and Design

MLP

The MLP model is a fully connected feed-forward neural network designed to capture nonlinear relationships between input features. The architecture consists of an input layer that receives the feature vector, followed by multiple dense layers with ReLU activation functions, and a final output layer that produces class probabilities.

Architecture:



- Input Layer: The input layer receives the preprocessed data from the asset, with the indicators included, momentum, volatility and volume, and the closing prices
- Hidden Layers**:** The only layer applies a linear transformation followed by a nonlinear activation (ReLU). These layers learn high-level representations of the input features.
- Output Layer: Produces a probability distribution over three classes (long, hold, short) using the Softmax activation function

Training Parameters:

MLP Params		
Parameter	**Type**	**Value**
dense_layers	int	2
dens_units	int	128
activation	str	"relu"
optimizer	str	"adam"
output_units	int	3
output_activation	str	"softmax"
loss	str	"sparse_categorical_crossentropy"
metrics	tuple	"accuracy"
bathc_size	int	32
epochs	int	100
verbose	int	2
Average	str	"weighted"

The network applies convolutional filters to extract features at different spatial scales, followed by pooling operations to reduce dimensionality and prevent overfitting.

MLP

We use a 1D sequential CNN over lookback windows of 20-time steps per sample. The feature extractor stacks conv_layers=2 identical blocks; each block applies Conv1D (filters, kernel_size=3, activation=relu, padding="same") followed by MaxPooling1D (pool size=2). Filters start at 32 and double after each block, trading temporal resolution for channel depth. The stride-2 pooling halves the sequence length per block and expands the effective receptive field: with two blocks the model "sees" ~16 original time steps (RF progression: 3 → 6 → 8 → 16), which is close to the entire 20-step window and helps capture short-to-medium-range temporal patterns without exploding parameters. Features are flattened and passed to a dense head Dense (64, relu) and a Dense (3, SoftMax) for the three classes (short/hold/long). Training uses Adam, sparse_categorical_crossentropy, batch size 252 (≈ one trading year), epochs=10, and reports accuracy (weighted averaging downstream).

CNN

Training Parameters:

CNN Params		
Parameter	**Type**	**Value**
lookback	int	20
conv_layers	int	2
filters	int	64
kernel_size	int	3
dens_units	int	64
activation	str	"relu"
optimizer	str	"adam"
output_units	int	3
output_activation	str	"softmax"
loss	str	"sparse_categorical_crossentropy"
metrics	tuple	"accuracy"
bathc_size	int	128
epochs	int	60
verbose	int	2
Average	str	"weighted"

MLFlow Experiment Tracking

Structure

The experiment was titled “Advanced-Trading-Strategies”, within which the MLP and CNN models were executed independently, each following its own predefined set of hyperparameters.
Each training session was recorded as an individual run within the experiment, enabling structured tracking of metrics, parameters, and artifacts through MLFlow.

The models were identified as MLP_Model_003 and CNN_Model_003 and were managed through internal functions that allow retrieving the latest version of each model for integration into the main pipeline. Additionally, the system automatically links the model’s name and version registered in the MLFlow Model Registry, using the URL provided by the platform.

During training, automatic logging was enabled to capture key metrics, parameters, and performance curves, complemented by manual logging for specific metrics such as the F1-score. In this way, each model version is stored together with its metadata, configuration, and performance results, ensuring complete traceability throughout the entire experimental lifecycle.

Model Comparison

Run Details

Run Name:	CNN_layers2_filters64	MLP_layers2_units128
Start Time:	10/28/2025, 04:14:58 PM	10/28/2025, 04:14:41 PM
End Time:	10/28/2025, 04:15:18 PM	10/28/2025, 04:14:58 PM
Duration:	20.3s	16.8s

Parameters

batch_size	252	252
class_weight	{0: 1.0105166895290352, 1: 2.0928030303030303, 2: 0.6524948331857101}	{0: 1.0105166895290352, 1: 2.0928030303030303, 2: 0.6524948331857101}
epochs	60	100
initial_epoch	0	0
opt_amsgrad	False	False
opt_beta_1	0.9	0.9
opt_beta_2	0.999	0.999
opt_clipnorm	None	None
opt_clipvalue	None	None
opt_ema_momentum	0.99	0.99

Metrics

val_accuracy	0.326	0.545
val_f1	0.16	0.385
val_loss	1687439500378112	1730118456180736

During training, the MLP model demonstrated faster convergence and superior overall performance compared to the CNN model.

Although both models used the same batch size of 252 (corresponding to the number of trading days), the CNN was trained for only 60 epochs, whereas the MLP ran for 100 epochs, which likely contributed to its more stable learning and better generalization.

Quantitatively, the MLP achieved higher validation accuracy (0.545 vs 0.326) and a better F1-score (0.385 vs 0.16), alongside a lower validation loss, indicating a more effective capture of underlying market patterns.

In contrast, the CNN model required more computational time (20.3s vs 16.8s) and exhibited weaker validation metrics, possibly due to underfitting caused by the reduced number of epochs or insufficient temporal context (lookback window). The MLP model outperformed the CNN in both speed and predictive performance under the tested configuration. Future experiments could focus on extending the CNN's training epochs, adjusting its lookback window, or tuning filter sizes to improve temporal feature extraction.

Data Drift Monitoring

Sub

Backtesting Methodology

Sub

Results and Performance Analysis

Performance Metrics

To measure the effectiveness of the strategy, the following metrics were used:

• **Sharpe Ratio:** measures how much excess return a strategy generates for each unit of risk taken.

$$\text{Sharpe Ratio} = \frac{\mu_{\text{return}}}{\sigma_{\text{return}}}$$

where $\mu_{\text{return}}$is the average return of the investment and $\sigma_{\text{return}}$is the standard deviation of returns.

• **Sortino Ratio:** similar to the Sharpe Ratio, but it only considers downside risk (the standard deviation of negative returns).

$$\text{Sortino Ratio} = \frac{\mu_{\text{return}}}{\sigma_{\text{neg}}}$$

where $\sigma_{\text{neg}}$is the standard deviation of negative returns.

• **Maximum Drawdown (MDD):** indicates the largest loss an investment experiences from a peak to the lowest point before recovering.

$$\text{Maximum Drawdown} = \frac{\text{Maximum Value} - \text{Subsequent Minimum Value}}{\text{Maximum Value}}$$

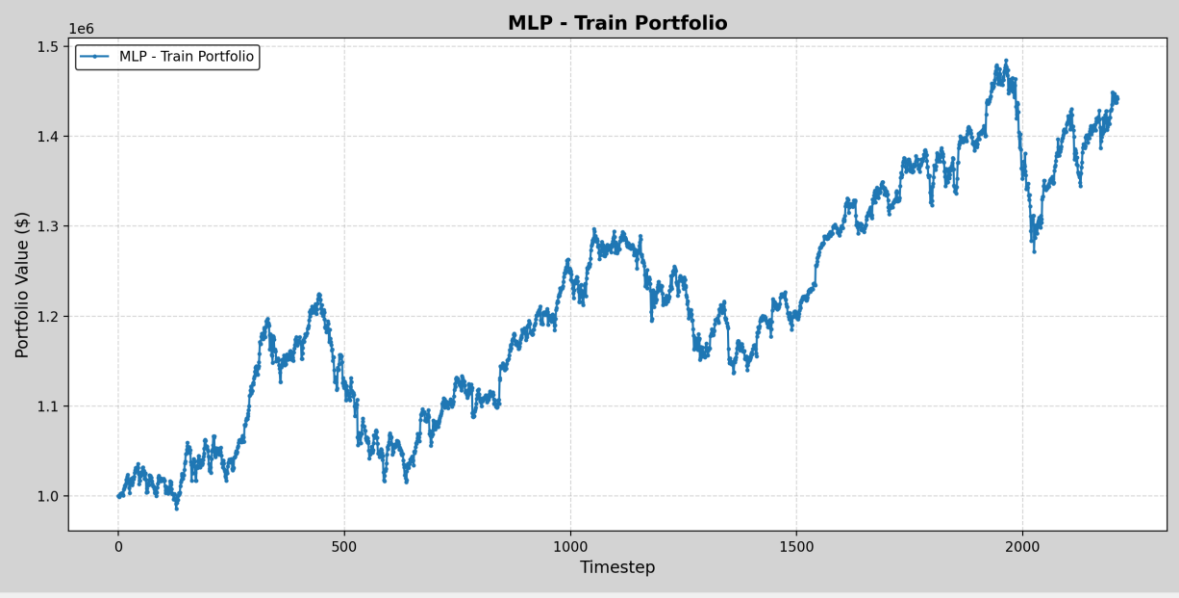
• **Calmar Ratio:** relates the average annual return to the maximum drawdown.

$$\text{Calmar Ratio} = \frac{\text{Average Annual Return}}{\text{Maximum Drawdown}}$$

These metrics allow evaluating both the profitability and the risk of the strategy, providing a comprehensive view of its performance and stability over time.

MLP PERFORMANCE

TRAIN (0.60)



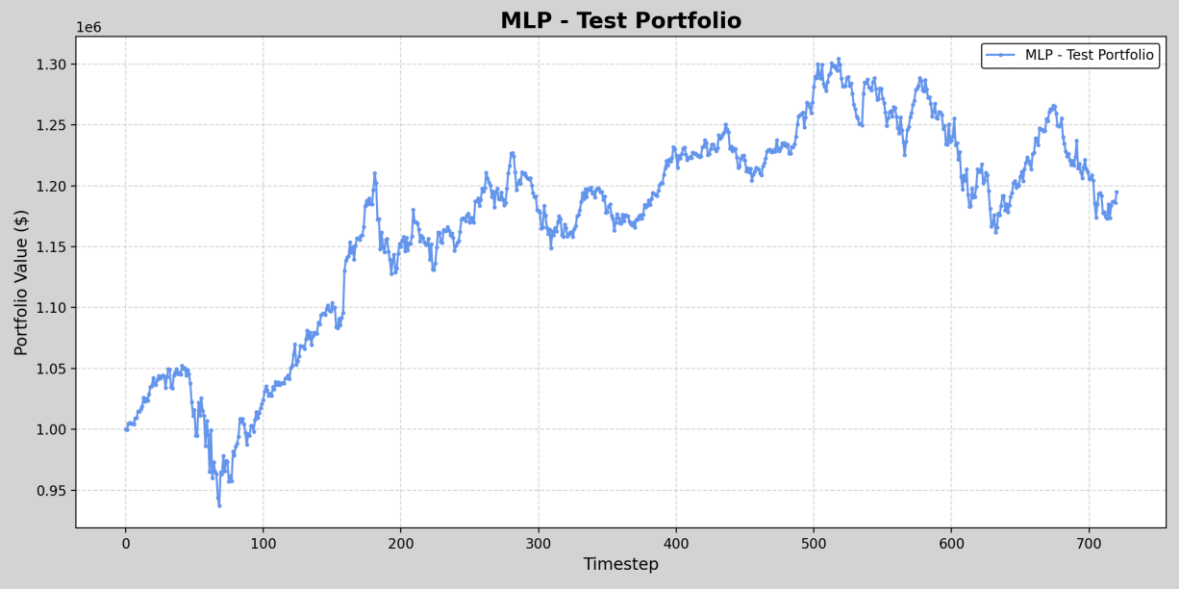
TRADING PERFORMANCE TRAIN

MLP TRAIN RESULTS	
Metric	**Value**
Final Capital	$999,884.88
Portafolio Value	1,441,924.59
Profit	441,924.59
Win Rate	44.2%
Return	43.4%

MLP TRAIN - Operations	
Metric	**Value**
Buy	282
Sell	0
Hold	1927
Total	281

MLP TRAIN - METRICS	
Metric	**Value**
Sharpe Ratio	0.0361
Sortino Ratio	0.0481
Maximum Drawdown	0.1704
Calmar Ratio	0.2689
Win Rate	0.5224

TEST (0.20)



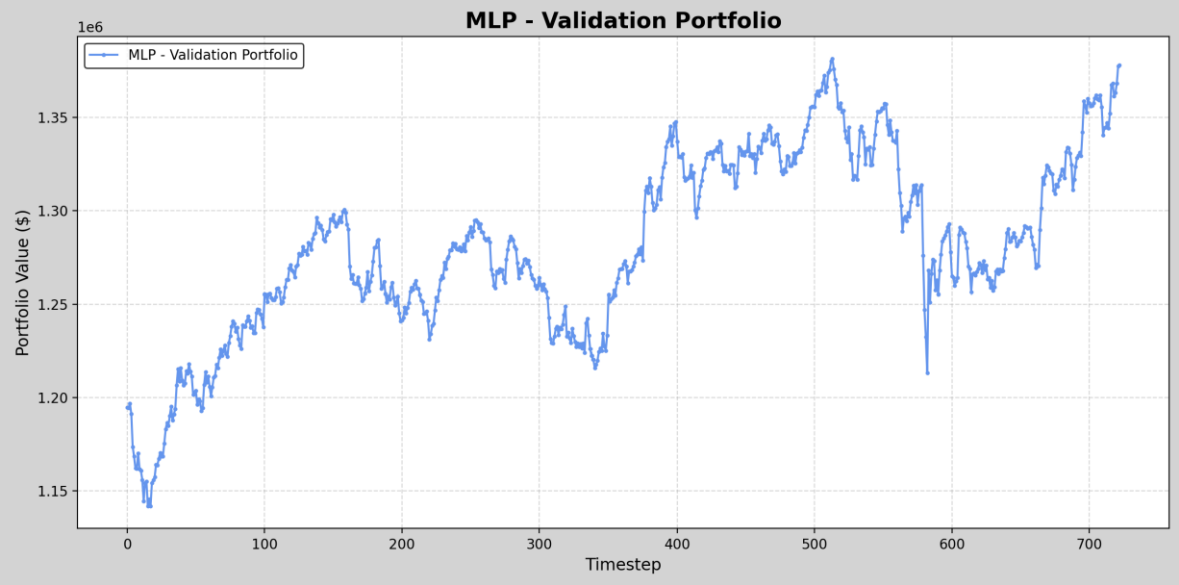
TRADING PERFORMANCE TEST

MLP TRAIN RESULTS	
Metric	**Value**
Final Capital	$836,260.56
Portafolio Value	1,194,850.04
Profit	194,850.04
Win Rate	41.7%
Return	19.5%

MLP TRAIN - Operations	
Metric	**Value**
Buy	128
Sell	0
Hold	592
Total	127

MLP TRAIN - METRICS	
Metric	**Value**
Sharpe Ratio	0.0391
Sortino Ratio	0.0532
Maximum Drawdown	0.1093
Calmar Ratio	0.6475
Win Rate	0.4174

VALIDATION (0.20)



TRADING PERFORMANCE VALIDATION

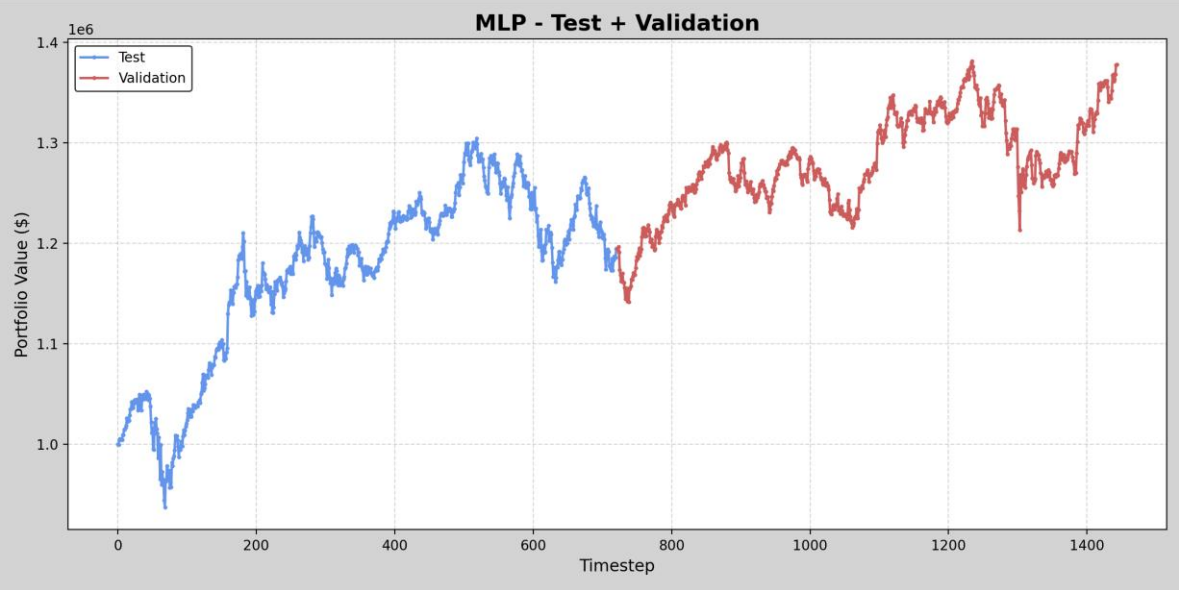
MLP VALIDATION RESULTS	
Metric	**Value**
Final Capital	$956,923.35
Portafolio Value	1,377,830.60
Profit	377,830.64
Win Rate	37.8%
Return	43.8%

MLP VALIDATION - Operations	
Metric	**Value**
Buy	97
Sell	0
Hold	625
Total	96

MLP VALIDATION - METRICS	
Metric	**Value**
Sharpe Ratio	0.0416
Sortino Ratio	0.0568
Maximum Drawdown	0.1217
Calmar Ratio	0.4466
Win Rate	0.5471

The MLP model achieved stable and positive growth throughout validation. Its portfolio increased steadily, showing profitable long positions and balanced risk exposure. With higher win rate and positive risk-adjusted ratios, the MLP demonstrated stronger predictive ability and robustness under market fluctuations.

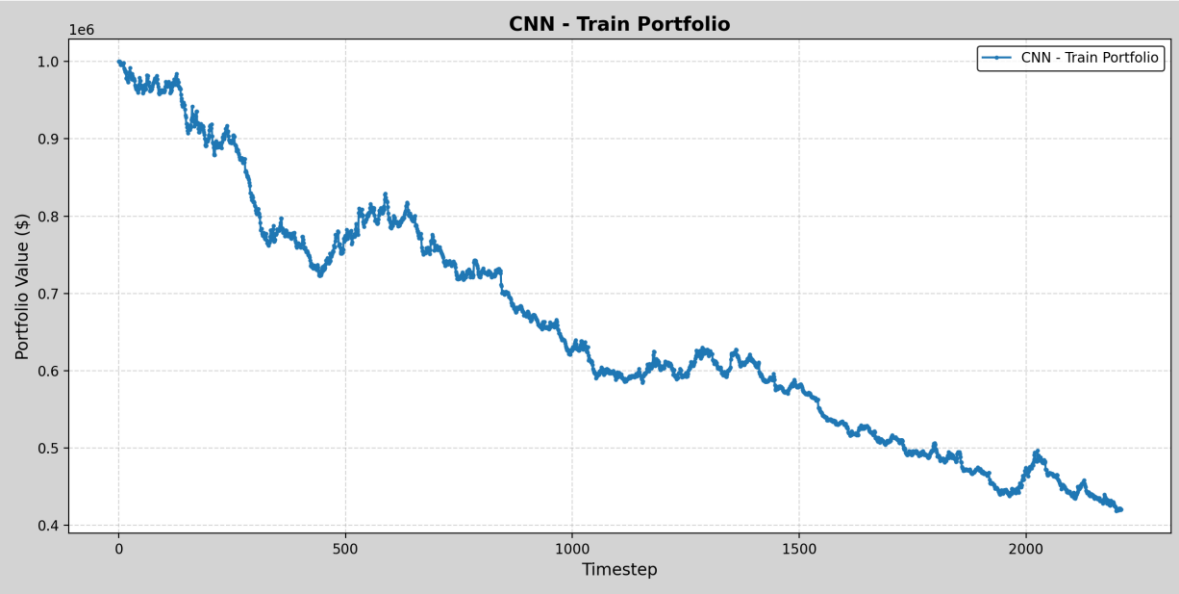
TEST + VALIDATION



The MLP model maintained a strong and consistent upward trend throughout both test and validation periods. The portfolio exhibited smooth growth with moderate fluctuations, reaching new highs toward the end of validation. This sustained performance suggests that the model generalized effectively, adapting well to changing market conditions. Its ability to recover quickly from drawdowns highlights solid risk management and predictive reliability. Overall, the MLP strategy demonstrated stable profitability, robustness, and strong potential for real-world trading applications.

CNN PERFORMANCE

TRAIN (0.60)



TRADING PERFORMANCE TRAIN

CNN TRAIN RESULTS	
Metric	**Value**
Final Capital	$418,442.62
Portafolio Value	420,520.80
Profit	-579,479.19
Win Rate	-58.0%
Return	26.1%

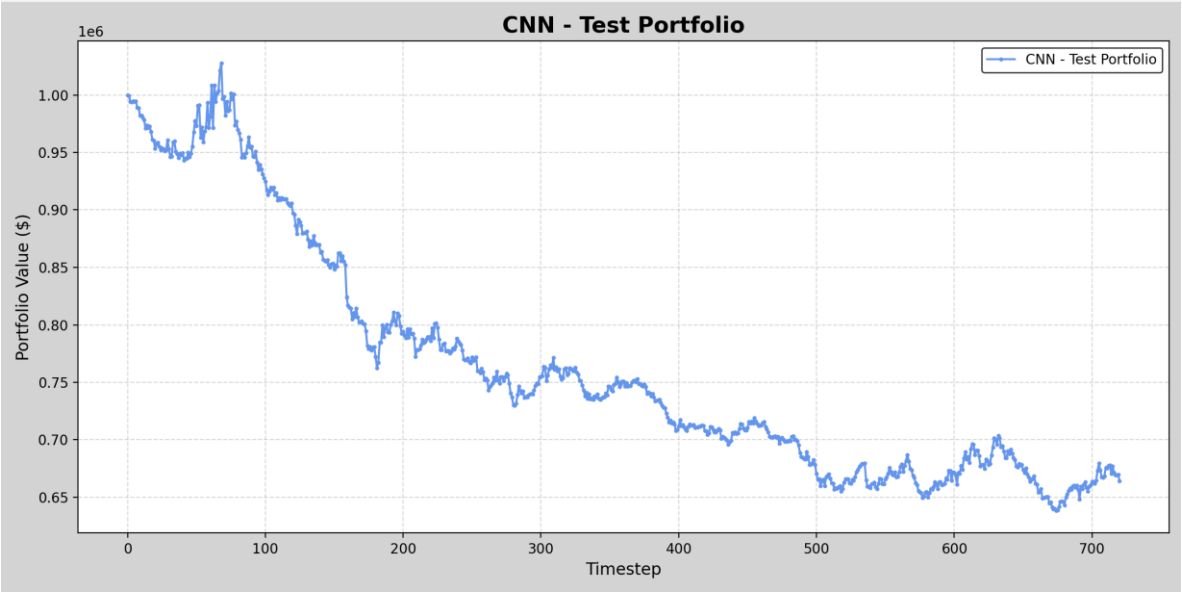
CNN TRAIN - Operations	
Metric	**Value**
Buy	0
Sell	311
Hold	1898
Total	310

CNN TRAIN - METRICS	
Metric	**Value**
Sharpe Ratio	-0.0779
Sortino Ratio	-0.1097
Maximum Drawdown	0.5816
Calmar Ratio	-0.1571
Win Rate	0.4753

The CNN model during training also performed poorly, ending with a final capital of $418,442.62 and a loss of $579,479.19. The negative Sharpe and Sortino ratios indicate that the strategy consistently took on risk without generating meaningful returns, while

the 58.16% maximum drawdown shows severe capital erosion and instability. The negative Calmar ratio further confirms that the losses were large relative to any potential reward. With only short positions taken, the strategy clearly struggled in the market conditions, producing weak signals and no diversification. Overall, this model shows high risk, large drawdowns, and persistent losses, making it unsuitable for trading in its current state.

TEST (0.20)



TRADING PERFORMANCE TEST

CNN TEST RESULTS	
Metric	**Value**
Final Capital	$663,821.85
Portafolio Value	$663,821.85
Profit	-336,178.15
Win Rate	-33.6%
Return	28.4%

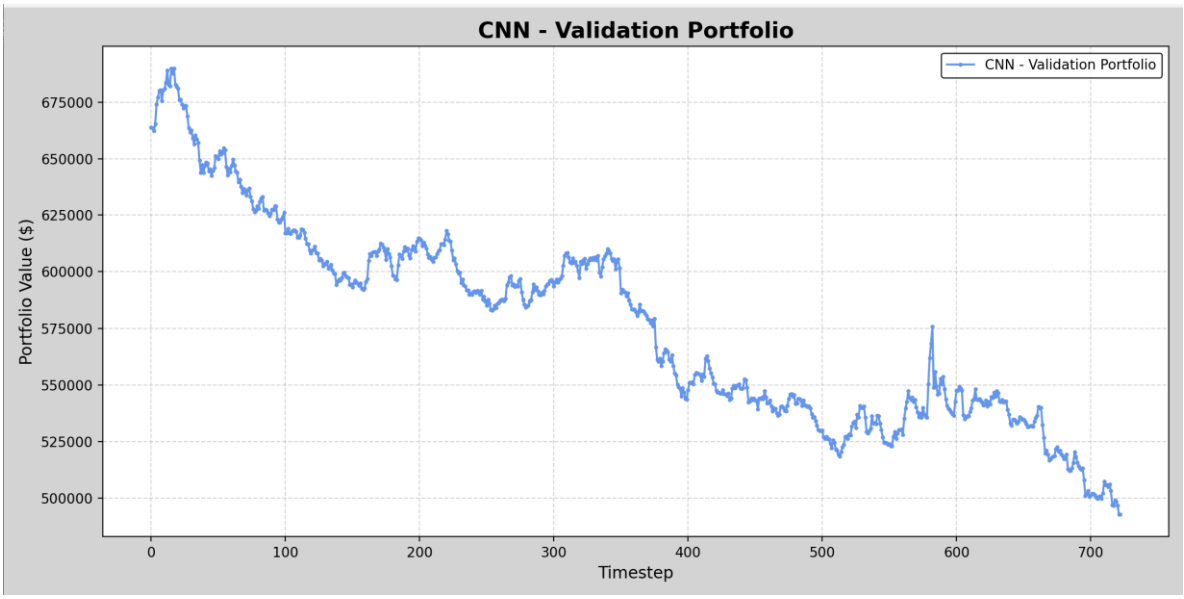
CNN TEST - Operations	
Metric	**Value**
Buy	0
Sell	135
Hold	585
Total	134

CNN TEST - METRICS	
Metric	**Value**
Sharpe Ratio	-0.0794
Sortino Ratio	-0.1106
Maximum Drawdown	0.3792
Calmar Ratio	-0.3387
Win Rate	0.4708

The CNN model during testing showed weak performance, consistently losing capital and ending with a final value of $663,821.85. Its negative Sharpe and Sortino ratios indicate that the strategy takes on risk without generating sufficient returns, while the 37.92% maximum drawdown reveals a high level of exposure and poor capital protection. The negative Calmar ratio further confirms that the losses outweigh any potential gains. Additionally, the strategy relied entirely on short positions, which likely performed poorly in the market environment. Overall, this model demonstrates a lack of robustness, poor risk-adjusted performance, and would not be viable for real trading in its current form.

The CNN model during testing showed weak performance, consistently losing capital and ending with a final value of $663,821.85. Its negative Sharpe and Sortino ratios indicate that the strategy takes on risk without generating sufficient returns, while the 37.92% maximum drawdown reveals a high level of exposure and poor capital protection. The negative Calmar ratio further confirms that the losses outweigh any potential gains. Additionally, the strategy relied entirely on short positions, which likely performed poorly in the market environment. Overall, this model demonstrates a lack of robustness, poor risk-adjusted performance, and would not be viable for real trading in its current form.

VALIDATION (0.20)



TRADING PERFORMANCE VALIDATION

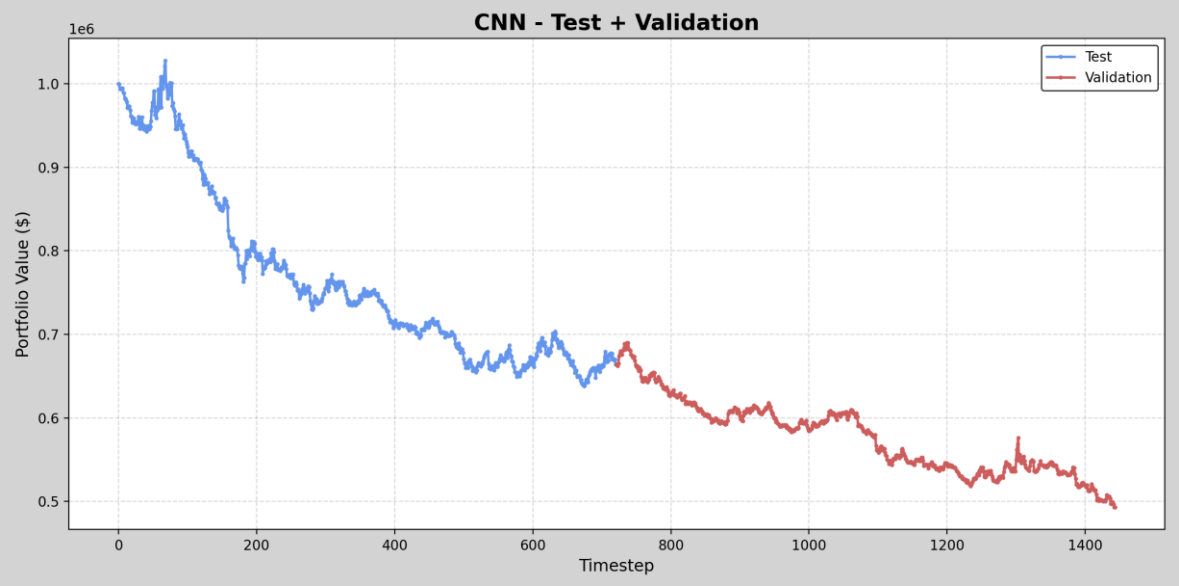
CNN VALIDATION RESULTS	
Metric	**Value**
Final Capital	$492,868.13
Portafolio Value	$492,868.57
Profit	-507,236.43
Win Rate	-50.7%
Return	25.0%

CNN VALIDATION - Operations	
Metric	**Value**
Buy	0
Sell	93
Hold	629
Total	92

CNN VALIDATION - METRICS	
Metric	**Value**
Sharpe Ratio	-0.0798
Sortino Ratio	-0.1037
Maximum Drawdown	0.2858
Calmar Ratio	-0.3356
Win Rate	0.4474

The validation results show a consistent decline in portfolio value, indicating poor generalization and ineffective signal prediction. The strategy produced only short positions, leading to cumulative losses and negative performance ratios. Overall, the CNN model failed to adapt to market dynamics and demonstrated weak profitability and risk control.

TEST + VALIDATION



The combined test and validation performance of the CNN model reveals a steady and persistent decline in portfolio value across both phases. The model failed to capture profitable trading patterns, maintaining a consistent downward trajectory from start to finish. This behavior indicates weak generalization and poor adaptability to market fluctuations. The lack of recovery during the validation period further confirms the CNN's inability to learn robust directional signals. Overall, the strategy demonstrates low profitability, limited resilience, and ineffective risk control.

Conclusions

Looking at both models, we can see that with the MLP model we made money, while with the CNN model we lost money. The MLP model gave us a final value of $1,337,830.60, whereas the CNN model lost a little more than half of the initial capital.

In the MLP model, we can observe a very low Sharpe ratio, which indicates that the strategy provides very little compensation for the level of risk taken. The Sortino ratio of 0.0568 is also very low, similar to the Sharpe ratio, meaning there is little reward for a considerable amount of risk. The drawdown of 12.17% is moderate, showing that the strategy experiences losses during certain periods but not extreme ones. The Calmar

ratio of 0.4466 indicates that the strategy does not provide enough return relative to its drawdowns.

In this case, we decided to use the MLP model because it was the one where we saw a profit, whereas with the CNN model we lost more than half of the initial capital.

Bibliography

OpenAI. (2025). ChatGPT (GPT-5-mini) [Large language model].

https://openai.com/chatgpt .

• Sharpe, W. F. (1966). Mutual fund performance. Journal of Business, 39(1), 119

138.

• Sharpe, W. F. (1994). The Sharpe ratio. Journal of Portfolio Management, 21(1),

49–58.

• Sortino, F. A., & Price, L. N. (1994). Performance measurement in a downside

risk framework. Journal of Investing, 3(3), 59–64.

• Calmar, T. (1991). The Calmar Ratio: A Measure of Return vs. Drawdown.

Futures, 20(4), 22–25.

• Maginn, J., Tuttle, D., McLeavey, D., & Pinto, J. (2007). Managing investment

portfolios: A dynamic process (3rd ed.). Wiley.

• Chan, E. (2009). Quantitative trading: How to build your own algorithmic trading

business. Wiley.

• Chan, E. (2009). Quantitative trading: How to build your own algorithmic trading

business. Wiley

• Antonopoulos, A. M. (2017). Mastering Bitcoin: Unlocking digital

cryptocurrencies (2nd ed.). O'Reilly Media.