Neural Networks for Stock Price Prediction

Pedro Leite
Master in Computer Science
Department of Computer Science
Faculty of Sciences of the University of Porto
2025

Neural Networks for Stock Price Prediction

Pedro Leite

Dissertation carried out as part of the Master in Computer Science
Department of Computer Science
2025

Supervisor
Inês Dutra, Assistant Professor,
Faculty of Sciences of the University of Porto

Acknowledgements

As I conclude my academic journey at the Faculty of Sciences of the University of Porto, where I had the privilege of completing both my Bachelor's and Master's degrees in Computer Science, I reflect with gratitude on this esteemed institution. I would also like to acknowledge the twelve formative years I spent at Externato Senhora do Carmo, which played a decisive role in shaping both the person and the student I am today. To all professors and staff who, through their relentless work and dedication, made this journey possible, I extend my sincere thanks.

A special note of gratitude goes to my supervisor, Professor Inês Dutra, for her invaluable time, guidance, and encouragement throughout this work. Her support in exploring this topic with me, her willingness to help in every way she could, and her insightful advice were fundamental to the development of this thesis.

Finally, I would like to express my deepest appreciation to my family, friends, and girlfriend for their continuous support, love, and patience. Their encouragement and understanding have been essential, not only during the challenges of my academic journey but throughout my entire life. A very special word goes to my father, who would be the proudest of me right now.

Abstract

Stock price prediction is a challenging problem due to the financial market's non-stationarity, noise, and complex temporal dependencies. Neural Networks (NNs) represent a rapidly expanding field across all domains, with stock market prediction being one of their most promising applications, due to their ability to capture non-linear patterns and complex relationships in data. While NNs have been widely used for this task in recent years, systematic comparisons across different configurations and optimization strategies, as well as assessments of real-world practicability, remain scarce. This thesis aims to address these gaps.

This study uses the Standard & Poor's 500 (S&P 500) stocks, with data from June 2020 to June 2025. The Apple (AAPL) stock is used as the optimization baseline, with final validation conducted on the broader S&P 500 set. Several configurations of Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) models are tested across univariate, market, and technical datasets, followed by multiple preprocessing methods, hyperparameters and regularization techniques.

Results show that the optimal model is a GRU 3x8 architecture with market variables, revealing mixed results across the 403 stocks, with an average Root Mean Squared Error (RMSE) of $15.47 and directional Accuracy of 51.1%. Although this type of model was proved unusable for real-world forecasting for most stocks tested, it showed promising results for a subset of them, with approximately 11.7% of the overall dataset achieving good performance suitable for practical application.

Keywords: Stock Price Prediction; Neural Networks; LSTM, GRU; S&P 500; Market Variables; Technical Indicators; PreProcessing; HyperParameters; Regularization.

Resumo

A previsão de preços de ações é um problema desafiante devido à não-estacionariedade, ruído e dependências temporais complexas dos mercados financeiros. As Neural Networks (NNs) representam um campo em rápida expansão em todos os domínios, sendo a previsão do mercado de ações uma das suas aplicações mais promissoras, devido à sua capacidade em capturar padrões não-lineares e relações complexas nos dados. Embora as NNs tenham sido amplamente utilizadas para esta tarefa nos últimos anos, comparações sistemáticas de diferentes configurações e estratégias de otimização, bem como avaliações da aplicabilidade no mundo real, permanecem escassas. Esta tese pretende abordar estas lacunas.

Este estudo utiliza as ações do Standard & Poor's 500 (S&P 500), com dados de Junho de 2020 a Junho de 2025. A ação Apple (AAPL) é usada como referência de otimização, com a validação final conduzida no conjunto mais amplo do S&P 500. Várias configurações de modelos Long Short-Term Memory (LSTM) e Gated Recurrent Unit (GRU) são testadas em conjuntos de dados univariados, de mercado e técnicos, seguidas de múltiplos métodos de pré-processamento, hiperparâmetros e técnicas de regularização.

Os resultados mostram que o modelo ótimo é uma arquitetura GRU 3x8 com variáveis de mercado, demonstrando resultados mistos nas 403 ações, com Root Mean Squared Error (RMSE) médio de $15.47 e precisão direcional de 51.1%. Embora este tipo de modelo se tenha revelado inutilizável para a previsão de preços no mundo real para a maioria das ações testadas, mostrou resultados promissores para um subconjunto delas, com aproximadamente 11.7% do conjunto de dados global a alcançar bom desempenho, adequado para aplicação prática.

Palavras-chave: Previsão de Preços de Ações; Redes Neuronais; LSTM; GRU; S&P 500; Variáveis de Mercado; Indicadores Técnicos; PréProcessamento; HiperParâmetros; Regularização.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Adaptive Delta Algorithm (AdaDelta)
Adaptive Gradient Algorithm (AdaGrad)
Adaptive Moment Estimation (Adam)
Apple (AAPL)
Artificial Intelligence (AI)
Artificial Neural Networks (ANNs)
AutoCorrelation Function (ACF)
Convolutional Neural Network (CNN)
Deep Learning (DL)
Efficient Market Hypothesis (EMH)
Exploratory Data Analysis (EDA)
Gated Recurrent Unit (GRU)
Gaussian Filter (GF)
Generative Adversarial Network (GAN)
Long Short-Term Memory (LSTM)
Machine Learning (ML)
Mean Absolute Error (MAE)
Mean Absolute Percentage Error (MAPE)
Mean Squared Error (MSE)
Multilayer Perceptron (MLP)
Neural Networks (NNs)
Partial AutoCorrelation Function (PACF)
Recurrent Neural Networks (RNNs)
Root Mean Square Propagation (RMSprop)
Root Mean Squared Error (RMSE)
Rolling Median (RM)
R Squared (R^2)
Savitzky-Golay Filter (SGF)
Simple Moving Average (SMA)
Standard & Poor's 500 (S&P 500)
Stochastic Gradient Descent (SGD)
Temporal Fusion Transformer (TFT)

1. Introduction

1.1 Context

Financial markets represent the backbone of modern economies, with the stock market serving as the primary venue for trading company shares and facilitating capital formation [1]. These markets enable corporations to secure funding for growth and innovation while offering investors the opportunity to participate in economic prosperity through equity ownership. Stock market movements function as crucial economic indicators, reflecting business confidence and economic health through their price discovery mechanisms [2].

The stock market stands as one of the most challenging domains for prediction due to temporal dependency, non-stationarity, high noise levels, various interacting influences, and sudden regime changes [3]. Markets face destabilizing forces including speculative asset bubbles, cascading failures across interconnected systems, liquidity crises, and sentiment-driven volatility from investor psychology and behavioural biases [4]. The competitive nature of financial markets, as described by the Efficient Market Hypothesis (EMH), suggests that prices rapidly incorporate all available information, creating an environment where any predictive advantage is quickly eroded as more participants discover and exploit it [5], [6].

Traditional methodologies, including fundamental, technical, and sentiment analysis, rely heavily on established financial theories and human expertise [7]. While these approaches offer interpretability and historical credibility, they have proven increasingly insufficient for navigating today's interconnected and rapidly evolving markets. Statistical models such as ARIMA and GARCH provide computational efficiency but suffer from restrictive assumptions of linearity and stationarity and are further limited by their inability to process multiple variables simultaneously [8]. Standard Machine Learning (ML) techniques like Random Forests and traditional regression models demonstrate greater flexibility in capturing non-linear relationships, yet they remain constrained by their limited capacity to model temporal dependencies, susceptibility to overfitting, and requirement for frequent recalibration as market dynamics shift [9].

1.2 Motivation and Objectives

The limitations of traditional, statistical and standard ML approaches have motivated the exploration of NN models for stock price prediction. NNs are computational models that mimic the brain's interconnected neuron structure through layers of mathematical nodes, representing a subset of ML that becomes Deep Learning (DL) when utilizing multiple hidden layers [10]. NNs have become one of the most transformative technologies in history, with its global market valued at nearly $97 billion in 2024 and projected to surpass $500 billion by 2030 [11]. In recent years, NNs have shifted from a niche research area to a cornerstone of innovation, driving economic growth, scientific discovery, and technological progress.

For the stock price prediction task, NN architectures, particularly sequence models like LSTMs and GRUs, have shown the ability to capture non-linear relationships and temporal dependencies that other methods cannot handle effectively, making them especially suitable for financial forecasting [2]. However, their application faces significant challenges, despite excelling at discovering complex patterns, these models are: highly opaque, prone to overfitting, and difficult to interpret, with training processes requiring substantial time and resources [12].

Although NNs are increasingly employed in financial forecasting, systematic studies that rigorously compare various methodological components remain under-explored, as do assessments of real-world applicability. This research aims to address these critical gaps through comprehensive systematic empirical evaluation. Additionally, current literature analysis reveals specific deficiencies that this research seeks to address:

- No consensus exists on optimal model selection. GRUs show the best results overall, but LSTMs have better results in specific contexts and are more widely used.
- Conflicting results emerge between univariate versus multivariate approaches.

- Mixed performance improvements reported from technical indicators.
- Absence of studies comparing performance across diverse stocks.

1.3 Methodology and Results

This research implements a structured empirical approach to identify the optimal NN model and its configurations for stock prices forecasting. The methodology leverages S&P 500 data encompassing 403 stocks across a 5-year period, from June 2020 to June 2025, with 24 numerical variables categorized into three groups: univariate, market, and technical.

AAPL serves as the baseline stock for the optimization phase. The experimental design systematically compares LSTM and GRU architectures through the three variable groups, followed by iterative refinement of preprocessing approaches, hyperparameter configurations, and regularization strategies. The investigation culminates with comprehensive performance evaluation for all stocks, employing both regression and classification metrics complemented by characteristics, sector and risk-return analysis.

The study reveals that simpler approaches, featuring basic data preprocessing, no regularization, fewer network layers, and smaller variable sets, consistently delivered superior performance compared to complex methodologies. The optimal GRU 3x8 architecture with market variables achieved a 15.47 RMSE and 51.1% directional prediction rate across all stocks, with only 11.7% of the set of stocks exhibiting good levels of performance suitable for practical deployment. The good performing stocks tend to have lower prices, lower volatility and bigger risk-adjusted metrics. Additionally, risk-return analysis confirmed the patterns found, but showing more preference towards higher priced stocks, because of their higher potential gains. The Financial sector showed the best performance and risk-return results.

1.4 Organization

This thesis is organized into six chapters following this introduction:

2. **Background:** Establishes the theoretical framework underlying financial NN applications, examining key concepts such as stock market behaviour, data properties, preprocessing techniques and NN architectures.
3. **State-of-the-Art:** Conducts a comprehensive literature review through systematic methodology, beginning with carefully defined search queries, followed by multi-stage filtering and finally a detailed article analysis.
4. **Methodology:** Outlines the methodological framework, covering dataset development, preprocessing methods, optimization stages, and the final evaluation procedure.
5. **Results and Discussion:** Presents the comprehensive findings from all three modelling phases, analysing the performance of different architectures, variable configurations, preprocessing methods, hyperparameters, regularization techniques, before analysing the final model performance across a broad set of stocks to evaluate its generalizability.
6. **Conclusion:** Synthesizes key findings, discusses the main contributions of NNs for stock forecasting research, identifies limitations of the current study, and suggests directions for future research.

2. Background

This chapter lays the theoretical foundation for the work developed in this thesis. It explores the relevant concepts and challenges associated with applying NNs for stock prices prediction, by presenting the necessary context on the stock market, data characteristics, and NN modelling. The study also explores the role of data preprocessing, as well as considerations for model optimization and evaluation.

2.1 Stock Market

The stock market represents the most popular of all financial markets, serving as a platform where shares of publicly traded companies are bought and sold. Understanding the full impact and structure of the stock market requires exploring its economic importance and the risks and challenges that shape its behaviour.

2.1.1 Importance

The stock market plays a critical role in fostering economic development by enabling companies to raise long-term capital, which fuels corporate expansion, innovation, and job creation. It supports efficient capital allocation through transparent price discovery mechanisms driven by supply and demand, helping to reflect the true value of firms. By going public, companies gain greater visibility, access to diversified funding, and the benefits of risk-sharing with a broad investor base, contributing to financial stability and resilience during economic uncertainty. This process guides both corporate investment strategies and individual investor decisions, making the stock market a barometer for national and global economic health [13].

Moreover, it enhances financial inclusion by providing individuals and small enterprises with access to investment and funding opportunities beyond traditional banking systems. By offering opportunities to invest in a wide range of companies, industries, and innovations, it enables individuals to participate directly in economic growth and benefit from the success of businesses. The markets play a crucial role in wealth creation and retirement planning, with pension funds and individual savings many times dependent on equity performance for long-term financial security [4].

2.1.2 Risks and the Efficient Market Hypothesis

While the stock market offers substantial opportunities, it also faces significant risks that threaten financial stability. Key vulnerabilities include speculative bubbles where asset prices exceed intrinsic value before bursting, systemic risks that spread disruptions across interconnected financial systems, liquidity risks during market panics when assets cannot be traded without major price impacts, and sentiment-driven volatility influenced by psychological factors like herd behaviour and news overreactions. Understanding these interconnected risks is essential for effective risk management and informed investment decisions [4], [14].

These market complexities and risks are further understood through the theoretical framework of the EMH, which asserts that all accessible information is reflected in asset pricing. The competitive nature of financial markets suggests that prices rapidly incorporate all available information, creating an environment where any predictive advantage is quickly eroded as more participants discover and exploit it. While EMH provides a theoretical foundation for understanding market behaviour, empirical evidence suggests that market inefficiencies exist, driven by the very behavioural biases and institutional constraints that create the risks outlined above [6].

2.1.3 Standard & Poor’s 500

The S&P 500 is a stock market index that tracks approximately 500 of the largest publicly traded companies in the United States (U.S.) by market capitalization. This market capitalization-weighted index gives companies with higher market values greater influence on the index's overall performance. The S&P 500 effectively represents the U.S. equity market, which constitutes the world's largest financial market by total capitalization, with U.S. based companies accounting for nearly 60% of global equity market capitalization. The index encompasses companies across all major economic sectors, including technology, healthcare, financials, and energy [15].

The S&P 500 serves as a key barometer of American financial market health and economic performance. To qualify for inclusion, companies must meet specific criteria including a minimum market capitalization threshold of $5 billion, ensuring the index focuses on large-cap enterprises. While all constituents meet this large-cap classification, they demonstrate considerable variation in size, with market capitalizations spanning from companies near the minimum requirement to trillion-dollar mega-cap corporations [16].

2.2 Stock Data

Stock data serves as the foundation for all market prediction models, encompassing diverse information that captures the financial market dynamics. Understanding the key types of variables, from basic market metrics to technical indicators, is essential alongside the temporal frameworks that guide prediction strategies. Working with financial data presents inherent challenges, including noise, non-stationarity, and collection obstacles that make stock market forecasting particularly complex.

2.2.1 Variables

In stock market analysis, different types of variables are used to capture different behaviours, influences and patterns. These variables can be grouped into distinct categories based on the nature of the information they convey. Below is an overview of two of the most popular categories and variables for the task of stock price prediction:

- Market Variables[1]: The most used and most important type of stock's variables, they describe the prices and volume traded. They are:
    - Open: Price at which the stock first trades in the given time interval.
    - Close: Price at which the stock last trades in the given time interval.

[1] https://support.google.com/docs/answer/3093281

  - High: Highest price at which the stock trades in the given time interval.
  - Low: Lowest price at which the stock trades in the given time interval.
  - Volume: Number of trades in the given time interval.
- Technical Indicators[2]: Derived from market variables, technical indicators are mathematical calculations used to analyse market strength. Some of them are:
  - Trend:
    - Simple Moving Average (SMA): Calculates the average of closing prices over a specific number of periods.
    - Exponential Moving Average (EMA): Calculates a weighted moving average that gives more importance to recent prices.
    - Moving Average Convergence Divergence (MACD): Shows the relationship between two EMAs.
    - Average Directional Index (ADX): Measures the strength of a trend, regardless of its direction.
    - Parabolic SAR (PSAR): Places dots above or below price to indicate trend direction.
  - Momentum:
    - Relative Strength Index (RSI): Measures momentum on a scale from 0 to 100.
    - Rate of Change (RC): Measures the percentage change in price between the current price and a previous price.
    - Stochastic Oscillator K (SOK): Compares the current closing price to the price range over a specific period.
    - Momentum: Calculates the difference between the current price and the price some periods ago.
    - Williams %R: Measures overbought and oversold levels by comparing the closing price to the high-low range over a set period.
    - Triple Exponential Average (TRIX): Shows the rate of change of a triple EMA.
    - Chande Momentum Oscillator (CMO): Measures momentum on both up and down days.
  - Volatility:
    - Bollinger Bands (BB): Consists of a moving average and two standard deviation lines above and below it.

[2] https://technical-analysis-library-in-python.readthedocs.io/en/latest/ta.html

- Standard Deviation (SD): Measures the dispersion of price data from its mean.

o Volume:
  - On Balance Volume (OBV): Adds volume on up days and subtracts volume on down days cumulatively.
  - Accumulation/Distribution (AD): Evaluates the cumulative flow of money into and out of a stock using both price movement and volume.
  - Money Flow Index (MFI): Uses both price and volume to identify overbought or oversold conditions.

o Statistical:
  - Commodity Channel Index (CCI): Evaluates how far the current price has deviated from its statistical average.
  - Balance of Power (BOP): Assesses the strength of buying and selling pressure by comparing the close price to the trading range.

2.2.2 Setup

Stock data can be recorded at various time intervals depending on the analysis objectives. Each frequency serves specific types of strategies and insights, balancing between detail and long-term trends. They can go from high-frequency (milliseconds or microseconds) to daily or even monthly [17].

The choice of data frequency is closely linked to the prediction time horizon and modelling approach. Depending on the selected data frequency and investment strategy, stock data can be used for predictions across different time horizons: short-term (intra-day to a few days), medium-term (weeks to a few months) or long-term (several months to years) [17].

Additionally, the structure of stock data determines how prediction tasks are formulated. Stock price prediction can be approached through two main modelling frameworks: regression where the goal is to predict continuous values, such as exact future prices or expected returns or classification where the goal is to predict discrete categories, such as whether prices will move up, down, or remain stable [17].

2.2.3 Nature, Challenges and Biases

Stock market data represents one of the most challenging domains for predictive modelling due to its inherent complexity and dynamic characteristics. Unlike many other prediction tasks where patterns remain relatively stable, stock data exhibits several distinctive properties that make forecasting particularly difficult, like [3], [18]:

- Temporal Dependency: Stock prices form time series data where present values are influenced by past observations.
- Non-stationarity: Stock data distributions change over time, this means statistical properties like mean, variance, and autocorrelation vary across different periods.
- High Noise: Stock prices contain substantial random components alongside genuine signals, making pattern identification difficult.
- Various Influences: Stock movements are driven by numerous interacting factors including company performance, sector trends, macroeconomic conditions, geopolitical events, and investor psychology.
- Regime Changes: Markets can suddenly shift between different states, requiring adaptation to varying regimes.

Beyond these inherent data characteristics, researchers must also address systematic biases in data selection. When conducting financial market research, survivorship bias represents a significant methodological concern that can lead to overly optimistic results. This bias occurs when analysis focuses only on companies that have remained successful, systematically excluding firms that have failed, been delisted or severally decline. In stock market studies, this creates an artificial upward bias in performance metrics, as the sample inherently excludes the worst-performing stocks. To obtain realistic and representative results, researchers must make deliberate efforts to include both high-performing and underperforming stocks in their datasets [19].

Collecting and utilizing stock data presents multifaceted challenges that researchers must navigate carefully, ranging from data quality and integration issues to access limitations and regulatory constraints. Access becomes particularly challenging for smaller exchanges or emerging markets, where data may be sparse or expensive, and historical data availability varies significantly by provider and data type, especially for newer information sources like social media sentiment, which lack long-term historical presence. Legal and regulatory constraints further complicate data collection across different regions, making transparency and methodological rigor essential for building trustworthy and reproducible forecasting systems [20].

2.3 Data PreProcessing

Data preprocessing serves as an essential step to build balanced predictive models by transforming raw data into formats suitable for analysis. This multi-step process encompasses: Exploratory Data Analysis (EDA) to understand temporal patterns and statistical properties, data cleaning to handle missing values and outliers, data transformation through smoothing and normalization techniques, and structured data preparation using sliding window approaches.

2.3.1 Exploratory Data Analysis

EDA serves as an important step for developing effective predictive models, particularly in the complex domain of financial time series forecasting. EDA enables researchers to understand the underlying structure, patterns, and statistical properties of the data before modelling.

To explore the characteristics in time-series data, a series of visualizations and statistical diagnostics can be applied to reveal temporal dependencies, identify non-stationarity, uncover seasonal patterns, and expose noise characteristics. Some of them are [21]:

- Seasonal Decomposition (Trend, Seasonal, Residual): Separates a time series into three additive or multiplicative components: trend, seasonality, and residuals. The trend component captures long-term directionality, the seasonal component represents regular, repeating cycles, and the residual reflects irregular noise or shocks:

$$Y_t = T_t + S_t + R_t \qquad T_t = \frac{1}{2k+1}\sum_{i=-k}^{k} Y_{t+1} \qquad S_t = Y_t - T_t \qquad R_t = Y_t - T_t - S_t$$

- AutoCorrelation Function (ACF): Quantifies the linear relationship between current values and past values at different lags:

$$p_k = \frac{Cov(Y_t, Y_{t-k})}{\sqrt{Var(Y_t)Var(Y_{t-k})}}$$

- Partial AutoCorrelation Function (PACF): Measures the correlation between a time series and its lagged version after removing the influence of all shorter lags:

$$\alpha_k = \phi_{kk} \qquad Y_t = \sum_{i=1}^{k} \phi_{ki} Y_{t-i} + \varepsilon_t^{(k)}$$

In addition to the methods described above, specific to time-series data, there are other general EDA techniques that are useful for every modelling task, like: distribution analysis, outlier detection, heatmaps, and many others. Among these is correlation analysis. Correlation analysis evaluates the strength and direction of linear relationships between pairs of variables. The most used correlation measure is the Pearson Correlation Coefficient [21]:

$$r_{X,Y} = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2 \sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}}}$$

While the previous techniques focus on data patterns and relationships, evaluating fundamental financial characteristics offers additional insights into stock behaviour. For comprehensive market analysis, several key financial metrics serve as essential descriptors [6], [21]:

- Growth: Quantifies the geometric average annual return over the investment period:

$$Growth = (\frac{V_f}{V_i})^{\frac{1}{n}} - 1$$

- Volatility: Measures the dispersion of price movements around the mean return using the standard deviation of daily returns:

$$Volatility = \sqrt{\frac{\sum_{t=1}^{n}(r_t - \bar{r})^2}{n-1}}$$

- Seasonality: Quantifies the proportion of total variance attributable to regular, predictable seasonal patterns through additive decomposition:

$$Seasonality = \frac{Var(S_t)}{Var(X_t)}$$

- Sharpe Ratio: Measures risk-adjusted returns by comparing excess returns to volatility:

$$SR = \frac{\overline{r_p} - r_f}{\sigma_p}$$

- Maximum Drawdown: Captures the largest peak-to-trough decline in cumulative returns:

$$MD = \min(\frac{V_t - p_t}{p_t})$$

2.3.2 Data Cleaning

Data cleaning represents a crucial step in preprocessing stock market data, focusing on identifying and addressing data quality issues that could compromise model performance. This process primarily involves two key tasks: imputing missing values that occur due to market closures or data collection gaps and appropriately handling outliers that may represent either data errors or legitimate extreme market events.

Stock market data commonly exhibits null values stemming from various sources, like scheduled market closures during weekends and holidays, as well as general data gaps that can occur at any time. While planned trading halts differ conceptually from random missing data, both scenarios can be addressed using similar preprocessing strategies. One of those techniques is Forward Filling, which uses the last known price before the gap, which assumes the price remains unchanged during missing periods.

Beyond missing data, handling outliers in stock price data requires careful consideration as they can significantly impact model performance while sometimes representing legitimate market signals. Outliers in financial time series often include extreme price movements during market crashes, earnings announcements, or major news events. Unlike in other domains, financial outliers frequently contain valuable information about market dynamics and shouldn't be automatically removed. Instead of deletion, effective approaches preserve these data points while reducing their extreme impact through capping or limiting techniques. The most popular approaches are [22]:

- Winsorization: Limits extreme values in a dataset by replacing them with the nearest values within a specified percentile range.
- Clipping: Sets all observations outside a fixed user-defined limit to the nearest boundary value.

2.3.3 Data Transformation

Preprocessing techniques such as data smoothing and normalization are essential in preparing financial time series for modelling. These steps help reduce noise, improve model stability, and enhance learning efficiency.

Data smoothing reduces noise and short-term fluctuations in time series data to reveal underlying trends or patterns that are more relevant for modelling. Common data smoothing techniques are [21], [23]:

- SMA: Computes the arithmetic mean of a specified number of consecutive data points within a sliding window:

$$SMA_t = \frac{1}{n}\sum_{i=0}^{n-1} x_{t-i}$$

- Rolling Median (RM): Calculates the median value within a moving window of specified length:

$$RM_t = median(x_{t-n+1}, \dots, x_t)$$

- Gaussian Filter (GF): Applies a weighted moving average using Gaussian weights that decrease exponentially with distance from the centre:

$$GF_t = \sum_{i=-k}^{k} x_{t-1}\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{i^2}{2\sigma^2}}$$

- Savitzky-Golay Filter (SGF): Performs local polynomial regression within a moving window to smooth data while preserving higher-order moments like peaks and valleys:

$$SGF_t = \sum_{i=-k}^{k} c_i x_{t+1}$$

Normalization converts values measured on different scales to a conceptually shared scale, which enhances training efficiency, prevents dominance by high-magnitude features, and supports model convergence. Common normalization techniques include [22]:

- MinMax: Rescales data to a fixed range by linearly transforming values based on the observed minimum and maximum:

$$MinMax(x) = \frac{x - \min(x)}{\max(x) - min(x)}$$

- Z-Score: Transforms data to have zero mean and unit variance by subtracting the mean and dividing by the standard deviation:

$$Z(x) = \frac{x - \mu}{\sigma}$$

- Log Normalization: Applies logarithmic transformation followed by MinMax, while ensuring the final output is bounded between the fixed range:

$$LogNorm(x) = \frac{\log(x - x_{min} + 1) - \min(\log(x - x_{min} + 1))}{\max(\log(x - x_{min} + 1)) - \min(\log(x - x_{min} + 1))}$$

2.3.4 Data Preparation for Sequential Neural Networks

Once the data has been analysed, cleaned and transformed, the final preprocessing step involves structuring the data into formats specifically required by sequential NN architectures. This essential step converts raw time series data into supervised learning formats using systematic data partitioning and sequential windowing approaches.

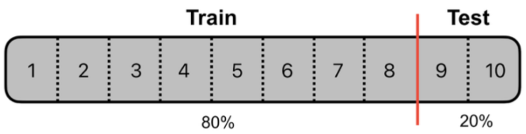

Figure 2.1: 80-20 train-test split

Splitting data into training and testing sets is a fundamental step in building and evaluating predictive models. The training set is used to fit the model and learn patterns from historical data, while the test set assesses the model's ability to generalize to unseen data. In time series forecasting, this split must respect the temporal order to avoid data leakage. Common strategies include using a fixed split based on time, like in Figure 2.1 [21].

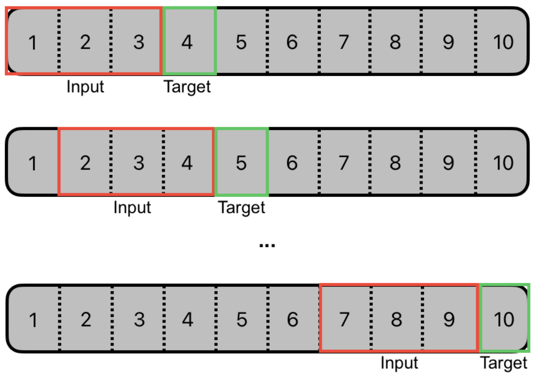

Figure 2.2: Sliding window with size 3 and 1 variable

Time series forecasting with NNs requires converting sequential data into a supervised learning format using the sliding window technique. This method generates input-output pairs by applying a fixed-size window of historical observations to predict the next value in the sequence. As illustrated in Figure 2.2, with a window size of 3, the example shows a univariate time series, where the method systematically moves across the dataset, producing multiple samples where each input sequence consists of consecutive time steps, and the corresponding target is the immediate next value. If multiple variables were used, each time step would contain observations from all variables, making the input window a matrix rather than a vector [24].

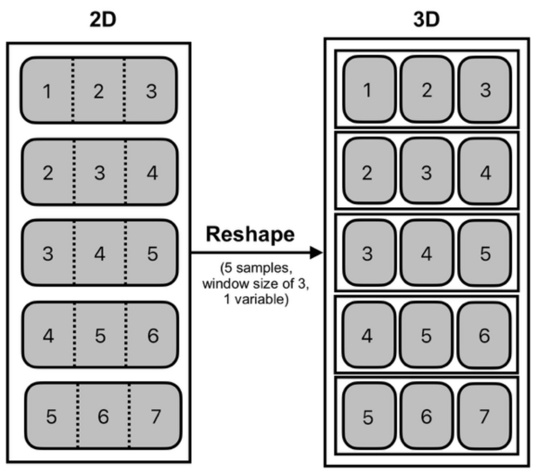

Figure 2.3: Reshaping the sliding window with 5 samples, size 3 and 1 variable

After creating the input-output pairs, the data should be reshaped to match the input format required by NNs like LSTMs and GRUs. As shown in Figure 2.3, this involves organizing the overlapping sequences into a 3-dimensional structure representing samples, window size, and variables. This format allows the model to correctly interpret the temporal order and feature information, which is essential for learning from sequential data [24].

2.4 Neural Networks

NNs provide powerful frameworks for modelling complex, non-linear dependencies in sequential financial data. Their design spans from foundational structures such as Artificial Neural Networks (ANNs) and Recurrent Neural Networks (RNNs) to more advanced architectures like Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), which are particularly suited for capturing temporal dynamics. Effective application of NNs relies on optimization strategies, regularization techniques, and rigorous evaluation methods.

2.4.1 Artificial Neural Network

ANNs are computational models inspired by the structure and functioning of the human brain. At their essence, ANNs are mathematical frameworks consisting of interconnected neurons, organized in layers to create a computational architecture capable of making predictions from data.

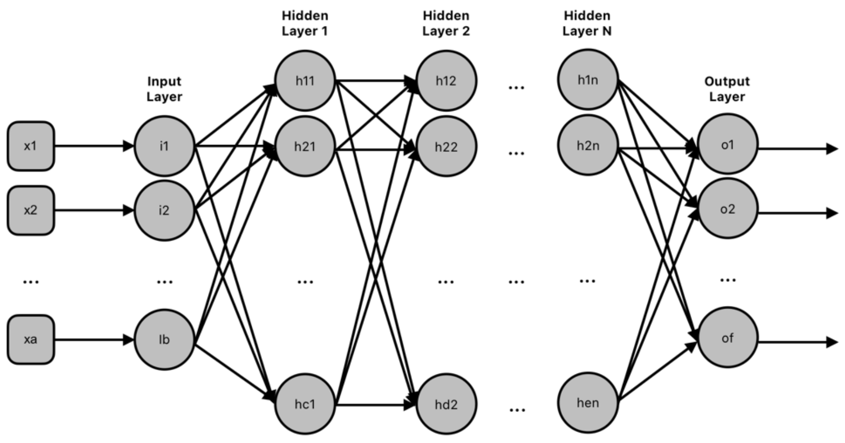

Figure 2.4: Structure of ANNs

As shown in Figure 2.4, an ANN consists of neurons organized in layers. There are three types of layers [25]:

- Input Layer: Composed with the neurons i_1, i_2, …, i_b that receive the inputs x_1, x_2, …, x_a (raw data).
- Hidden Layers: Composed with multiple layers with the neurons h_{11}, h_{21}, …, h_{en}.
- Output Layer: Composed with the neurons o_1, o_2, …,o_f producing the results, which is a predicted class label or a numerical value.

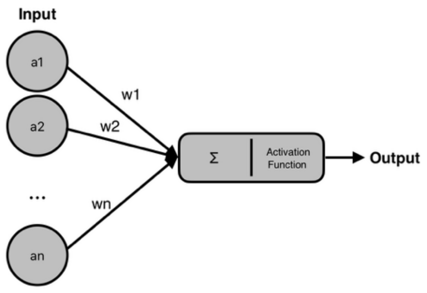

Figure 2.5: Structure of a neuron in ANNs

Figure 2.5 illustrates how a single neuron works. Comprised with [25]:

- Input Processing: Each neuron receives inputs a_1, a_2, …, a_n.

- Weighted Sum: The neuron computes a weighted sum of its inputs:

$$z = w_1 a_1 + w_2 a_2 + \cdots + w_n a_n + b$$

  - Where:
    - $w_1, w_2, \ldots, w_n$.are weights (determining the strength of influence).
    - b is a bias term (providing flexibility in the output).
- Activation Function: The weighted sum passes through an activation function. The activation function introduces non-linearity, allowing the network to model complex relationships in data.

When data is passed through the network, the process is called forward propagation. The model produces an output, which is then compared to the actual result using a loss function, a mathematical way of measuring how far off the prediction is. Based on this error, the model updates its weights and biases through a process called backpropagation. Backpropagation uses gradient descent, an optimization algorithm, to minimize the loss function and improve the model's accuracy over time. Backpropagation uses the chain rule of calculus to efficiently calculate these derivatives throughout the network [25].

2.4.2 Recurrent Neural Network

ANNs were created to handle static data, and they fall short when working with sequential data, where the order and context matter, such as time series data. RNNs were specifically designed to address this limitation by introducing a memory mechanism.

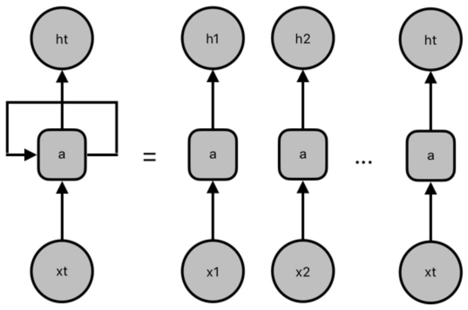

Figure 2.6: Structure of RNNs

As illustrated in Figure 2.6, RNNs differ from traditional feedforward networks by including recurrent connections that allow them to maintain a hidden state over time. This hidden state acts like memory, enabling the network to use past information when processing current inputs [26].

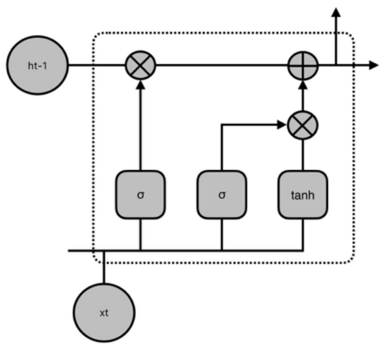

Figure 2.7: Structure of a neuron in RNNs

Figure 2.7 shows the RNN neuron structure, comprised with [26]:

- State Update: The core computation combines the current input x_t with the previous hidden state h_{t-1}, using learned weight matrices. It applies an activation function f to introduce non-linearity and control the information flow:
$$h_t = f(w_x x_t + w_h h_{t-1} + b)$$
- Output Generation: The hidden state is transformed to produce the final output, often using a linear transformation followed by the activation function f:
$$y_t = w_y h_t + b_y$$
- Weight Sharing: Unlike traditional ANNs, the same weight matrices are reused across all time steps, allowing the network to learn consistent temporal patterns while maintaining parameter efficiency.

2.4.3 Long Short-Term Memory

The LSTM model is an advanced RNN, created to escape the exploding/vanishing gradient problem, where gradients become extremely small as they are backpropagated through layers, making weight updates negligible and preventing the model from learning long-range dependencies, simultaneously, they grow uncontrollably, causing unstable updates and divergence during training. LSTMs manage to address this limitation by using special components called gates, which control the flow of information [2].

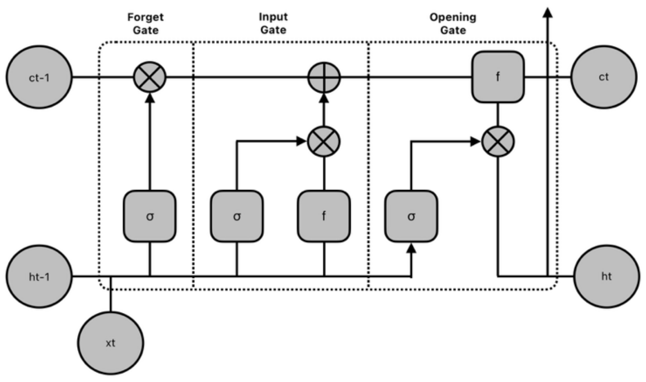

Figure 2.8: Structure of a neuron in LSTMs

Figure 2.8 shows the LSTM neuron structure, comprised with [2]:

- Input Gate: Regulates how much new information from the current input x_t and previous hidden state h_{t-1} should be added to the cell state. It applies a Sigmoid activation to determine the importance of each input feature and combine it with a candidate cell state, generated via the activation function f. The product of these two defines what will be added to the updated cell state c_t:
$$i_{ga} = \sigma(w_{ip}[h_{t-1}, x_c] + b_i)$$
- Forget Gate: Decides what portion of the previous cell state c_{t-1} should be retained or discarded. It uses a Sigmoid function to output a value between 0 and 1 for each element, 0 means "forget completely," and 1 means "fully keep". The result is multiplied element-wise with c_{t-1}, effectively filtering the memory carried over:
$$f_{ga} = \sigma(w_{fp}[h_{t-1}, x_c] + b_f)$$
- Output Gate: Determines what part of the new cell state should be exposed as the hidden state h_t and possibly passed to the next layer or time step. It applies a Sigmoid to the combined input and multiplies it element-wise with the activation function f cell state. This final product becomes the output of the LSTM unit:
$$o_{ga} = \sigma\big(w_{op}[h_{t-1}, x_c] + b_o\big)$$

2.4.4 Gated Recurrent Unit

Just like LSTMs, GRU is a RNN based model designed to escape the exploding/vanishing gradient problem, but it’s more computationally efficient and faster to train. GRU cells are designed to efficiently capture dependencies across time steps in sequential data. Instead of separate memory and hidden states like LSTMs, GRUs merge them into a single hidden state. They use two gates, update and reset, to control the flow of information [2].

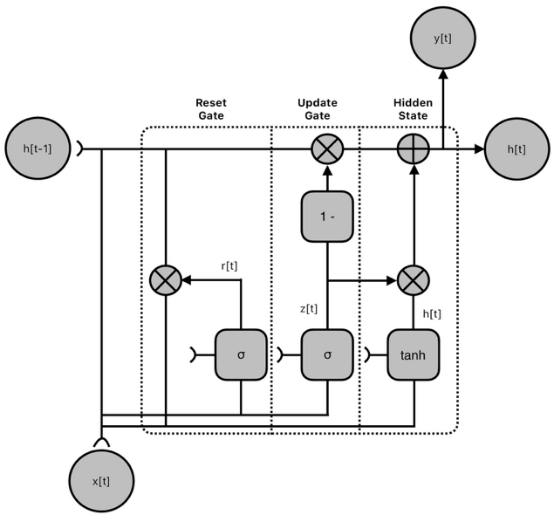

Figure 2.9: Structure of a neuron in GRUs

Figure 2.9 shows the GRU neuron structure, comprised with [2]:

- Update Gate: Decides how much of the previous hidden state h_{t-1} should be preserved and carried over to the current hidden state h_t. It uses a Sigmoid activation applied to a combination of the current input x_t and the previous hidden state h_{t-1}. A value of 1 means "keep everything from the past," while a value of 0 means "discard the past entirely." The update gate blends the candidate hidden state and the old hidden state to form the new one:
$$z[t] = \sigma(w^{(z)}x_t + u^{(z)}h_{t-1})$$
- Reset Gate: Controls how much of the previous hidden state h_{t-1} should influence the candidate's hidden state. Like the update gate, it applies a Sigmoid activation to a combination of x_t and h_{t-1}. When the reset gate outputs values close to 0, it "resets" the memory, essentially ignoring past context and focusing on the current input. When close to 1, it allows past information to influence the candidate's hidden state fully:
$$r[t] = \sigma\left(w^{(r)}x_t + u^{(r)}h_{t-1}\right)$$

2.4.5 HyperParameters

Unlike model parameters (such as weights and biases), hyperparameters are not learned during training, they must be defined before the learning process begins. Hyperparameter tuning is the process of systematically selecting the best set of hyperparameters that govern the model's training dynamics and architecture. Key hyperparameters include [25], [27]:

- Number of Layers: Refers to the count of hidden layers within the NN architecture.
- Number of Neurons per Layer: Indicates the number of cells within each hidden layer.
- Number of Epochs: Refers to the number of complete passes through the training dataset.
- Batch Size: Denotes the number of training samples used in one forward/backward pass.
- Learning Rate: Determines the step size at each iteration while moving toward a minimum of the loss function.
- Optimizer: Updates the weights during training. Some of them are:
  - Stochastic Gradient Descent (SGD): Updates model weights by subtracting the gradient scaled by a learning rate.
  - Adaptive Gradient Algorithm (AdaGrad): Adjusts the learning rate for each parameter individually, decreasing it over time based on how frequently it's updated.
  - Adaptive Delta Algorithm (AdaDelta): Extends AdaGrad by avoiding its aggressive learning rate decay by using a moving window of gradient updates.
  - Root Mean Square Propagation (RMSprop): Improves AdaGrad by using an exponentially decaying average of squared gradients.
  - Adaptive Moment Estimation (Adam): Combines the benefits of RMSprop and momentum by tracking both the mean and variance of gradients.
- Activation Function: Introduces non-linearity, allowing the model to learn complex relationships. Here are some common choices:
  - TanH: Maps input values to the range [-1, 1] using a hyperbolic tangent function, providing symmetric output around zero:

$$\mathrm{TanH}(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

- Sigmoid: Transforms input values to the range [0, 1] using a logistic function, creating a smooth S-shaped curve:

$$Sigmoid(x) = \frac{1}{1+e^{-x}}$$

- ReLU: Applies a simple thresholding operation that outputs the input directly if positive, otherwise zero:

$$ReLU(x) = \max(0, x)$$

- Leaky ReLU: Extends ReLU by allowing a small positive slope for negative inputs:

$$LeakyReLU(x) = \begin{matrix} x\ if\ x \geq 0 \\ \propto x\ if\ x \leq 0 \end{matrix}$$

- Loss Function: Defines how the model's predictions are penalized, different loss functions guide the model toward varying trade-offs between bias and variance. The most used are:
  - Mean Squared Error (MSE): Calculates the average of squared differences between predicted and actual values:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2$$

  - RMSE: Takes the square root of MSE to return error measurements in the same units as the target variable:

$$RMSE = \sqrt{MSE}$$

  - Mean Absolute Error (MAE): Computes the average absolute difference between predicted and actual values:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y_i}|$$

  - Mean Absolute Percentage Error (MAPE): Expresses prediction errors as percentages of actual values:

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y_i}}{y_i}\right|$$

2.4.6 Regularization Techniques

Overfitting occurs when a model learns not only the underlying data patterns but also the noise present in the training set. This results in excellent performance on the training data but poor generalization to unseen data. To mitigate overfitting, several regularization techniques can be employed [12], [27]:

- Dropout: Deactivates a random subset of neurons during each training iteration.
- Early Stopping: Monitors model performance on a validation set during training and halts training once performance stops improving.
- L2 Regularization: Adds a penalty term to the loss function proportional to the square of the magnitude of the model's weights.
- Batch Normalization: Normalizes the activations of each mini-batch to have zero mean and unit variance.
- Recurrent Dropout: Applies dropout to the recurrent connections.
- Gradient Clipping: Imposes a maximum norm on the gradients during backpropagation.
- Data Augmentation: Expands the training dataset by applying transformations to the input data.

2.4.7 Evaluation

Accurate evaluation when building and training NNs is crucial to ensure the model generalizes well to new data and to compare different architectures and parameters. Evaluation metrics provide quantitative insights into how well a model approximates the true relationships in the data and inform decisions about optimization, model selection and overfitting regularization.

For regression tasks we have [12]:

- MSE: Quantifies prediction accuracy by averaging squared residuals.
- RMSE: Provides error measurement in original units by taking the square root of MSE.
- MAE: Measures average prediction error magnitude without considering direction.
- MAPE: Calculates percentage-based prediction errors relative to actual values.
- R Squared (R^2): Measures the proportion of variance in the dependent variable explained by the model, with values ranging from 0 to 1:

$$R^2 = 1 - \frac{\sum(y_i - \hat{y_i})^2}{\sum(y_i - \bar{y})^2}$$

For classifications tasks we have [12]:

- Accuracy: Calculates the proportion of correctly classified instances out of total instances:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- Precision: Measures the proportion of true positive predictions among all positive predictions:

$$Precision = \frac{TP}{TP + FP}$$

- Recall: Quantifies the proportion of actual positive cases correctly identified by the model:

$$Recall = \frac{TP}{TP + FN}$$

- F1 Score: Computes the harmonic mean of precision and recall:

$$F1 = 2\frac{Pre \cdot Rec}{Pre + Rec}$$

3. State-of-the-Art

This chapter presents a systematic literature review of NNs for stock price prediction. It begins with carefully defined search queries across multiple academic databases, followed by a multi-stage filtering process to identify the most relevant studies. The selected papers undergo comprehensive analysis, with key findings extracted and presented.

3.1 Defining Searches

Strategic keyword group development was the first step in creating the queries, these groups comprehensively represent the three critical dimensions of the research: methodology, domain, and task. The methodology keywords refer to the more general fields that can include NN models used in time-series forecasting, but also those specific models. The domain keyword is centred on stocks, ensuring that the retrieved literature specifically analyses stock prices. The task keywords focus on the objective of the studies. Table 3.1 shows the complete grouping of the keywords.

Methodology	Domain	Task
Artificial Intelligence (AI)		
Machine Learning (ML)		
Deep Learning (DL)		Prediction
Multilayer Perceptron (MLP)		
Artificial Neural Network (ANN)		
Recurrent Neural Network (RNN)	Stock	
Convolutional Neural Network (CNN)		

Long Short-Term Memory (LSTM)		Forecasting
Gated Recurrent Unit (GRU)		
Generative Adversarial Network (GAN)		
Transfomer (TFT)		

Table 3.1: Groups of keywords

In Table 3.2 we can see all the queries, where each query follows the general structure: ((“Methodology” OR “Methodology Abbreviation”) AND (“Domain”) AND (“Task 1” OR “Task 2”)).

Query 1	(“Artificial Intelligence” OR “AI”) AND (“Stock”) AND (“Prediction” OR “Forecasting”))
Query 2	((“Machine Learning” OR “ML”) AND (“Stock”) AND (“Prediction” OR “Forecasting”))
Query 3	((“Deep Learning OR “DL”) AND (“Stock”) AND (“Prediction” OR “Forecasting”))
Query 4	((“Multilayer Perceptron” OR “MLP”) AND (“Stock”) AND (“Prediction” OR “Forecasting”))
Query 5	((“Artificial Neural Network” OR “ANN”) AND (“Stock”) AND (“Prediction” OR “Forecasting”))
Query 6	(“Recurrent Neural Network” OR “RNN”) AND (“Stock”) AND (“Prediction” OR “Forecasting”))
Query 7	((“Convolutional Neural Network” OR “CNN”) AND (“Stock”) AND (“Prediction” OR “Forecasting”))
Query 8	((“Long Short-Term Memory” OR “LSTM”) AND (“Stock”) AND (“Prediction” OR “Forecasting”))
Query 9	((“Gated Recurrent Unit” OR “GRU”) AND (“Stock”) AND (“Prediction” OR “Forecasting”))
Query 10	((“Generative Adversarial Network” OR “GAN”) AND (“Stock”) AND (“Prediction” OR “Forecasting”))
Query 11	((“Transformer” OR “TFT”) AND (“Stock”) AND (“Prediction” OR “Forecasting”))

Table 3.2: Queries

Scopus, ArXiv and IEEE Xplore were the three search engines chosen for this project, due for being some of the most popular and relevant search engines in the

Computer Science academic community. To ensure relevance, accessibility, and recency, the following search parameters were defined:

- Type of Query: Title only.
- Type of Access: Open access only.
- Language: English only.
- Timeframe: From January 1st, 2022 to March 3rd, 2025 (day of search).

3.2 Initial Results

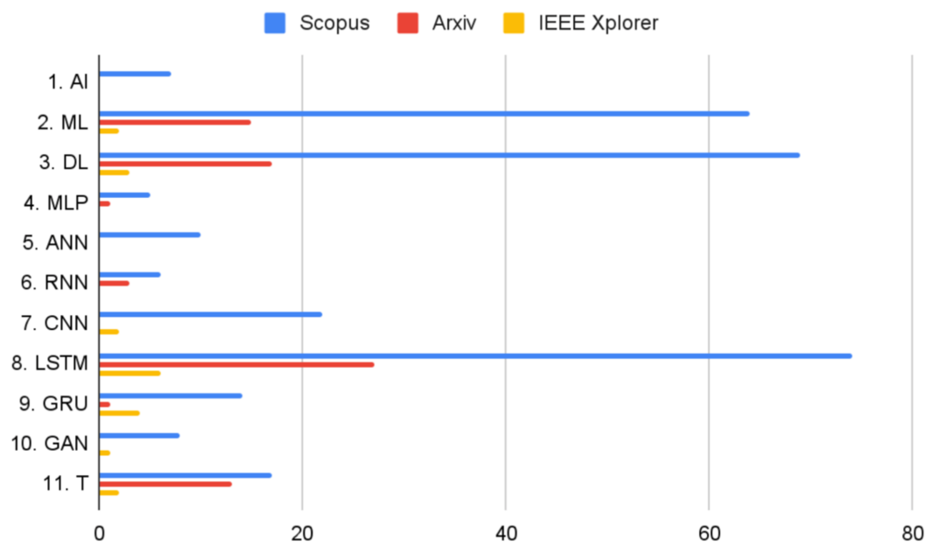

Figure 3.1: Distribution of retrieved papers per search engine and queries

In total, 393 articles were returned. As we can see in Figure 3.1, Scopus returned more articles than the other search engines in every single query, with 75% of the total number of articles, ArXiv with 20% and IEEE Xplore with only 5%. Query 8 (LSTMs) returned by far the most articles, with 27% of the total number articles, showing the prominence of LSTMs in stock price prediction. In terms of queries for specific models, only query 11 (Transformers) had more than 5% of the total number of hits, besides query 8. The more general queries, 3 (DL) and 2 (ML), followed query 8 in number of hits.

3.3 Filtering Process

To ensure that the selected literature was relevant to the defined task, a filtering process was applied to the initial 393 articles obtained. This process was conducted in three stages: first by title, then by abstract, and finally by full text. Each stage aimed to refine the selection by removing articles that did not align with the research focus. Several criteria were used to filter the articles:

- Some articles could be repeated, because they appeared on previous queries or in the same search engine, so they were removed.
- Systematic reviews are articles that critically evaluate and synthesize current research in a specific topic. These types of articles were excluded since they did not contribute original methodologies or experimental results.
- Articles that did not focus mainly on NNs were also filtered out, as they focused on traditional statistical methods or other ML techniques.
- Hybrid models, which combine different architectures and techniques (e.g., CNN-LSTM, GA-LSTM), were deemed outside the research's scope due to the lack of consensus on optimal configurations and the vast diversity in combination approaches across the literature. This research focuses specifically on traditional NN architectures, which refer to well-established models such as MLPs, RNNs, LSTMs, and GRUs. The use of traditional architectures enables a systematic methodology for both literature analysis and optimization phases, as they are easier to analyse, implement and configure, allowing for consistent evaluation criteria and reproducible results. In the comparative phase, using traditional NNs avoids the complexity of managing multiple interacting components, facilitating clearer performance comparisons across individual models. Also, establishing strong baselines with traditional models creates a solid foundation for future research, providing clear benchmarks against which more complex approaches can be meaningfully evaluated.
- Alternative models are custom non-traditional NN models. Articles proposing these types of models were removed for the same reasons as hybrid models.
- Natural Language Processing (NLP) refers to the field of AI that enables machines to understand and interpret human language. In the context of stock price prediction, NLP incorporates unstructured textual data, such as financial news articles or social media posts, into forecasting models. While it would be very relevant to use sentiment analysis of news articles and social media posts

to enhance the model's predictive power, the articles that focus on this technique were excluded for practical limitations regarding data access. The free APIs that provide this type of data would take too much time to get daily articles/posts for the several years used in the project.

- Articles with a different primary goal, such as market efficiency analysis, financial risk assessment or portfolio optimization, were also filtered.
- Some studies were removed because they were poorly explained, lacking sufficient details on methodology or results.
- Finally, a small number of articles were excluded for other miscellaneous reasons, primarily due to a mismatch between the semantic context of the keyword terms and the specific focus of the project.

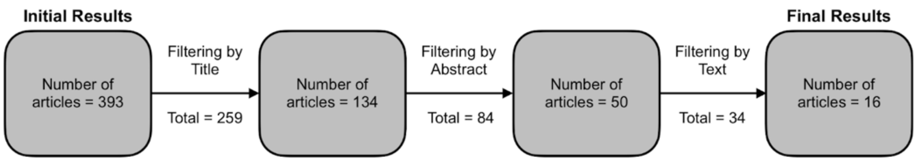

Figure 3.2: Systematic filtering process results

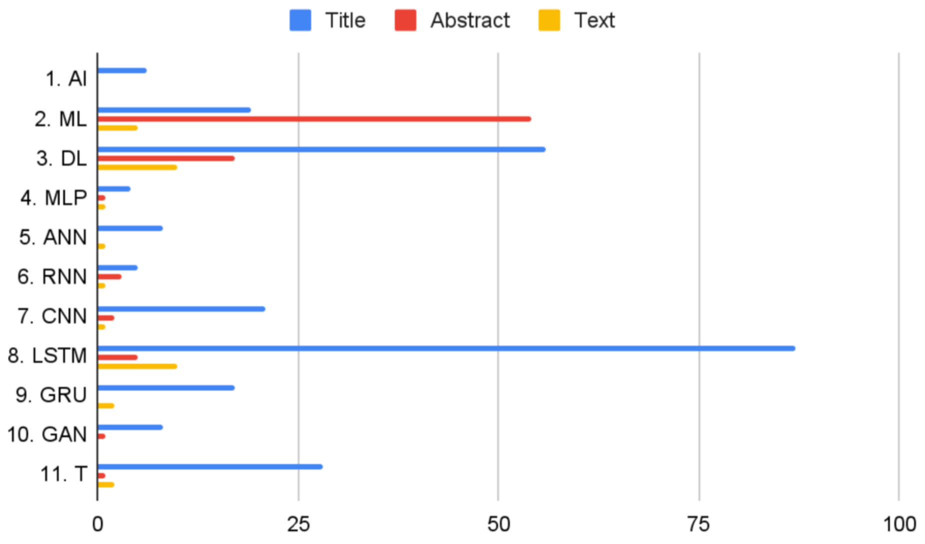

Figure 3.3: Distribution of filtered papers per stage of filtering and queries

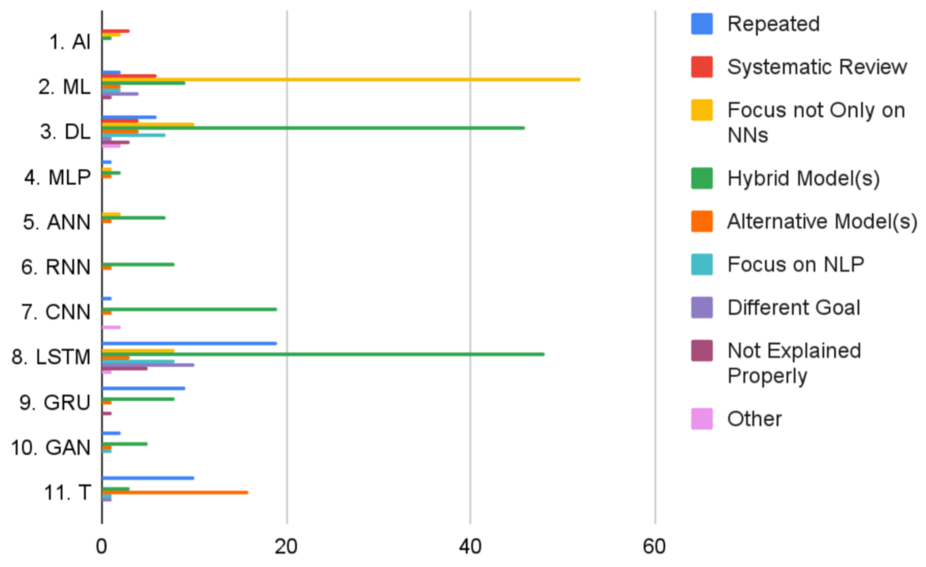

Figure 3.4: Distribution of filtered papers per criteria and queries

Most articles, as we can see by the Figures 3.2 and 3.3, were filtered by their title (66%), additionally 21% articles were filtered by the abstract, 9% by text, and only 4% were hits and deemed for article analysis: one from query 1 (AI), three from query 2 (ML), six from query 3 (DL), five from query 8 (LSTM) and one from query 11 (Transformer). Figure 3.4 demonstrates that the primary basis for article exclusion was their focus on hybrid models (41%), showing the relatively small exploration of traditional NNs in the academic space. Also, about 8% used alternative models, reinforcing this statement. The second biggest filter parameter was the “Focus not Only on NNs”, predictably so, since we have two queries (1 and 2) that are more general and returned articles that are AI/ML but don’t focus on NNs. We also had a lot of repeated articles (13%), since some of them didn’t focus only on one type of model (for example if they use LSTMs and GRUs, the article is going to show on queries 8 and 9), and different search engines returned the same articles.

For repeated articles and systematic reviews, only the title was needed to be read, to filter them. For most of hybrid, alternative and models that focused on NLP, the title was enough to filter them. The articles that focused not only on NNs, were the only ones where the abstract was the main source of filtration. Articles that were filtered because they had a different goal or weren’t explained properly, the text needed to be read.

3.4 Article Analysis

The article analysis is presented in chronological order of publication date. Papers sharing the same month and year (when only these were specified) were grouped and organized by query.

Orsel and Yamada [28] to predict future stock prices, applied the Kalman filter and various LSTM architectures and compared their performance. Tesla (TSLA) was chosen as an example of a highly volatile stock, while Microsoft (MSFT) was considered non-volatile. The key findings revealed that model accuracy was significantly affected by stock volatility. For low-volatility stocks like MSFT, the linear Kalman filter performed surprisingly well in predicting the next day's prices. However, for highly volatile stocks like TSLA, more complex LSTM architectures, particularly the CNN-LSTM and bidirectional LSTM, significantly outperformed the Kalman filter. Additionally, the study found that LSTM models trained on stocks with similar volatility levels could accurately predict both volatile (Russell Microcap) and non-volatile (S&P 500) market indices without retraining.

Alkhatib *et al.* [1] studied the effects of adding two new variables (High - Low and Open - Close) to the traditional set of features. It also analyses the impact of dataset size from different companies on forecasting accuracy. Additionally, it compares the performance of six NNs: MLP, GRU, LSTM, Bi-LSTM, CNN, and CNN-LSTM, in the forecasting task. The use of the new set of six features generally improved model performance, resulting in lower losses compared to the traditional 4-feature approach. There was no model that consistently outperformed all others across all datasets and metrics. The dataset size seemed to influence performance, with small datasets presenting varied results. The volatility of stock prices also affected the results, generally showing higher losses.

He and Dai [3] proposed a model to predict price variations of five stocks in a sub-network of the stock market's securities board. The model integrates Independent Component Analysis (ICA) and an LSTM model to analyse trading noise and improve prediction accuracy. The experimental results demonstrated that the proposed model outperformed benchmark approaches, including a LSTM model without ICA, as well as other hybrid and traditional models.

Montesinos *et al.* [24] assessed the impact of feature selection and hyperparameter optimization on prediction quality using historical data from NIFTY 50 and NNs. The study tested different feature combinations (Open, High, Low, Close, and Volume), implemented three neural network architectures (RNN, LSTM, and CNN), and evaluated three activation functions (Sigmoid, ReLU, and Softmax). With the optimal configuration utilizing ReLU for RNN and LSTM models, and Sigmoid for CNN models. In the test dataset, the LSTM model yielded the best results when trained with the features High, Low, Open, and Close, using 500 epochs.

Khalil and Bakar [29] presented a comparative study of NNs for univariate and multivariate prediction of the Malaysian stock market. The study uses three models: MLP, CNN, and LSTM. The task consisted of predicting the closing price of stocks for 1, 3, 5, and 7 days in the future. Although univariate models presented slightly fewer errors, the difference compared to multivariate models was considered statistically insignificant in most cases, suggesting that multivariate prediction remains competitive and benefits from the inclusion of more information. The LSTM model showed the best performance, achieving the lowest prediction error among the tested models.

Choi and Choi [30] presented a LSTM model that predicts airline stock prices. The training results showed that the loss function value converged close to zero and that the MSE measured in the model's accuracy with the test data was very low. The study concluded that the trained LSTM model was able to follow the trend of real values with high precision.

Ifleh and Kabbouri [31] applied a correlation-based feature selection model to identify important technical indicators, which are combined with multiple NNs to predict stock market indices. The principle was to select variables correlated with the dependent variable and not correlated with each other. The NNs used were ANN, CNN and LSTM. In general, using all variables as inputs performed better with the ANN model in almost all markets. Predictions using CNN and LSTM combined with correlation-selected variables outperformed predictions using all variables in almost all indices.

Paul and Das [32] studied the application of NNs to predict trends in the Indian stock market, with specific focus on the NIFTY 50. Three models were implemented and compared: RNN, LSTM, and BiLSTM. Various configurations of each model were tested, varying the number of neurons, batch size, and epochs. The LSTM model demonstrated the best performance in predicting the closing movement of the NIFTY 50 index.

Razouk *et al.* [10] investigated the application of ANNs and technical indicators, to predict the Moroccan All Shares Index (MASI). To identify the ten most relevant indicators, feature selection was done based on prediction accuracy. The model trained with this set of reduced features showed performance improvements compared to a model trained with the total set of features.

Kumar *et al.* [18] explored the application of LSTMs to predict the price of the S&P 500 Healthcare index. A crucial aspect of the study is the use of Explainable Artificial Intelligence (XAI) techniques, specifically the Local Interpretable Model-Agnostic Explanations (LIME) method, to interpret the predictions of the LSTM model. The “Average Close” variable was found to be the dominant feature, with a positive influence on predictions for 2018 and 2019 but a negative influence for 2017.

Kenrick *et al.* [33] investigated the potential of LSTM models to predict the opening and closing prices of six Indonesian banks (three national and three private). The error rates for the predicted open prices are lower and the accuracy is higher compared to the predicted close prices. The results demonstrated exceptional accuracy and very low error rates for predicting opening and closing prices.

Hartanto and Gunawan [34] studied the use of the Temporal Fusion Transformer (TFT) model for short-term stock price prediction on the Indonesia Stock Exchange (IDX). The results demonstrated that the TFT significantly outperformed the baseline models, showing better predictive accuracy. The study also analysed the importance of encoder and decoder variables for each stock, showing that these vary, which underscores the need for customized approaches.

Makinda [35] investigated the effectiveness of two optimization techniques, Adam and Nesterov Accelerated Gradient (NAG), in training LSTMs and GRUs for stock market time series forecasting. The results demonstrated that the GRU model optimized with Adam achieved the lowest RMSE, outperforming the other combinations in accuracy. In general, GRU models performed better than LSTM models, and the Adam optimizer outperformed NAG for both network architectures.

Xiao *et al.* [5] compare the application of LSTM, GRU, and Transformer models, to predict the TSLA stock prices. The main prediction task consisted of predicting the trend of TSLA stock price for the next 30 days. The results of the model evaluation demonstrated that the LSTM model achieved the highest accuracy, and proved to be more consistent with the actual stock price trend compared to the GRU and Transformer models.

Teixeira [36] analysed 40 years of AAPL data, supplemented with major financial indices. Feature engineering added 43 variables, including technical indicators and economic data. The study compared: RNN, LSTM, GRU, CNN, Extreme Gradient

Boosting (XGBoost), and hybrid models. Models were evaluated using Time Series Cross-Validation (10 folds). GRU and XGBoost performed the best, with GRU excelling in MSE & MAPE and XGBoost in MAE, RMSE and R². RNN and CNN had variable performance, while GRU-based hybrids improved results. LSTM performed well but was outperformed by GRU and XGBoost.

Dwiandiyanta *et al.* [37] evaluated the effectiveness of combining NNs (RNN, LSTM, and GRU) with technical indicators for predicting blue-chip stock prices on the IDX. The NNs were trained and tested using two approaches: with market variables only and with a combination of market variables and technical indicators. The results demonstrated that, in general, models that integrated technical indicators performed better than models based solely on market data. Interestingly, the results suggested that, in some cases, the RNN model performed comparably or better than LSTM and GRU models when combined with technical indicators, indicating that feature selection can mitigate some limitations of simpler architectures.

Article	Stock(s)	Years of Data	Variable(s)	PreProcessing
[28]	TSLA	Jan ‘11 - Dec ‘22 (10 years)	1 market variable	Normalization
2022	MSFT			Split 75-15
[1]	AAPL	It changes from stock to stock, ending always on Oct, Nov or Dec ’21 (average 14 years)	4 market variables	Normalization
2022	TSLA			Split 70-15-15
	SNAP		High - Low	
	XOM		Open - Close	
[3]	SWHI	Jul ‘15 - Dec ‘20 (5.5 years)	4 market variables	Remotion of null and wrong values
2022	GUOSEN			Purification and smoothing
	DFZQ			
	DXZQ			ICA
	GTJA			Split 80-20

[24] 2022	NIFTY50	Jan ‘14 - Dec ‘18 (5 years)	5 market variables	Replacement of raw numerical attribute values with intervals Filling missing values Remotion of duplicates Normalization Split 80-20
[29] 2023	AXIATA PETGAS	Jan ‘17 - Dec ‘21 (5 years)	6 market variables	Imputation of missing values Correction of wrong values Remotion of “Adj Close” from the multivariate dataset Normalization
[30] 2023	AAPL	Jan ‘16 - Apr ‘23 (7.5 years)	6 market variables Change	Replacement with the average for missing or null values Normalization Split 70-30
[31] 2023	MASI MADEX EGX30 FTSE100 SP500 NASDAQ100	It changes from stock to stock, ending always on Apr ’21 (average 22 years)	2 market variables 29 technical indicators	Correlation for feature selection (for each stock remained between 8 and 15 features) Split 80-20
[32] 2023	NIFTY50	Jan ‘97 - Dec ‘21 (25 years)	4 market variables	Remotion of missing and wrong values Split 70-30

[10]	MASI	Jan ‘09 - Dec ‘18 (10 years)	1 market variable	Top 10 features were found through modelling
2023			26 technical indicators	Normalization
				Split 80-20
[18]	SP500-35	Jan ‘17 - Dec ‘19 (3 years)	4 market variables	Normalization
2024			Google searches volume	Conversion to time-series (14 time step)
				Split 90-10
[33]	BBNI	Apr ‘19 - April ‘24 (5 years)	6 market variables	Imputation of missing values
2024	BBRI			Normalization
	BMRI			Split 85-15
	BNGA			
	BBCA			
	NISP			
[34]	ANTM	Mar ‘19 - Mar ‘24 (5 years)	2 market variables	Remotion of irrelevant variables
2024	EXCL		Open - Close	Variance inflation factor
	ASII		Months	Split 90-10
			Working days	
[35]	AAPL	Jan ‘14 - Aug ‘24 (10.5 years)	1 market variable	Imputation of missing values with the average or natural interpolation
2024				Normalization
				Split 70-15-15
[5]	TSLA	Jan ‘15 - Jan ‘24 (10 years)	5 market variables	Treatment of missing values and outliers
2024				Data enhancement
				Time series differentiation

[36]	AAPL	Sep ‘84 - Sep ‘24 (41 years)	6 market variables	Top 20 variables were chosen using correlation and SelectKBest
2025			27 technical indicators	Normalization
			12 interest rate indicators	Split 80-20
			5 credit spread indicators	
			4 indices indicators	
			Crude oil prices	
[37]	BBCA	Jan ‘16 - Dec ‘22 (7 years)	5 market variables	Remotion of missing values
2025	BBRI		5 technical indicators	Handlement of outliers
	BMRI			Normalization
	BBNI			Split 70-15-15
	TLKM			
	ASII			

Table 3.3: Data characteristics and preprocessing methods summary

Article	Model(s)	Hyperparameters and Task	Results
[28]	LSTM1	1 layer w/ 64 neurons	2.97 RMSE
2022		Predict “Close” for the next day	2.19 MAE
			0.96 R
	LSTM2	2 layers w/ 64 neurons	13.66 RMSE
		Predict “Close” for the next day	10.90 MAE
			0.78 R
	BiLSTM	1 layer w/ 64 neurons	2.64 RMSE
		Predict “Close” for the next day	1.96 MAE
			0.97 R

[1]	MLP	1 hidden layer w/ 100 neurons	30.87 MSE
2022		ELU | Adam	4.94 MAPE
		2 batch size	
		100 epochs	
		Predict “Close”	
	CNN	4 hidden layers	21.70 MSE
		ELU | Adam	5.52 MAPE
		4 batch size	
		100 epochs	
		Predict “Close”	
	LSTM	2 hidden layers w/ 32 and 16 neurons	28.38 MSE
		ELU | Adam	4.08 MAPE
		2 batch size	
		100 epochs	
		Predict “Close”	
	BiLSTM	2 hidden layers w/ 32 and 16 neurons	7.04 MSE
		ELU | Adam	3.63 MAPE
		2 batch size	
		100 epochs	
		Predict “Close”	
	GRU	2 hidden layers w/ 50 and 25 neurons	12.52 MSE
		ELU | Adam	2.66 MAPE
		2 batch size	
		70 epochs	
		Predict “Close”	
[3]	LSTM	2 hidden layers w/ 64 and 10 neurons	0.27 RMSE

2022			0.08 MSE
		25% dropout	0.24 MAE
		TanH | Adam	
		1 batch size	
		30 time window size	
		30 epochs	
		Predict “Open” for the next day	
[24]	RNN	2 layers	0.007 RMSE
2022		2 dropout layers	0.006 MSE
		ReLU	0.004 MAE
		500 epochs	0.63 R²
		Predict “Close” for the next day	
	CNN	3 layers	Not found
		Sigmoid	
		500 epochs	
		Predict “Close” for the next day	
	LSTM	2 layers	0.04 RMSE
		2 dropout layers	0.05 MSE
		ReLU	0.02 MAE
		500 epochs	0.24 R²
		Predict “Close” for the next day	
[29]	MLP	2 hidden layers w/ 200 neurons	0.12 RMSE
2023		ReLU | Adam | MSE	0.08 MAE
		244 batch size	1.18 MAPE
		150 epochs	
		7 time window size	
		0.001 learning rate	
		Predict the “Close” for the next day	

	CNN	2 hidden layers w/ 256 neurons	0.12 RMSE
		ReLU | Adam | MSE	0.09 MAE
		32 batch size	1.20 MAPE
		7 time window size	
		0.001 learning rate	
		Predict the "Close" for the next day	
	LSTM	2 hidden layers w/ 100 neurons	0.11 RMSE
		ReLU | Adam | MSE	0.08 MAE
		32 batch size	1.11 MAPE
		100 epochs	
		7 time window size	
		0.001 learning rate	
		Predict the "Close" for the next day	
[30]	LSTM	2 layers w/ 32 neurons	0.0004 MSE
2023		25% dropout	
		TanH | Adam | MSE	
		2 batch size	
		7 time window size	
		50 epochs	
		Predict "Close" for the next day	
[31]	MLP	2 hidden layers	289.03 RMSE
2023		100 epochs	84,598.27 MSE
		Predict "Close"	195.90 MAE
			0.04 RMSLE
			0.003 MSLE
	CNN	2 hidden layers	185.59 RMSE
		100 epochs	28,964.67 MSE

		Predict "Close"	120.03 MAE
			0.02 RMSLE
			0.0005 MSLE
	LSTM	1 hidden layer	459.06 RMSE
		100 epochs	91,383.54 MSE
		Predict "Close"	267.06 MAE
			0.04 RMSLE
			0.004 MSLE
[32]	RNN	4 layers w/ 64 neurons	1.35 MAPE
2023		20% dropout	0.98 PA
		64 batch size	
		15 time window size	
		50 epochs	
		Predict "Close"	
[32]	RNN	4 layers w/ 64 neurons	1.35 MAPE
2023		20% dropout	0.98 PA
		64 batch size	
		15 time window size	
		50 epochs	
		Predict "Close"	
	LSTM	Stacked layers w/ 32 neurons	0.90 MAPE
		10 batch size	0.99 PA
		15 time window size	
		50 epochs	
		Predict "Close"	
	BiLSTM	layers w/ 64 neurons	3.73 MAPE
		32 batch size	0.96 PA
		20 time window size	

		50 epochs Predict “Close”	
[10] 2023	MLP	10 layers w/ 26 neurons Predict “Close”	1.70 RMSE 0.03 MSE 1.30 MAE
[18] 2024	LSTM	2 layers w/ 64 and 32 neurons 1 20% dropout layer ReLU | Adam | MSE 16 batch size Early stopping 25 epochs Predict “High” 15 days from now	7.23 RMSE 297.11 MSE
[33] 2024	LSTM	1 layer w/ 200 neurons 1 20% dropout layer Adam 20 time window size 100 epochs Predict “Close”	0.02 RMSE 0.00 MSE 0.01 MAE 0.94 R^2
[34] 2024	TFT	12 time window size Predict “Close” for the next 3 days	17.10 MAE 0.25 MAPE 0.004 SMAPE
[35] 2024	LSTM	Adam 1 batch size 10 epochs 0.001 learning rate Predict “Close”	176.62 RMSE
	GRU	Adam	172.40 RMSE

		1 batch size	
		10 epochs	
		0.001 learning rate	
		Predict “Close”	
[5]	LSTM	Predict “Close”	26.09 RMSE
2024			260.87 MSE
			12.78 MAE
			0.98 R²
	GRU	Predict “Close”	18.43 RMSE
			339.90 MSE
			14.73 MAE
			0.84 R²
	T	Predict “Close”	16.39 RMSE
			360.28 MSE
			16.29 MAE
			0.80 R²
[36]	RNN	3 layers w/ 64, 64 and 160 neurons	14.07 RMSE
2025		3 10% dropout layers	203.15 MSE
		Linear | Bayesian	11.00 MAE
		100 time window size	8.00 MAPE
		Predict “Adj Close” for the next day	0.92 R
	CNN	4 layers	24.67 RMSE
		Linear | Bayesian	625.07 MSE
		100 time window size	19.67 MAE
		Predict “Adj Close” for the next day	13.97 MAPE
			0.77 R²
	LSTM	2 hidden layers w/ 256 neurons	6.30 RMSE
		2 10% dropout layers	40.44 MSE

		Linear | Bayesian	4.47 MAE
		100 time window size	3.67 MAPE
		Predict “Adj Close” for the next day	0.98 R²
	GRU	2 hidden layers w/ 196 and 256 neurons	8.32 RMSE
		2 10% dropout layers	70.33 MSE
		Linear | Bayesian	6.05 MAE
		100 time window size	4.52 MAPE
		Predict “Adj Close” for the next day	0.97 R²
[37]	RNN	2 hidden layers w/ 64 neurons	0.02 RMSE
2025		Adam	0.0008 MSE
		64 batch size	0.92 R²
		0.1, 0.01, 0.001, 0.0001 learning rate	
	LSTM	2 hidden layers w/ 64 neurons	0.03 RMSE
		Adam	0.001 MSE
		64 batch size	0.92 R²
		0.1, 0.01, 0.001, 0.0001 learning rate	
	GRU	2 hidden layers w/ 64 neurons	0.03 RMSE
		Adam	0.001 MSE
		64 batch size	0.92 R²
		0.1, 0.01, 0.001, 0.0001 learning rate	

Table 3.4: Models and results summary

As Table 3.3 shows, for the analysed articles, the most common stocks (or stock indices) were AAPL, used 4 times, and TSLA used 3 times. About half of the articles analysed only 1 stock, three articles used 2 stocks, and the same number of articles used 6 stocks. The daily historical data used ranges from 3 years to 41 years, with the articles using on average 11.5 years. Three articles explore the use of univariate models.

On average, the articles used 11 variables. Every study used between 1 and 6 market variables, using on average 4 market variables. In total, four articles used technical indicators and five used other types of variables. In terms of preprocessing methods, the most mentioned were normalization and the split for training and testing (and sometimes validation), with authors using on average the 80-20 split. Nine articles used methods to deal with missing, null, or wrong data, and four articles performed some type of feature selection.

As we can see from Table 3.4, the most common model was LSTM, used in almost every single article, followed by GRU, used in almost half of the articles. Nearly every model uses stacked hidden layers, with 2 being the most used number of layers, with only two articles using deeper models (more than 3 layers). Nearly half of the models mention the use of dropout. ReLU is the most used activation function, Adam the most used optimizer, and MSE the most used loss function. Almost every article tries to predict the variable "Close" for the next day (some don't specify the timeframe they're trying to predict, but it's assumed to be the next day). Some articles mention the batch size, time window size, number of epochs, and learning rate. Nearly none mention the use of overfitting regularization measures, with only dropout layers being employed in half of the articles. The most used evaluation metrics were RMSE, MSE, and MAE. GRU is the model that performed best across the multiple studies, but LSTM has competing results. The articles analysed showed a big dispersion in results from 0.0004 RMSE to 459.06 RMSE.

3.5 Research Direction

The overall goal of this literature review is to identify and analyse the most used and best performing: stocks, dataset sizes, variables, preprocessing methods, NN models and hyperparameters for stock price prediction in the current research landscape. The information found will be crucial for guiding the experimental phase of this research, where the models will be implemented, compared, optimized and evaluated. While the article analysis reveals important patterns and trends in the field, it also exposes a critical lack of consensus across studies and highlights numerous gaps in the current research. These inconsistencies and unexplored areas present significant opportunities for investigation and raise fundamental questions that need to be addressed:

- Model Performance Comparison: There is no clear consensus on which NN architecture performs best. While GRU shows promising signs and LSTM demonstrates competing results with higher usage rates, determining the truly superior model remains unresolved, particularly regarding how different models perform with specific variable combinations.
- Univariate-Multivariate Analysis: The debate between univariate and multivariate approaches remains unresolved, with both showing competitive results in different contexts, warranting further clarification.
- Technical Indicators Impact: While technical indicators generally appear to improve performance, their effectiveness varies across studies.
- Preprocessing Methodology: Authors generally don't evaluate preprocessing methods, leaving valuable comparisons between different techniques unexplored.
- Hyperparameter Optimization: Although some studies compare basic model configurations like layers and neurons, critical hyperparameters such as activation functions, loss functions, and optimizers remain largely unexamined.
- Regularization Techniques: The literature shows a notable absence of overfitting regularization measures, despite their potential importance for model generalization.
- Stock-Specific Performance: There is a lack of analysis comparing model performance across diverse stocks, particularly with large stock datasets, leaving questions about which stock characteristics (sector, growth patterns, volatility levels, etc.) are most suitable for NN prediction
- Real-World Application: There is an absence of analysis regarding the practical applicability of NNs for real-world investing decisions, as authors focused primarily on regression metrics without exploring critical time constraints (such as whether a model's execution time aligns with the prediction timeframe required for actionable investment decisions).

To address these gaps, this project aims to conduct a comprehensive empirical comparison of almost every aspect of using NNs for stock price prediction. Using the findings from this literature analysis as a guideline, the research will systematically examine variables, models, preprocessing methods, hyperparameters, and regularization techniques while answering the questions that emerged during the state-of-the-art review. The final optimized models will undergo thorough evaluation to determine whether traditional NNs provide genuine value for real-world stock investing and to identify which stock sectors and characteristics are most suitable for these modelling approaches. This work creates a robust methodological foundation that clarifies the fundamental performance characteristics and practical applicability of NNs architectures on the challenging task of stock price prediction.

4. Methodology

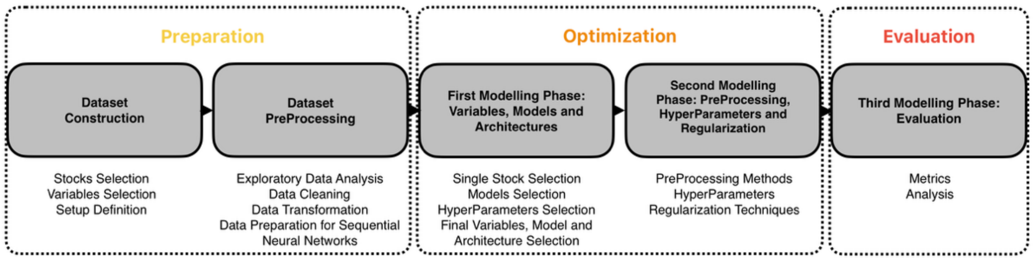
Preparation
Optimization
Evaluation
Dataset Construction
Dataset PreProcessing
First Modelling Phase: Variables, Models and Architectures
Second Modelling Phase: PreProcessing, HyperParameters and Regularization
Third Modelling Phase: Evaluation
Stocks Selection
Variables Selection
Setup Definition
Exploratory Data Analysis
Data Cleaning
Data Transformation
Data Preparation for Sequential Neural Networks
Single Stock Selection
Models Selection
HyperParameters Selection
Final Variables, Model and Architecture Selection
PreProcessing Methods
HyperParameters
Regularization Techniques
Metrics
Analysis

Figure 4.1: Methodology pipeline

This chapter describes the methodological framework of the project, which follows a structured multi-phase approach as illustrated in Figure 4.1. The methodology begins with comprehensive dataset preparation, including stock selection, variables identification, and data preprocessing. The consequent steps systematically optimize each component of the modelling pipeline using a single representative stock, progressively refining variable selection, model architecture, preprocessing methods, and hyperparameters. Once the optimal configuration is identified, it is validated across multiple S&P 500 stocks to assess generalizability and performance patterns.

4.1 Dataset Construction

The dataset construction process involves several key components: systematic selection of stocks to ensure broad market representation, strategic identification and categorization of predictive variables, and establishment of suitable parameters for data collection.

4.1.1 Stocks Selection

To effectively validate and test the NN model for stock price prediction, a comprehensive group of stocks is required with: broad market representation to capture diverse trading patterns, sectoral variety to expose models to different business cycles and market sensitivities, and a range of company sizes to ensure generalizability across various market capitalizations.

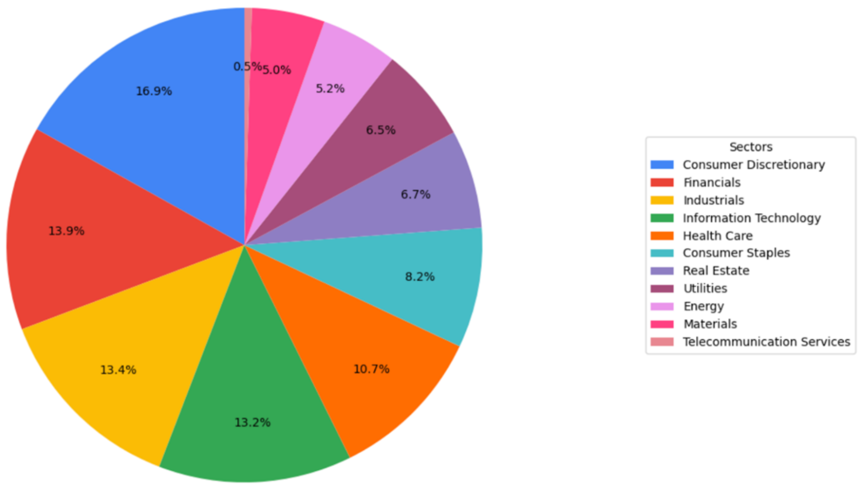

Figure 4.2: Distribution of sectors in the S&P 500

The S&P 500 index was therefore selected as the stock universe, due to its comprehensive representation of the U.S. equity market, encompassing approximately 500 large-cap companies across all major economic sectors including Consumer Discretionary, Financials, Industrials, Information Technology, and Health Care, as illustrated in Figure 4.2. Although all constituent companies qualify as large-cap enterprises, they exhibit substantial variation in market capitalization, ranging from companies near the $5 billion minimum threshold to mega-cap corporations with market values exceeding $1 trillion. This size diversity, combined with the index's sectoral breadth, ensures exposure to companies with distinct market dynamics, varying sensitivity to economic cycles, and diverse corporate performance characteristics shaped by their respective industries, business models, and market positions.

To ensure an even better comprehensive model validation, the S&P 500 composition as of July 2020[3] was selected. This 5-year historical reference point allows for the inclusion of companies that may have since been removed from the index for various reasons, providing valuable insights into the model's performance across

[3] https://kaggle.com/datasets/paytonfisher/sp-500-companies-with-financial-information

different market scenarios. Importantly, this timeframe captures stocks that have experienced significant price declines in the subsequent years, enabling the model to be tested on companies with diverse performance trajectories, including those with sustained downward trends. This approach helps mitigate the survivorship bias by deliberately including both high-performing and underperforming stocks, making the evaluation process more realistic and reflective of actual market diversity. However, eliminating this bias remains challenging, as some stocks have been delisted in the meantime or may no longer be accessible through the API used.

4.1.2 Variables Selection

A critical component of this project is the systematic selection and evaluation of input variables for the predictive models. The variables were selected based on their prevalence in the literature analysis and organized into distinct categories to enable isolated assessment and comparative analysis of each group's predictive contribution during the subsequent optimization phases.

The first category consists of univariate models that rely solely on the closing price as the input variable. Despite their simplicity, these models have shown promising results in stock price forecasting. Specifically, 3 out of the 16 academic articles reviewed employed univariate approaches, and the results were generally favourable, demonstrating that even models based on a single variable can yield competitive forecasting performance.

The second category is composed of market variables[4]. Market variables are the most prevalent type of variables for stock price prediction, with 100% of the academic papers analysed in the state-of-the-art review incorporating this type of features. These variables capture the fundamental price and volume dynamics that drive stock movements and include: Open, Close, High, Low, and Volume.

Technical indicators[5] form the third category. Technical indicators represent the second most used type of variables, utilized in 25% of the academic papers analysed in the state-of-the-art. These derived metrics transform raw price and volume data into meaningful signals that capture market dynamics. The technical indicators selected correspond to those employed in the academic papers reviewed [10], [31], [36], [37]:

[4] https://support.google.com/docs/answer/3093281
[5] https://technical-analysis-library-in-python.readthedocs.io/en/latest/ta.html

- Trend: SMA, EMA, MACD, ADX, PSAR.
- Momentum: RSI, RC, SOK, Momentum, Williams %R, TRIX, CMO.
- Volatility: BB, SD.
- Volume: OBV, AD, MFI.
- Statistical: CCI, BOP.

4.1.3 Setup Definition

To ensure a fair and consistent experimental setup, clear design choices regarding the dataset's historical span, sampling frequency, and prediction task must be established. These decisions are guided both by practices observed in the state-of-the-art literature and by the specific goals of this work.

Regarding the dataset size, the average historical span used in the studies analysed through the state-of-the-art is approximately 11.5 years, with durations ranging from 3 to 40 years. However, a 5-year historical window was selected for this work, from June 1st, 2020, to May 30th, 2025. Unlike most of the analysed studies which typically consider between 1 and 6 stocks, this research aims to train and evaluate the model on a much broader universe of stocks. Furthermore, the extensive model optimization process required for this comprehensive evaluation across multiple configurations makes a shorter dataset more practical. Additionally, the 5-year span is a popular choice within the literature, with 5 out of the 16 reviewed articles adopting a historical window of around that number. Using a 5-year window offers a practical balance between capturing relevant market cycles, economic regimes, and company performance variations, while ensuring computational feasibility when dealing with such a large and diversified dataset.

As for the sampling frequency, daily data was chosen, following the standard across all studies reviewed. Daily frequency allows capturing short-term price movements, volatility patterns, and the dynamic effects of market, macroeconomic, and company-specific factors. Furthermore, this frequency fits the goal of this project, which is to predict the next day's price using all information available up to the current day, allowing for practical day ahead forecasts.

This task is framed as a regression problem, as in all state-of-the-art studies analysed, where the objective is to predict the continuous numerical value of the next day's closing price rather than classifying movements into discrete categories. Regression provides the actual magnitude of price changes, which is essential for effective model optimization and evaluation in financial forecasting. RMSE was chosen as the primary evaluation metric for assessing model performance, as it is widely adopted in time-series forecasting. RMSE penalizes larger prediction errors more heavily than smaller ones, making it particularly well-suited for financial forecasting. This choice is supported by the literature review, where RMSE appeared in 12 out of the 16 analysed articles, making it the most frequently used metric, closely followed by MSE in 11 articles.

4.2 Dataset PreProcessing

The dataset preprocessing phase encompasses a comprehensive pipeline designed to transform raw financial market data into a clean, structured format suitable for NNs. This critical stage involves four main components: EDA to understand underlying patterns and statistical properties, data cleaning to ensure quality and consistency by handling missing values, data transformation to normalize features, and data preparation to structure the time series data into appropriate formats for modelling.

4.2.1 Exploratory Data Analysis

A comprehensive data analysis is important to understand the underlying patterns, relationships, and statistical properties of the S&P 500 dataset before model development. This analysis examines variable correlations to identify redundancies and complementary information, profiles the statistical characteristics and heterogeneity across individual stocks and sectors, and investigates the temporal dependencies and non-stationary properties inherent in financial time series data that present fundamental challenges for predictive modelling.

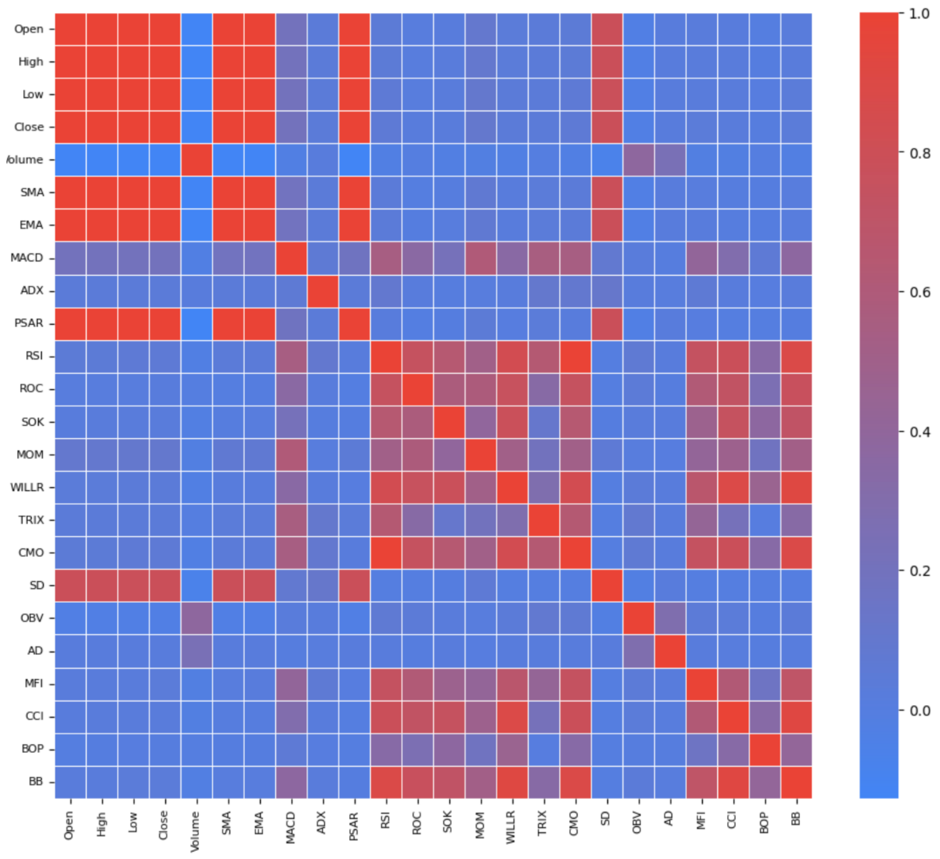

Figure 4.3: Correlation between variables

Looking at the correlation heatmap in Figure 4.3, the market price variables (Open, High, Low, Close) exhibit extremely strong positive correlations with each other, approaching perfect correlation, which indicates that these variables capture essentially the same underlying movement information. Volume demonstrates a distinctly different correlation pattern compared to the other market variables, suggesting it captures complementary information about market activity and liquidity rather than pure price dynamics. The technical indicators display more heterogeneous correlation structures, with correlations ranging from strong positive to moderate values across different indicator pairs. Trend metrics (SMA, EMA, PSAR) naturally show high correlations with price variables since they are direct derivatives of historical prices, while momentum oscillators (RSI, RC, SOK, Momentum, Williams %R, CMO) shows strong correlations with BB, CCI and MFI since all of them measure change.

For each stock, seven key financial metrics were defined to provide a comprehensive statistical analysis: price, volume, volatility, seasonality, Sharpe ratio and maximum drawdown.

The price distribution shows a highly right-skewed pattern with a mean of \$173.14 but a median of only \$96.28, indicating that a relatively small number of high-priced stocks pull the average significantly higher than the typical stock price. At the extremes, AZO commands the highest price at \$3,733.04, while XRX trades at just \$4.89, representing a price differential of over 760x. This skewness is even more pronounced in volume data, where the mean volume of 9.25 million shares vastly exceeds the median of 4.31 million shares. NVDA dominates trading activity with over 333 million shares, while AZO shows the lowest trading volume at just 191,000 shares, highlighting the vast disparity in market participation across index constituents.

Growth performance spans an extraordinary range from NVDA's exceptional 72.7% growth to VFC's severe decline of -26.5%, with a mean growth of 9.6% matching historical S&P 500 expectations. The growth distribution demonstrates that while 75% of stocks delivered positive returns (given the 25th percentile at 1.4%), a significant minority experienced substantial declines, illustrating the divergent fortunes within the index during this period. Volatility characteristics show FL exhibiting the highest daily volatility of 3.5% compared to JNJ's remarkably stable 0.8%, though the relatively tight distribution around the mean of 1.6% suggests most stocks cluster within a typical volatility band. Seasonality patterns also vary, with WYNN showing the highest seasonal effects at 24.7% compared to FTI's minimal seasonal influence of just 0.2%, though the mean seasonality of 4.2% indicates relatively modest seasonal patterns across most stocks.

The Sharpe ratio distribution, with both mean and median at 0.02, indicates generally positive risk-adjusted returns, though individual stocks range from PWR's exceptional 0.07 to BAX's poor -0.03. Maximum drawdown statistics reveal perhaps the most sobering aspect of equity risk, with average maximum declines of -45.9% and catastrophic cases like KSS experiencing severe losses of -90.4%, while the most resilient stocks like WM limited maximum drawdowns to just -17.0%. These findings demonstrate that even within the supposedly stable S&P 500 index, individual constituents can exhibit vastly different risk-return profiles, from growth champions and defensive stalwarts to speculative positions and distressed situations.

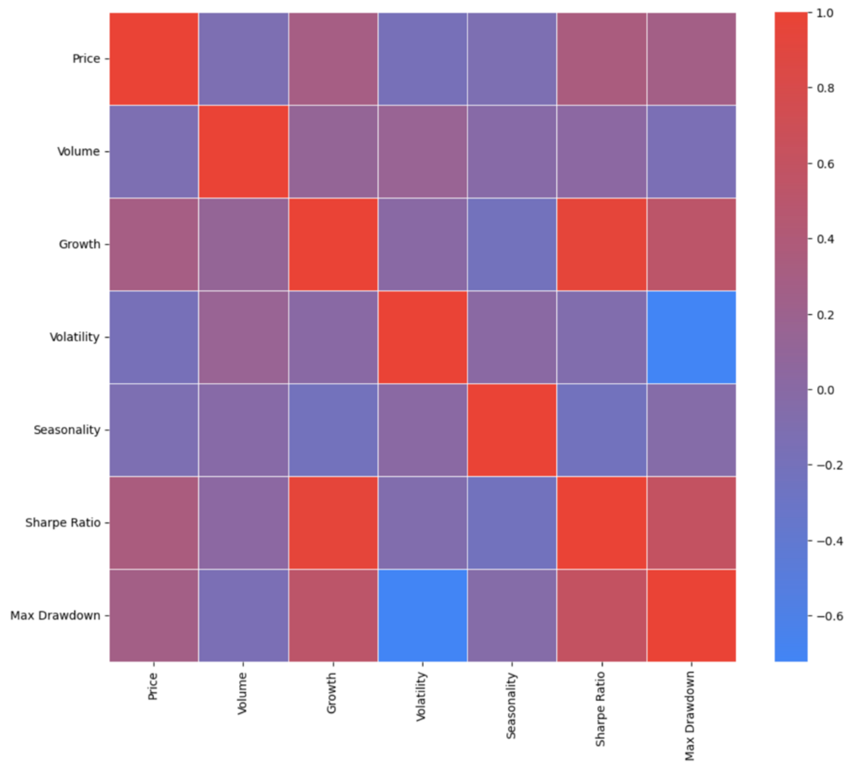

Figure 4.4: Correlation between metrics

Figure 4.4 shows the correlation matrix between the seven key financial metrics. As expected, volatility and maximum drawdown demonstrate a strong negative correlation, reflecting the fundamental relationship between risk and potential losses. Growth shows positive correlation with Sharpe ratio, indicating that higher-growth stocks tend to deliver better risk-adjusted returns. Notably, seasonality appears relatively independent from other financial metrics, with minimal correlation to traditional risk-return measures, suggesting it captures unique patterns.

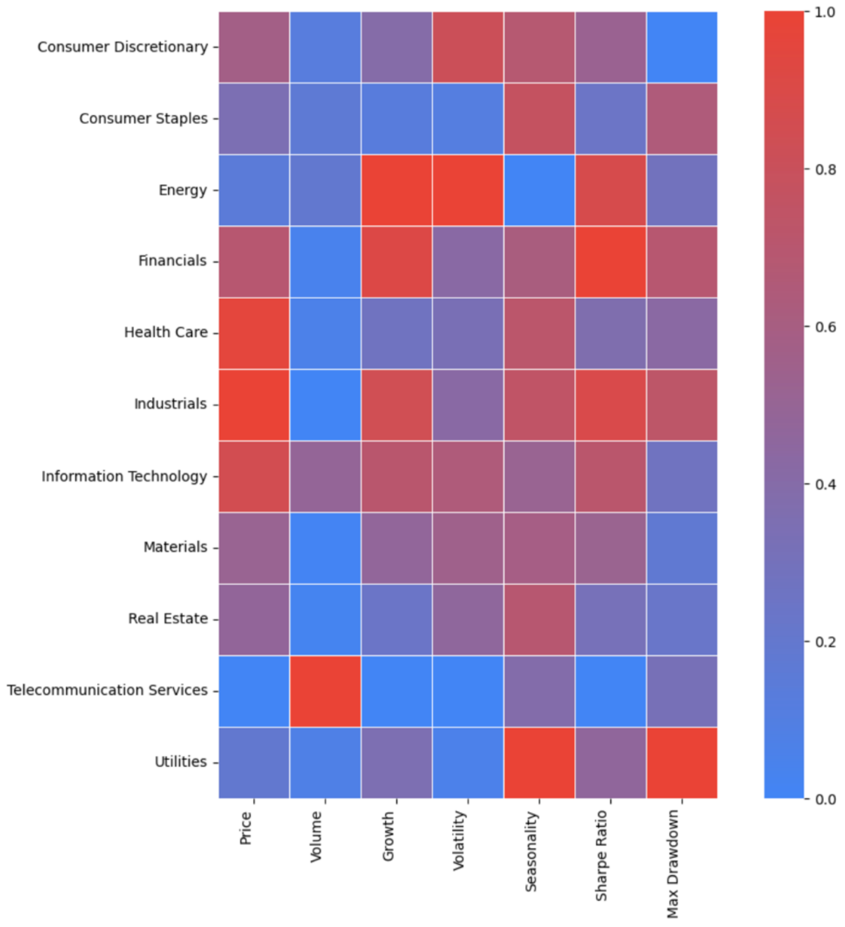

Figure 4.5: Heatmap between sectors and metrics

Sector-level aggregation reveals systematic patterns across different market segments, as illustrated in Figure 4.5. The Energy sector demonstrates strong growth and volatility values, reflecting the cyclical and commodity-driven nature of oil and gas markets. Information Technology, Industrials and Financials exhibit high growth with good risk characteristics, while having moderate-high volatility patterns, which reflects their strong return potential. Consumer Staples and Utilities, conversely, show lower growth and volatility values but demonstrate higher stability in ratio and max drawdown metrics, highlighting their defensive characteristics and role as portfolio stabilizers.

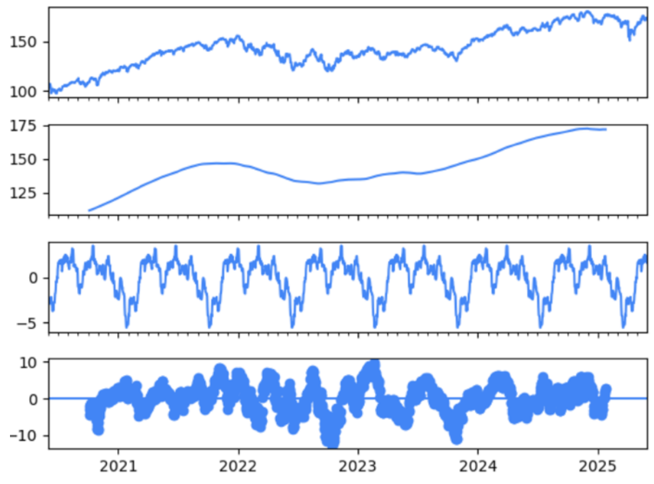

Figure 4.6: Seasonal decomposition for the average close price of all stocks

The seasonal decomposition in Figure 4.6 confirms non-stationarity through its trend component, which shows non-linear growth patterns rather than a constant mean. The varying magnitude of residuals across different time periods provides additional evidence of changing statistical properties over time, specifically heteroskedasticity, a violation of the constant variance assumption fundamental to many traditional forecasting approaches. This high noise-to-signal ratio represents a big challenge for predictive modelling, as genuine market signals must be extracted from substantial random fluctuations. Despite the dominant trend and noise components, the seasonal decomposition reveals consistent cyclical patterns in the data. However, the relatively small magnitude of seasonal effects compared to the residual component highlights the challenge of signal identification amid substantial noise.

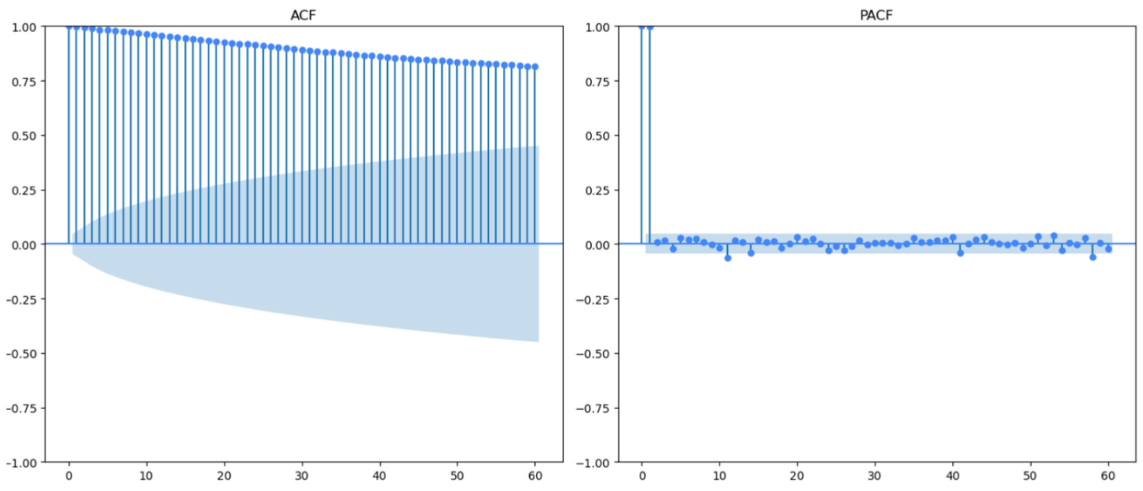

Figure 4.7: ACF and PACF with 60 lags for the average close price of all stocks

In Figure 4.7, the ACF and PACF analysis provides empirical evidence of the strong temporal dependencies inherent in stock market data. The ACF exhibits a slow, gradual decay pattern characteristic of non-stationary time series, with significant correlations extending beyond 60 lags. This confirms that current stock prices are substantially influenced by a long history of past observations, validating the theoretical need for models capable of capturing long-term memory effects. The PACF shows significant correlations at early lags that quickly diminish after the first few periods, suggesting that while direct lag dependencies may be limited, the cumulative effect of historical information remains substantial.

4.2.2 Data Cleaning

Reliable predictive models depend critically on the integrity of their input data. Before modelling can occur, it’s important to ensure the dataset is free from gaps and structural issues that could compromise learning.

The dataset was filtered to retain only the stocks with sufficient historical coverage. First, 50 stocks were removed for having no data. This occurs because the stock has been delisted or is no longer accessible via the API. Then, an additional 52 stocks were dropped for having missing data points. After this filtering, the dataset consisted of 403 stocks with consistent, high-quality historical data and no missing values.

As the U.S. stock market is closed on weekends, data for Saturdays and Sundays was naturally absent. To preserve a continuous daily time series, crucial for NNs that require regular temporal intervals, these weekend gaps were filled using Forward Filling with the market values from the previous Friday, maintaining the last known market value until markets reopened.

After computing technical indicators, an extra two months of early data, originally added to support sliding-window calculations, were trimmed. Once indicators were calculated, this temporary buffer was removed, ensuring the final dataset covered the exact 5-year window from June 1st, 2020, to May 30th, 2025. At the end of the data cleaning process, the resulting dataset comprised 735,47 observations and 26 columns, 2 of them categorical (Symbol and Date).

Outliers were not addressed at this stage of preprocessing. Since the modelling phase involves testing different outlier-handling techniques, preserving the raw data ensures an unbiased comparison of their effectiveness.

4.2.3 Data Transformation

Transforming raw data into a format is a critical step that directly influences model convergence and predictive accuracy. In this stage, the only operation performed was the normalization of features.

A Min-Max normalization technique was chosen to scale all numerical variables, ensuring that all features contributed proportionally to the learning algorithms while preserving their relationships. This choice was influenced by the reviewed articles in the state-of-the-art section, which predominantly applied Min-Max scaling. Specifically, out of the 16 reviewed studies, 7 articles either explicitly applied a Min-Max normalization formula or used equivalent expressions that map data between minimum and maximum values.

In the MinMax normalization all input variables were scaled to the [0, 1] range. Scaling inputs to the [0, 1] interval is preferable because ReLU outputs are non-negative and unbounded above, and feeding normalized, non-negative data helps improve convergence and model stability.

Data smoothing techniques, like handling outliers techniques, were not applied at this stage. The impact of such techniques on model performance will be systematically explored during the optimization phase, where different smoothing strategies will be tested and compared to determine which yields the best forecasting results.

4.2.4 Data Preparation for Sequential Neural Networks

Proper dataset partitioning and sequence construction are critical to preventing data leakage and ensuring models generalize to future market conditions rather than simply memorizing historical patterns.

Before modelling, the dataset was split into training and testing subsets. This split is essential to ensure the model is trained on past data and evaluated on future, unseen observations, preserving the temporal integrity required for time series forecasting. An

80-20 split ratio was adopted, meaning that 80% of the data was allocated for training and 20% for testing. This ratio provides sufficient historical data for the model to learn complex patterns while maintaining a substantial testing period to reliably evaluate performance on unseen data. This decision was guided by the articles in the state-of-the-art review, where the 80-20 partition was the most frequently employed strategy. Within this window, the training data comprises observations from June 1st, 2020, to May 30th, 2024 (4 years), while the testing data covers the period from June 1st, 2024, to May 30th, 2025 (1 year).

A sliding time window approach was implemented to transform the time series data into a supervised learning format. A window size of 30 days was selected based on the analysis of the state-of-the-art literature, where the average window size across reviewed articles was 27.5 days. This value was rounded to 30 days to represent approximately one month, providing sufficient historical context for the model to learn meaningful patterns while maintaining computational efficiency.

The sliding window sequences were reshaped to meet the input structure required by NNs, ensuring that the temporal order of the sequences is preserved. These sequences were converted into 3-dimensional arrays, where the first dimension represents the number of samples, the second represents the window size, and the third represents the number of variables per time step.

4.3 First Modelling Phase: Variables, Models and Architectures

The first modelling phase represents the initial step of the NN modelling optimization approach. It encompasses four critical components: strategic selection of a representative stock that serves as the primary testing ground for model optimization, identification of appropriate NN models and architectures based on their proven effectiveness in financial time series prediction, systematic determination of hyperparameter configurations based on state-of-the-art practices, and comprehensive evaluation methodology for comparing model performance across different variable groups and architectural configurations.

4.3.1 Single Stock Selection

As mentioned earlier, an initial model was developed and optimized in a single stock. This targeted approach enables the careful evaluation and comparation of variables, preprocessing methods, NN models, architectures, parameters and regularization techniques. Therefore, choosing the stock wisely is crucial, as it will serve as the foundation for optimizing the modelling configurations before generalizing to a broader set of stocks.

From a behavioural perspective, the selected stock should exhibit moderate volatility with clear trend patterns and cyclical behaviour, while replicating the trends and movement of the broader set of S&P 500 stocks. Market representation is equally important, requiring a large, highly liquid company that represents a major sector but isn't dominated by sector-specific price movements.

Based on these criteria, the AAPL stock emerged as an optimal choice for several technical and empirical reasons. AAPL demonstrates excellent data quality through its position as one of the highest average daily trading volume stocks in the S&P 500, it places 4th out of the 403 stocks in the dataset in trading volume (99.3 percentile), demonstrating exceptional liquidity and data reliability essential for robust model development.

As one of the largest companies globally by market capitalization, AAPL's price movements exhibit stability and is less prone to abrupt distortions caused by low trading activity, while still maintaining sufficient volatility to provide meaningful learning signals for model training. The ranking analysis further validates this balance, positioning AAPL at the 197th rank out of 403 stocks for volatility (51.4 percentile), placing it in the middle range of volatility metrics. This moderate volatility profile is advantageous for model optimization, as it avoids the architectural biases that would emerge from extreme volatility conditions. Specifically, optimizing on a low-volatility stock would likely result in overly simplistic model architectures that fail to capture complex non-linear patterns when applied to more volatile stocks, while optimization on high-volatility stocks would tend toward unnecessarily complex architectures with excessive regularization that could underperform on stable stocks by over-smoothing meaningful signals.

AAPL's suitability is further reinforced by its prevalence in the existing academic literature, appearing in 25% of the 16 articles reviewed in this project's bibliographic analysis, the highest frequency observed. This strong presence highlights its role as a benchmark stock in this type of studies, further justifying its selection for this work.

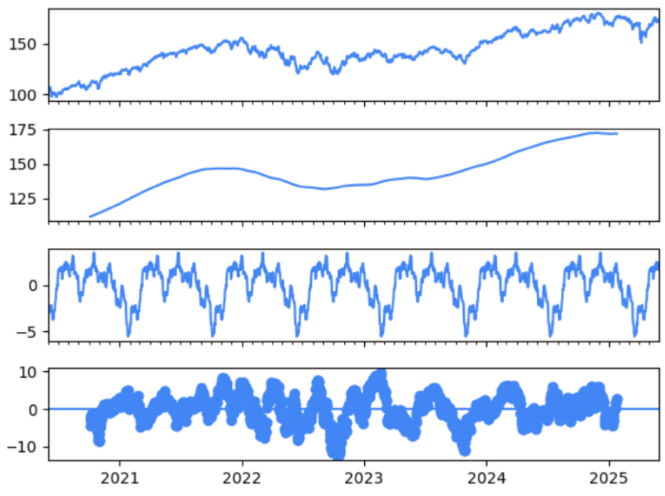

Figure 4.8: Seasonal decomposition for the close price of AAPL

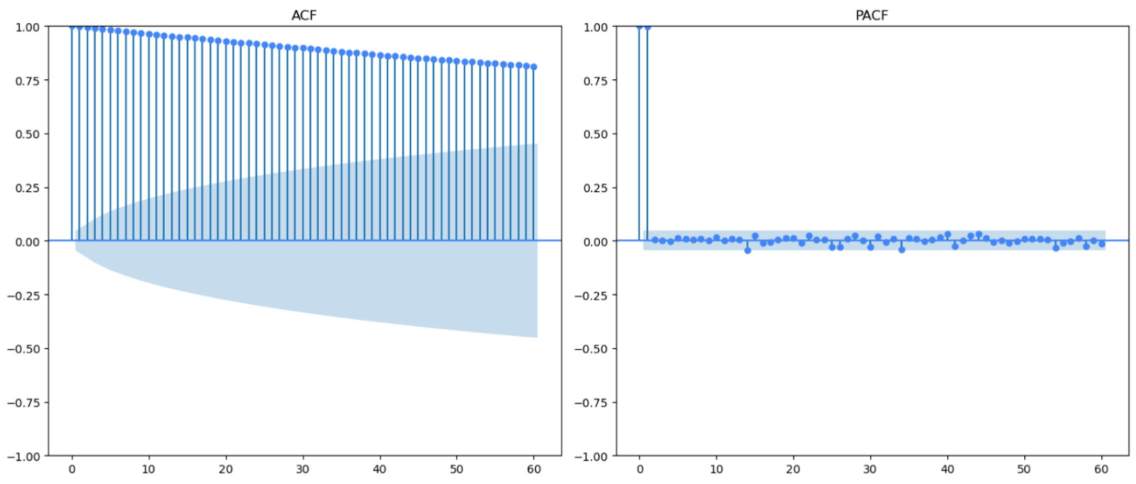

Figure 4.9: ACF and PACF with 60 lags for the close price of AAPL

As observed in Figures 4.8 and 4.9, AAPL demonstrates characteristics that closely mirror broader S&P 500 market behaviour, validating its selection as a representative modelling foundation, ensuring that models developed on AAPL will address the same fundamental time series characteristics present across the broader market and transfer effectively to other S&P 500 constituents.

4.3.2 Models Selection

Following the single stock selection, the next step involved selecting the appropriate NN models. Two types of NN models emerged as obvious choices: LSTM and GRU. Both are widely recognized for their effectiveness in time-series forecasting tasks due to their ability to capture long-term dependencies and sequential patterns in data.

Both were implemented due to being the most efficient and popular NN models in the domain of stock price prediction. LSTMs were the most used models in the state-of-the-art literature, appearing in almost every single article. GRUs, while less prevalent, were employed in almost half of the studies and often outperformed LSTMs in terms of predictive accuracy and computational efficiency, although LSTMs demonstrated competitive results.

4.3.3 HyperParameters Selection

For this initial modelling phase, several hyperparameters were selected before training and evaluation. Given the exploratory nature of this stage, the chosen hyperparameters were based on the most used configurations found in the literature. Instead of extensive tuning, this stage adopted standard and widely accepted values aligned with the current state-of-the-art practices for time-series forecasting tasks. This approach ensured a fair and efficient comparison between model types while deferring more advanced and systematic hyperparameter fine-tuning to the subsequent model optimization phase.

In this stage, the only hyperparameters optimized were the number of layers and neurons per layer. Since the models will be tested across different datasets all with different sizes and complexities, it is essential to first determine the architectural configuration that best captures patterns within each data context. After preliminary experiments and using the state-of-the-art as a guide, the following ranges were chosen for the systematic evaluation:

- Number of Layers: 1, 2 and 3.
- Number of Neurons per Layer: 8, 16, 32, 64 and 128.

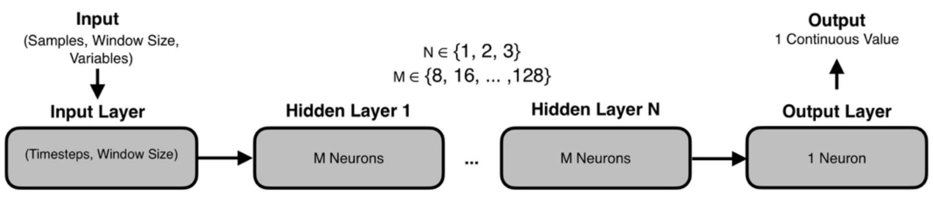

Figure 4.10: Architectures summary

Like Figure 4.10 shows, every architecture has an input layer that receives sequential time series data with shape (samples, window size, variables), followed by N hidden layers, each with M neurons, and concludes with one output layer containing a single dense neuron that produces a continuous predicted value. Deeper models were not explored as they are very rare in the state-of-the-art literature, and preliminary experiments confirmed that increasing layer depth deteriorated performance, making further exploration impractical given the project's time constraints and energy efficiency considerations.

The selection of the remaining hyperparameters was also guided by the practices adopted in the state-of-the-art studies as well as preliminary testing:

- Number of Epochs: A number of 50 epochs was chosen based on preliminary experiments, which suggested that this number is sufficient for the model to converge.
- Batch Size: A batch size of 1 was used, representing a standard default that ensures stable updates during training.
- Learning Rate: The learning rate was set to 0.001, a commonly accepted default that balances convergence speed with training stability.
- Optimizer: Adam was selected due to its widespread use in current literature and its proven effectiveness.
- Activation Function: ReLU was employed, as it is the most widely used activation in the analysed studies and facilitates efficient gradient propagation.
- Loss Function: MSE was adopted, consistent with standard practices for regression tasks and commonly used in similar studies.

4.3.4 Final Variables, Model and Architecture Selection

After defining the variable groups, selecting the stock, choosing the models, and setting the hyperparameters, all 15 combinations of model architectures were tested for each model and variable group. Each configuration was trained and evaluated five times with different random seeds to obtain robust performance estimates. The choice of five runs was made due to time constraints, as this optimization phase alone is expected to run for several days.

The evaluation process incorporated multiple statistical measures: mean for primary predictive accuracy, standard deviation to assess consistency across runs, coefficient of variation to measure relative variability, and a composite scoring system for holistic model ranking. The composite scoring system weights the three performance aspects: 70%, 20% and 10%, respectively, where the lowest composite score indicates the best overall performance as it represents superior predictive accuracy combined with minimal variability across runs.

For each variable group, models were ranked using this composite score, and the best-performing architecture was identified. This comprehensive evaluation approach reduces the impact of randomness and overfitting while ensuring that the selected model demonstrates both strong predictive performance and stable behaviour across different training conditions. Following this systematic evaluation across all dataset-model-architecture combinations, the best overall performing group was selected as the final configuration for the next modelling phase.

4.4 Second Modelling Phase: PreProcessing, HyperParameters and Regularization

The second modelling phase represents a comprehensive optimization process aimed at refining the ideal architecture identified in the previous stage. This phase systematically addresses three components: preprocessing methods examining outlier handling, smoothing techniques, normalization formulas, and sliding window sizes, hyperparameter tuning exploring batch sizes, optimizer-learning rate combinations, activation functions, and loss functions, and regularization techniques implemented to prevent overfitting and improve generalization capability.

4.4.1 PreProcessing Methods

After determining the optimal combination of variables, model and architecture in the previous stage, the refinement pipeline continued with preprocessing methods optimization. The impact of several preprocessing strategies was evaluated, including different: outlier handling and smoothing techniques, normalization formulas, and sliding window sizes.

Each technique was assessed independently, with all other preprocessing steps and model hyperparameters held constant according to the default configuration established earlier. This approach ensured that the specific contribution of each preprocessing method can be isolated and evaluated accurately. For each method under consideration, the model was trained and evaluated five times using the same robust evaluation approach as previously described, with performance metrics including mean, standard deviation, coefficient of variation, and composite scores recorded.

Although preprocessing choices are underexplored or omitted in the literature analysed, the methods tested represent widely adopted and empirically effective practices in time-series forecasting and NN applications, while the parameters used were defined accordingly to preliminary testing. The specific preprocessing methods evaluated include the following:

- Outlier Handling Techniques: Winsorization ([5%, 95%]) and Clipping ([1%, 99%). These methods were compared with the raw data (default).
- Smoothing Techniques: SMA (window size = 7), RM (window size = 7), GF (sigma = 1.5) and SGF (window size = 7 and poly order = 2). These methods were compared with the raw data (default).
- Normalization Formulas: MinMax ([0,1]; default), Z-Score and Log Normalization ([0,1]).
- Sliding Window Sizes: 1, 2, 3, 4, 5, 6, 7, 15, 30 (default), 60 and 90.

4.4.2 HyperParameters

Following the optimization of preprocessing methods, hyperparameter tuning is conducted to further enhance model performance. This phase systematically explored key training parameters that significantly influence the learning dynamics and final performance of the NN architecture. This optimization phase followed the same rigorous methodology established in the preprocessing evaluation phase.

The selected parameter ranges represent commonly used values in NN literature and practical applications, providing comprehensive coverage of the hyperparameter space. The specific hyperparameters evaluated include the following:

- Batch Sizes: 1 (default), 2, 4, 8, 16, 32, 64 and 128.
- Optimizers and Learning Rates Combination:
    - Optimizers: SGD, AdaGrad, AdaDelta, RMSprop and Adam (default).
    - Learning Rates: 0.0001, 0.001 (default), 0.005, 0.01 and 0.05.

  - Since the effectiveness of a given optimizer can vary greatly depending on the step size, this approach ensures the hyperparameter search accounts for optimizer-learning rate interactions rather than assuming their impacts are independent.
- Activation Functions: TanH, Sigmoid, ReLU (default) and Leaky ReLU.
- Loss Functions: MSE (default), RMSE, MAE and MAPE.

4.4.3 Regularization Techniques

Following the optimization of preprocessing methods and hyperparameters, the final stage of model refinement focused on implementing overfitting regularization techniques. This phase is important for enhancing the model's generalization capability and preventing overfitting, which is particularly important in financial time-series forecasting where models may memorize historical patterns that do not generalize to future market conditions. Each regularization technique was evaluated independently using the same 5-run methodology, with all other parameters fixed at their optimal values to isolate the specific impact of each method compared to a baseline model without regularization.

The regularization techniques selected represent well-established methods in NNs literature that have proven effective in preventing overfitting across various domains. While the parameters used were determined through preliminary testing. The specific regularization methods evaluated include the following:

- Dropout (dropout rate = 0.1; after the hidden layers).
- Early Stopping (patience parameter = 10 epochs; only case in which a higher number of epochs was required).
- L2 Regularization (lambda parameter = 0.001; in the hidden layer's kernel weights).
- Batch Normalization (after the hidden layers).
- Recurrent Dropout (dropout rate = 0.1; in the recurrent connections in the hidden layers).
- Gradient Clipping (clipping norm = 1.0).
- Data Augmentation (Gaussian noise = 0.01).

4.5 Third Modelling Phase: Evaluation

The third modelling phase focuses on comprehensive evaluation of the optimized model architecture across the entire S&P 500 universe. This phase trains and tests the finalized model architecture individually on each of the 403 stocks for large-scale performance assessment, encompassing: quantitative analysis through regression and classification metrics, and qualitative analysis through visualization approaches examining prediction accuracy, sector-specific performance patterns, and risk-return characteristics.

4.5.1 Metrics

Beyond RMSE, which served as the evaluation function during the optimization phases, a comprehensive suite of metrics was employed to provide a holistic assessment of model performance. The evaluation framework encompassed both regression and classification measures, recognizing that stock price prediction involves not only accurate magnitude forecasting but also directional movement prediction. The regression metrics are: RMSE, MSE, MAE, MAPE and R^2. The classification metrics are: Accuracy, Precision, Recall and F1.

4.5.2 Analysis

The analysis employed multiple visualization and statistical approaches to comprehensively evaluate model performance across the S&P 500 universe. Individual stock prediction visualization used time series plots displaying actual prices alongside training predictions, test predictions, and future forecasts to assess model tracking accuracy and detect overfitting by comparing fit quality between training and test phases.

Statistical analysis included descriptive statistics for all performance metrics to identify patterns and outliers. Correlation analysis between stock characteristics and model performance was visualized to reveal relationships between fundamental stock properties and prediction accuracy.

Performance segmentation classified stocks into "Good", "Average", and "Bad" categories using a composite scoring system that weights five key metrics: MAPE, R^2, Accuracy, Precision, and F1 Score. Each metric contributes points based on performance thresholds: Accuracy scores range from +4 points (≥65%) to -3 points (<50%), MAPE from +3 points (≤5% error) to -2 points (>50% error), R^2 from +3 points (≥0.8) to -2 points (<0), F1 Score from 2 points (≥75%) to -1 point (<55%) and Precision from +2 points (≥80%) down to -1 point (<60%). The final classification depends on the total score: "Good" performance requires ≥8 points, "Average" spans 3-7 points, and "Bad" applies to scores below 3 points. This categorization enables understanding of how many stocks achieved strong predictive performance and identification of their common characteristics for broader market insights.

Drawing inspiration from Chang *et al.* [38], risk-return scatter plot analysis positioned all stocks according to predicted daily risk and return characteristics. Daily returns were computed as the percentage change between consecutive predicted prices, while daily risk was measured as the standard deviation of these predicted returns. This visualization provides crucial insights into the relationship between potential predicted gains and financial risk profiles, helping to assess which stocks and types of stocks presented better return metrics adjusted to their risk. This analysis enables portfolio-level insights and facilitates comparison of model performance across different risk profiles and market sectors.

5. Results and Discussion

This chapter presents the empirical findings from the systematic investigation. The analysis examines variable selection strategies and model architecture performance, complemented by preprocessing optimization, hyperparameter tunning and regularization evaluation. The chapter concludes with large-scale validation across S&P 500 stocks, examining performance patterns and practical deployment considerations.

5.1 First Modelling Phase: Variables, Models and Architectures

This section presents the comprehensive evaluation results from the first modelling phase, where LSTM and GRU models with 15 different architectures were systematically tested across three distinct variable groups using AAPL data. Each configuration is evaluated five times using mean, standard deviation, coefficient of variation, and composite scoring methodologies.

5.1.1 Results and General Discussion

Layers x Neurons	LSTM				GRU			
	Mean	SD	CV %	Score	Mean	SD	CV %	Score
1x8	4.41	0.52	11.9%	0.01	6.33	4.69	74.2%	0.18
1x16	18.49	28.70	155.2%	1.00	20.99	28.89	137.6%	1.00
1x32	4.26	0.50	11.8%	0.002	4.25	0.78	18.3%	0.02
1x64	4.22	0.65	15.5%	0.004	3.95	0.23	5.8%	0.00

1x128	4.46	0.69	15.6%	0.01	4.00	0.37	9.3%	0.005
2x8	6.63	1.81	27.3%	0.13	7.64	3.38	44.3%	0.20
2x16	5.67	1.28	22.5%	0.08	4.94	1.09	22.1%	0.05
2x32	5.64	1.73	30.8%	0.09	4.21	0.57	13.6%	0.01
2x64	6.01	1.41	23.5%	0.10	4.38	0.72	16.5%	0.02
2x128	5.43	1.10	20.4%	0.07	4.58	0.84	18.4%	0.04
3x8	6.73	1.30	19.3%	0.13	20.91	26.83	128.3%	0.97
3x16	7.80	1.11	14.3%	0.18	5.64	1.60	28.4%	0.09
3x32	7.09	0.94	13.3%	0.14	4.71	0.66	14.1%	0.04
3x64	6.96	1.52	21.8%	0.14	4.89	0.88	18.1%	0.05
3x128	8.01	1.17	14.7%	0.19	5.00	0.85	17.2%	0.05
Mean	**6.79**	**2.96**	**27.8%**	**0.15**	**7.09**	**4.82**	**37.7%**	**0.18**

Table 5.1: LSTM/GRU architectures using univariate dataset RMSE performance across five runs

As we can see in Table 5.1, the univariate dataset, utilizing only closing price as input, demonstrates good and stable performance across both model types, with LSTM on average outperforming GRU, mainly since two of the GRU's architectures became unstable, against the one of the LSTM. The addition of layers, mainly on the LSTM's architectures, generally worsens results because deeper networks are prone to overfitting on this relatively simple, single-feature dataset, and the vanishing gradient problem becomes more pronounced. The best performing configuration is the GRU 1x64 architecture with a mean RMSE of 3.95 and remarkably low standard deviation of 0.23, demonstrating that optimal capacity matching between model complexity and data complexity is essential for robust performance.

Layers x Neurons	**LSTM**				**GRU**			
	Mean	**SD**	**CV %**	**Score**	**Mean**	**SD**	**CV %**	**Score**
1x8	5.26	1.15	21.8%	0.04	4.77	1.28	27.0%	0.37
1x16	10.64	11.71	110.1%	0.78	6.19	1.55	25.1%	0.73
1x32	5.02	1.14	22.8%	0.02	4.37	0.66	15.1%	0.17
1x64	4.99	1.17	23.5%	0.02	4.42	0.36	8.2%	0.13
1x128	5.31	0.50	9.5%	0.02	4.07	0.51	12.7%	0.08

2x8	12.14	5.92	48.8%	0.74	5.47	1.15	21.1%	0.51
2x16	9.53	4.20	44.1%	0.49	4.95	0.41	8.4%	0.27
2x32	7.10	0.69	9.8%	0.18	4.58	0.65	14.3%	0.22
2x64	7.95	3.21	40.4%	0.33	4.49	0.48	10.8%	0.17
2x128	7.85	1.12	14.3%	0.26	5.70	1.66	29.2%	0.64
3x8	9.08	2.45	27.0%	0.40	4.04	0.08	2.2%	0.00
3x16	13.16	10.01	76.1%	0.93	4.39	1.07	24.6%	0.25
3x32	11.18	1.25	11.2%	0.54	4.71	0.67	14.2%	0.25
3x64	10.22	3.48	34.1%	0.52	4.35	0.42	9.7%	0.13
3x128	10.05	1.86	18.5%	0.46	6.96	2.17	31.3%	1.00
Mean	**8.63**	**3.32**	**34.1%**	**0.38**	**4.90**	**0.88**	**16.9%**	**0.33**

Table 5.2: LSTM/GRU architectures using market dataset RMSE performance across five runs

Table 5.2 shows that the market dataset, incorporating Open, High, Low, Close, and Volume variables, shows a reversal in model performance with GRU significantly outperforming LSTM, since this time there aren't any unstable predictions. Also, GRU's architectures performance is a lot more consistent with the rise in their complexity, where the LSTM's performance dipped. The optimal configuration is the deeper GRU 3x8 architecture, achieving a mean RMSE of 4.04 with exceptionally low standard deviation of 0.08.

Layers x Neurons	**LSTM**				**GRU**			
	Mean	**SD**	**CV %**	**Score**	**Mean**	**SD**	**CV %**	**Score**
1x8	8.94	0.77	8.7%	0.00	8.62	3.51	40.8%	0.17
1x16	16.73	12.09	72.3%	0.04	11.78	5.77	49.0%	0.49
1x32	23.23	26.81	115.4%	0.08	12.45	2.70	21.7%	0.32
1x64	24.58	26.55	108.0%	0.08	9.88	2.77	28.1%	0.18
1x128	19.35	12.68	65.5%	0.04	15.26	1.74	11.4%	0.42
2x8	31.96	14.82	46.4%	0.05	10.31	3.43	33.3%	0.25
2x16	18.90	6.74	35.7%	0.02	14.75	3.20	21.7%	0.48
2x32	75.80	90.99	120.0%	0.16	20.00	4.62	23.1%	0.88

2x64	192.01	336.88	175.4%	0.37	18.37	4.32	23.5%	0.76
2x128	22.57	5.27	23.4%	0.02	17.83	3.27	18.3%	0.67
3x8	62.40	36.39	58.3%	0.09	10.52	3.08	29.3%	0.24
3x16	91.78	94.88	103.4%	0.17	12.27	3.70	30.2%	0.38
3x32	31.26	8.14	26.0%	0.03	16.62	1.51	9.1%	0.49
3x64	595.73	1091.47	183.2%	1.00	17.72	3.56	20.1%	0.68
3x128	27.11	5.63	20.8%	0.02	12.39	2.04	16.5%	0.27
Mean	**82.82**	**118.01**	**77.5%**	**0.14**	**13.92**	**3.28**	**25.0%**	**0.44**

Table 5.3: LSTM/GRU architectures using technical dataset RMSE performance across five runs

The technical indicators dataset, featuring 19 derived metrics, presents the most challenging modelling environment with significantly degraded performance, as seen in Table 5.3. LSTM architectures suffer more severely, while GRU's simpler update mechanism proves more robust to feature noise, achieving roughly one-third the variance of LSTM results. Despite LSTM showing the worst overall results, the best performing model is surprisingly the LSTM 1x8 architecture with a mean RMSE of 8.94 and coefficient of variation of 8.7%, suggesting that when dealing with complex feature spaces, extremely simple architectures may be the only viable approach to avoid catastrophic overfitting, though the performance is still substantially worse than both previous datasets.

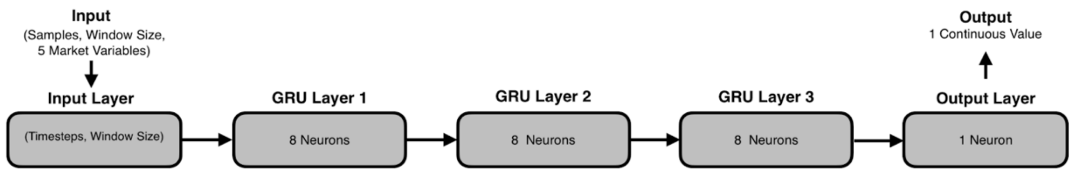

Figure 5.1: Best dataset-model-architecture combination

Based on the comprehensive evaluation across all criteria, using the same logic of the composite scoring system, the market GRU 3x8 architecture emerges as the optimal configuration for the subsequent modelling phases, illustrated on Figure 5.1. This selection is justified by its balanced performance, achieving competitive RMSE of 4.04 while maintaining exceptional stability with the lowest coefficient of variation among all competitive models.

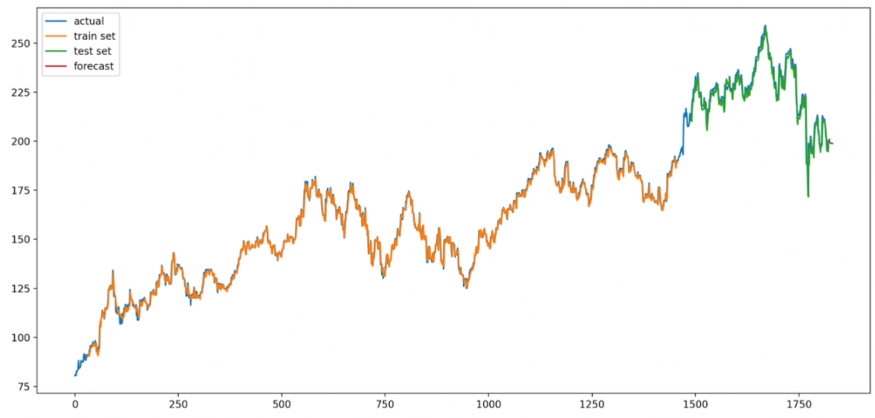

Figure 5.2: Performance of market GRU 3x8 architecture for AAPL

Based on Figure 5.2, showing the performance of the market GRU 3x8 architecture for the AAPL stock, the model demonstrates excellent learning capability and reasonable predictive accuracy across different phases of the time series. The training set (orange line) closely follows the actual values (blue line) throughout the historical period, indicating that the model successfully captures the underlying patterns and trends in the market data without significant overfitting. The test set predictions (green line) maintain good alignment with actual values during the evaluation period, though with some increased variance particularly during high volatility periods where the model shows some difficulty in capturing rougher price movements.

5.1.2 Variables

The systematic evaluation of different variable groups reveals that model performance varies across input configurations, with results challenging conventional assumptions about the superiority of multivariate approaches in financial forecasting. Univariate models achieve competitive performance across all architectural configurations, in relation to market models, and completely outperform technical models, supporting studies suggesting that univariate models can be more accurate and faster than multivariate ones for stock market prediction in certain contexts [29].

The failure of market variables to provide substantial improvement over univariate models stems from fundamental data characteristics, as Open, High, Low, and Close prices exhibit near-perfect correlations, creating redundant information that adds model complexity without proportional predictive value, with only Volume data capturing genuinely different dynamics. Technical indicators produced the most disappointing results with substantially higher error rates, directly contradicting literature findings about their forecasting benefits [37], though this poor performance likely stems from methodological factors including redundancy among chosen indicators, suggesting that feature selection and dimensionality reduction techniques could address these issues, as demonstrated effective in state-of-the-art analyses [3], [31], rather than challenging the theoretical value of these indicators for capturing market momentum and trends.

5.1.3 Models

The results show that GRU models generally achieve superior performance, with lower mean RMSE values in most architectural configurations, suggesting a general preference for GRU architectures in stock price prediction tasks. But the GRU model doesn’t show universal superiority across all contexts, consistent with literature indicating that optimal performance depends on specific context and data characteristics [36].

LSTM outperforms GRU on the univariate dataset, likely because with only one input, the simplified feature space allows LSTM's sophisticated memory management to effectively capture subtle long-term price patterns without the overfitting risks that emerge in more complex datasets [29]. In this low-dimensional context, LSTM's additional complexity provides genuine value rather than introducing unnecessary parameters that could lead to gradient explosions. Conversely, GRU dominates on the market dataset and shows superior stability on the technical indicators dataset, demonstrating that GRU's performance advantage becomes more pronounced with increasing feature complexity. This aligns with research indicating that GRU's simplified architecture can perform similarly to LSTM with fewer parameters and greater computational efficiency [1], making it particularly well-suited for high-dimensional datasets where the reduced parameter count prevents overfitting.

5.1.4 Architectures

The relationship between architectural complexity and performance reveals that shallow architectures consistently outperform deeper networks across all datasets, aligning with literature findings that increasing depth introduces overfitting risks and computational complexity that may outweigh benefits [24]. This pattern challenges assumptions about deeper networks and reflects financial data's unique characteristics, where high noise-to-signal ratios benefit from simpler, more robust models rather than complex feature hierarchies that may amplify noise or learn false patterns.

A consistent overfitting pattern emerges where models achieve lower training errors on larger datasets but higher testing errors, as increased data complexity enables models to learn noise and excessive detail, resulting in poor generalization that is exemplified most clearly in the technical indicators dataset. The challenges are compounded by gradient explosion phenomena encountered particularly in the technical indicators dataset, where despite using LSTM and GRU architectures designed to overcome such problems [29], the high dimensionality and temporal volatility of financial data can overwhelm protective mechanisms, with deeper networks showing higher susceptibility when combined with higher neuron counts.

Based on the comprehensive evaluation results, the study demonstrates significant energy saving benefits through simpler architectural approaches. The analysis reveals that shallow architectures consistently outperform deeper networks, while eliminating computational overhead associated with unnecessary complexity. This aligns with energy-efficient modelling principles, where reduced parameter counts translate to lower computational requirements, as dimensionality reduction techniques improve learning algorithm efficiency while reducing computational and storage demands, increasing sustainability through lower energy costs [22]. The selection of GRU architectures reinforces this energy-conscious approach, requiring fewer parameters than LSTM while maintaining superior performance and enabling faster training cycles [2].

5.2 Second Modelling Phase: PreProcessing, HyperParameters and Regularization

This section presents the comprehensive optimization results from the second modelling phase, where systematic refinement of the GRU 3x8 architecture with market variables was conducted through three critical optimization stages. The analysis encompasses 18 preprocessing configurations, 37 hyperparameter combinations, and 7 regularization techniques, with each configuration evaluated five times using the same robust methodology and scoring system established in the previous phase.

5.2.1 Preprocessing Methods

The preprocessing optimization phase systematically evaluated four distinct categories of data preparation techniques: outlier handling methods, smoothing algorithms, normalization formulas, and sliding window configurations, as seen in Table 5.4.

Method		Mean	SD	CV %	Score
Outlier	Nothing (Default)	4.04	0.08	2.2%	0.00
	Winsorization	5.18	1.60	30.9%	1.00
	Clipping	4.58	0.77	17.0%	0.47
Smoothing	Nothing (Default)	4.04	0.08	2.2%	0.00
	SMA	4.53	0.33	7.5%	0.11
	RM	4.38	0.61	14.0%	0.13
	GF	8.16	2.93	35.9%	1.00
	SGF	5.20	1.03	19.8%	0.31
Normalization	MinMax (Default)	4.04	0.08	2.2%	0.00
	Z-Score	4.67	1.04	22.2%	0.52
	Log	5.45	1.52	27.9%	1.00
Sliding Window	1	4.68	1.48	31.6%	0.48
	2	6.18	1.86	30.2%	0.99
	3	3.97	0.30	7.8%	0.04
	4	4.55	0.72	16.0%	0.30
	5	5.08	1.22	24.1%	0.55
	6	4.34	0.88	20.3%	0.26
	7	4.78	0.72	15.2%	0.37
	15	4.12	0.85	20.6%	0.19
	30 (Default)	4.04	0.08	2.2%	0.02
	45	5.60	1.26	22.5%	0.71

	60	4.42	0.69	15.7%	0.25

Table 5.4: PreProcessing methods RMSE performance across five runs

The systematic evaluation shows that neither Winsorization nor Clipping improves upon raw data processing. Winsorization performs worst with mean RMSE of 5.18, while Clipping shows moderate degradation at 4.58 RMSE. The poor performance likely occurs because extreme price movements often represent genuine market information rather than noise, and removing these values eliminates valuable learning signals capturing volatility and regime changes.

Most smoothing methods fail to improve the raw data baseline. SMA achieves modest results with RMSE of 4.53, while the RM shows slight enhancement at 4.38 RMSE. However, more sophisticated techniques like GF dramatically degrade performance with a RMSE of 8.16, and SGF provides marginal benefit at 5.20 RMSE. The failure of advanced smoothing techniques suggests financial data's non-linear, non-stationary characteristics resist traditional signal processing, and inherent noise may contain important information that smoothing removes.

MinMax normalization with [0,1] interval emerges as the optimal choice. Z-Score normalization shows moderate degradation with 4.67 RMSE, while Log normalization produces the worst results with 5.45 RMSE. MinMax's superior performance stems from its compatibility with ReLU activation functions, which operate optimally with non-negative inputs.

The sliding window size analysis reveals complex relationships between temporal memory and predictive performance, with the 30-day window emerging as optimal despite the 3-day window achieving lowest RMSE. ACF and PACF analysis validates this choice, showing slow-decaying temporal dependencies extending beyond short-term patterns, supporting sufficient historical context while avoiding overfitting risks from longer sequences.

5.2.2 HyperParameters

As shown in Table 5.5, the hyperparameter optimization phase systematically evaluated different configurations across four critical training parameters: batch sizes, optimizers and learning rates combinations, activation and loss functions.

Parameter			Mean	SD	CV %	Score
Batch	1 (Default)		4.04	0.08	2.2%	0.00
	2		5.30	0.80	15.1%	0.21
	4		5.56	1.32	23.7%	0.29
	8		6.00	1.26	21.0%	0.33
	16		4.96	0.79	16.0%	0.17
	32		6.47	2.44	37.7%	0.49
	64		7.28	1.43	19.8%	0.48
	128		10.22	4.01	39.2%	1.00
Optimizer x Learning Rate	SGD	0.0001	29.30	16.45	56.1%	0.31
		0.001	13.09	8.16	62.4%	0.19
		0.005	12.52	7.91	63.2%	0.18
		0.01	11.84	7.45	63.0%	0.18
		0.05	9.01	5.51	61.1%	0.15
	AdaGrad	0.0001	74.82	39.01	52.1%	0.66
		0.001	11.30	2.33	20.5%	0.08
		0.005	8.23	2.44	29.7%	0.08
		0.01	7.50	2.49	33.3%	0.03
		0.05	7.31	4.02	55.0%	0.12
	AdaDelta	0.0001	131.93	12.90	9.8%	0.77
		0.001	45.23	24.35	53.8%	0.43
		0.005	13.48	4.22	31.3%	0.12
		0.01	11.30	2.26	20.1%	0.08
		0.05	6.05	1.09	18.0%	0.04
	RMSprop	0.0001	4.82	0.39	8.2%	0.01
		0.001	4.47	0.29	6.6%	0.01
		0.005	5.18	1.37	26.5%	0.05
		0.01	5.66	0.92	16.4%	0.03
		0.05	65.35	21.29	32.6%	0.49

	Adam (Default)	0.0001	5.48	1.49	27.3%	0.05
		0.001 (Default)	4.04	0.08	2.2%	0.00
		0.005	5.83	2.83	48.5%	0.10
		0.01	5.64	1.79	31.8%	0.06
		0.05	56.85	27.95	49.2%	0.50
Activation	Tanh		14.64	2.95	20.2%	0.53
	Sigmoid		20.33	5.15	25.3%	0.82
	ReLU (Default)		4.04	0.08	2.2%	0.00
	Leaky ReLU		25.66	3.44	13.4%	0.88
Loss	MSE (Default)		4.04	0.08	2.2%	0.12
	RMSE		4.03	0.34	8.7%	0.24
	MAE		4.38	0.65	15.0%	1.00
	MAPE		3.96	0.30	7.6%	0.11

Table 5.5: HyperParameters RMSE performance across five runs

Single-batch processing performs best, with performance degrading as batch sizes increase to the worst results at batch size 128. Small batches work better for this data because individual samples contain unique temporal patterns that benefit from immediate gradient updates rather than averaged updates that smooth out important short-term signals.

Adam optimizer with 0.001 learning rate proves optimal, supporting academic findings. Other optimizers perform significantly worse: SGD underperforms even at higher learning rates, while AdaGrad and AdaDelta show moderate results around 0.01 learning rate. Adam's adaptive learning rate mechanism effectively handles sparse gradients and non-stationary optimization landscapes in financial data.

ReLU emerges as the optimal choice, aligning with the literature. Alternative functions show substantial degradation: TanH (14.64 RMSE), Sigmoid (20.33 RMSE), and Leaky ReLU (25.66 RMSE). ReLU's superiority stems from avoiding vanishing gradient problems, computational efficiency, and compatibility with MinMax [0,1] normalization.

MAPE provides the best performance with 3.96 RMSE, being the only hyperparameter modification to meaningfully improve upon the default baseline. This contradicts conventional MSE use for continuous outcomes, suggesting percentage-based metrics better suit stock prediction where relative accuracy matters more than absolute accuracy due to MAPE's scale-invariant properties, although all metrics provided good similar results.

5.2.3 Regularization Techniques

The overfitting regularization optimization phase systematically evaluated seven different techniques designed to prevent overfitting and improve model generalization, as shown in Table 5.6, using the same evaluation methodology as before.

Technique	Mean	SD	CV %	Score
Nothing (Default)	4.04	0.08	2.2%	0.00
Dropout	7.97	1.32	16.6%	0.03
Early Stopping	4.75	1.26	26.6%	0.03
L2 Regularization	14.00	3.68	26.3%	0.07
Batch Normalization	161.20	160.20	99.4%	1.00
Recurrent Dropout	4.58	0.94	20.6%	0.02
Gradient Clipping	4.78	1.23	25.7%	0.02
Data Augmentation	4.74	1.02	21.5%	0.02

Table 5.6: Regularization techniques RMSE performance across five runs

The default configuration without regularization achieves optimal performance, challenging conventional NN practices. This presumably occurs because the GRU 3x8 architecture already provides appropriate model complexity for the dataset, while financial data's inherent noise and non-stationary properties create natural regularization that makes additional constraints counter-productive. The limited effectiveness suggests these techniques might work better with deeper models or larger datasets, as the current single-stock dataset lacks sufficient complexity to benefit from aggressive regularization. These findings align with literature showing regularization is often neglected in financial forecasting.

5.3 Third Modelling Phase: Evaluation

This section provides comprehensive assessment of the optimal model performance across 403 S&P 500 stocks, requiring approximately 94 hours (almost 4 days) of computation time, which means the model takes about 14 minutes per stock. The evaluation encompasses multiple performance dimensions including: regression accuracy, directional prediction capability, sector-specific patterns, and practical applicability for real-world trading scenarios.

5.3.1 Results and Discussion

The comprehensive evaluation across stocks reveals a big dispersion in model performance with significant implications for trading applications. Regression metrics show substantial variability, with RMSE averaging 15.47 but ranging from 0.22 to 469.86, indicating that while 75% of stocks achieve manageable prediction errors (below 13.39), extreme outliers suggest catastrophic failures for specific stocks or market conditions. MAPE results further confirm the bifurcated performance, averaging 10.8% with a median of 2.8%, where half the stocks maintain reasonable relative accuracy, but some exceed 95% error rates, indicating the model's effectiveness varies dramatically across different stock characteristics. The concerning R^2 mean of -4.25, despite a median of 0.84, reveals that while many stocks achieve strong explanatory power, extreme negative outliers severely impact overall performance.

Classification metrics present mediocre results with 51.1% accuracy barely exceeding random chance, with a median accuracy of 50.6% reinforcing the proximity to random performance across the dataset. The precision-recall imbalance (66.9% vs 53.9%) indicates conservative bias where the model misses profitable opportunities despite reasonable accuracy when signalling upward movements, with precision showing the most consistent behaviour through a tight interquartile range of 4.6%. The F1-Score of 59.3% confirms moderate but insufficient classification performance, ultimately revealing a model suitable only for a subset of stocks with specific characteristics rather than providing reliable universal predictions across diverse market conditions.

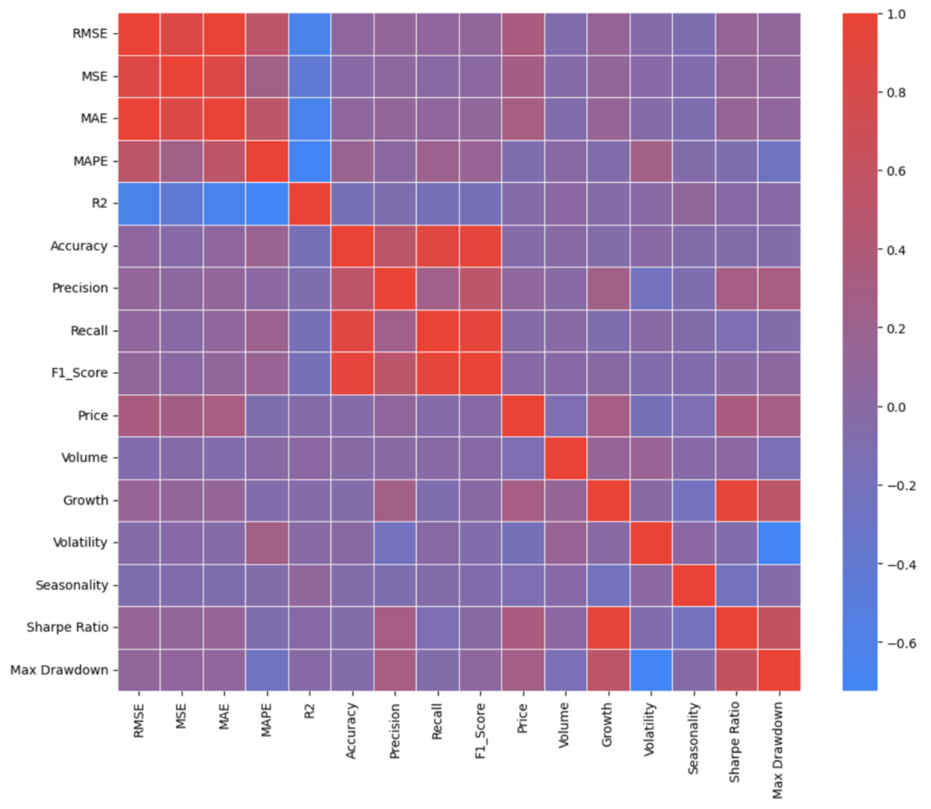

Figure 5.3: Correlation between performance and characteristics

The correlation analysis in Figure 5.3 provides quantitative analysis for the performance patterns, revealing the most significant relationships between model effectiveness and stock characteristics. The strongest correlations appear between price and the regression metrics, showing that lower-priced stocks have better performance. There's also a correlation present between volatility and some performance metrics. A positive correlation with MAPE and a negative correlation with Precision, which demonstrates that higher volatility stocks systematically degrade model performance by amplifying prediction errors and reducing signal reliability, confirming results found in the article analysis [1], [28]. Growth, Sharpe ratio, and max drawdown exhibit similar correlation patterns with model performance metrics, all showing positive relationships with classification accuracy and negative correlations with regression errors, indicating that stocks with superior risk-adjusted returns create more predictable patterns.

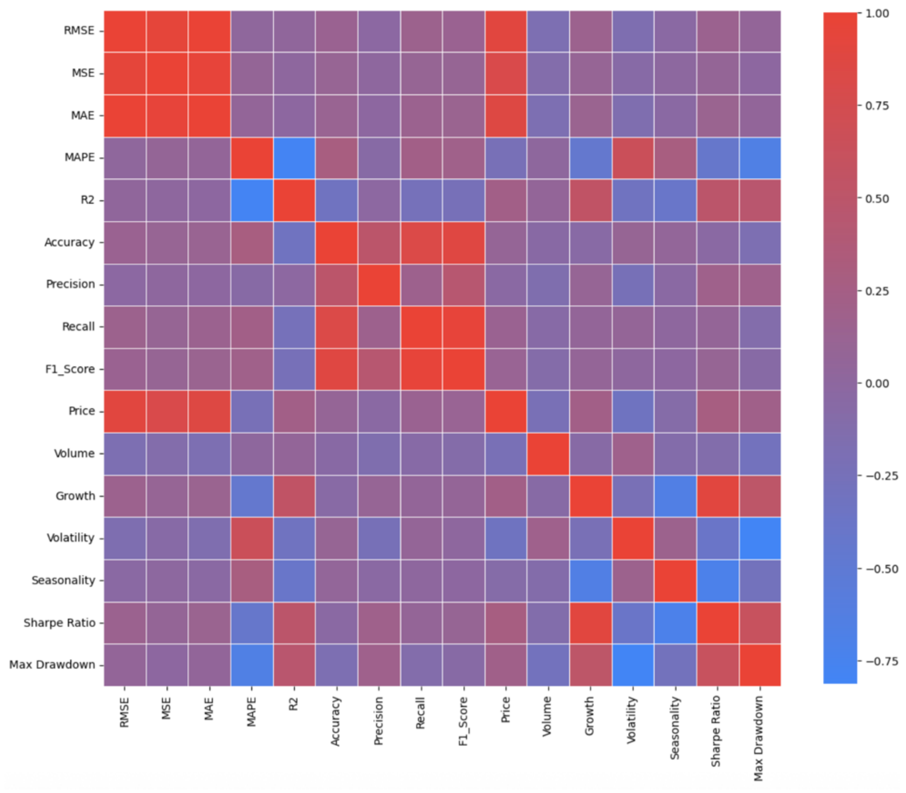

Figure 5.4: Correlation between performance and characteristics for the good performing stocks

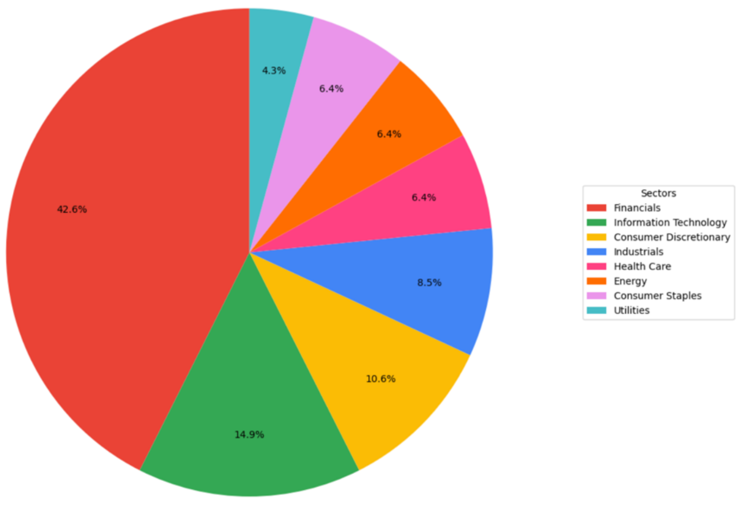

Figure 5.5: Distribution of sectors for the good performing stocks

The performance segmentation analysis reveals a stark distribution where only 11.7% of stocks achieve "Good" status, while the majority falls into "Average" (47.1%) and "Bad" (41.2%) categories, fundamentally validating the model's selective effectiveness across different market segments. The correlation matrix in Figure 5.4, further reinforces the previous correlation analysis, where lower prices and smaller volatility show better results. Additionally, growth, Sharpe ratio, and max drawdown now demonstrate strong negative correlations with MAPE and strong positive correlations with R², indicating that stocks with superior risk-adjusted characteristics generate more accurate percentage-based prediction and more predictable variance structures that the model can effectively capture and explain. As observed in Figure 5.5, Financials represents 42.6% of the good performing stocks, while the others have similar distributions as initially. This sector's dramatic overrepresentation among good performers reflects its alignment with the model's predictive strengths, as financial stocks typically exhibit the favourable characteristics identified, although they tend to have bigger prices, they have a moderate-low volatility, very high growth and good risk-adjusted metrics.

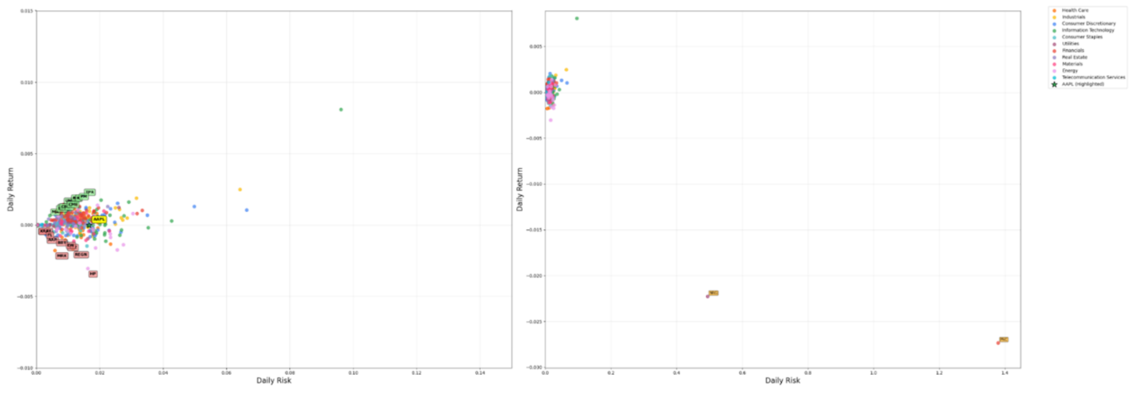

Figure 5.6: Risk-Return plot optimized and at full scale

As shown in Figure 5.6, the risk-return analysis reveals a concentrated distribution of stocks within the low-risk, moderate-return quadrant. This trade-off, opposing the previous analysis, predictably favours higher priced stock with enhanced growth, since these stocks tend to have higher percentage price changes. It also favours lower volatility ranges and bigger risk-adjusted metrics aligning with the correlation analysis findings, confirming that reduced price variability enhances model predictive accuracy and generates more reliable trading signals. Financials and Information Technology dominate top performers again, as these sectors demonstrate higher prices and growth, and moderate volatility characteristics.

6. Conclusion

6.1 Research Summary and Main Findings

This thesis conducted a systematic empirical optimization and evaluation of NNs for stock price prediction through a comprehensive three-phase methodology. The research utilized a meticulously constructed dataset comprising S&P 500 stocks spanning June 2020 to June 2025, incorporating three distinct variable categories: univariate, market, and technical. This methodological framework enabled rigorous evaluation of variable categories, architectural choices, optimization strategies, and real-world applicability across diverse market conditions.

The first phase systematically evaluated LSTM and GRU architectures across 15 architectural configurations and three datasets using AAPL as a representative case to identify the best combination. The optimal configuration emerged as the GRU 3x8 architecture using market variables. Although the optimal model uses multiple layers, the analysis revealed that shallow architectures generally outperformed deeper networks, confirming the widespread use of shallow architectures in nearly every study analysed in the literature review. GRU models demonstrated superior performance over LSTM overall, primarily due to GRU's simplified gating mechanism proving more robust to feature noise and requiring fewer parameters. However, LSTM outperformed GRU specifically on univariate datasets, where LSTM's sophisticated memory management effectively captured price patterns without the complexity-induced overfitting that emerged with multivariate inputs. Variable selection analysis contradicted widespread assumptions about multivariate superiority, as univariate models achieved competitive performance against the market dataset, while technical indicators produced disappointing results likely due to multicollinearity and feature noise rather than theoretical limitations of technical analysis itself.

The second phase systematically optimized 18 preprocessing methods, 36 hyperparameters, and 7 regularization techniques using the previously identified optimal combination. The optimization revealed that minimal data manipulation consistently outperformed complex approaches, as financial data's inherent volatility and non-stationary characteristics resist traditional signal processing techniques that may inadvertently remove valuable market information. Hyperparameter tuning identified Adam optimizer with 0.001 learning rate, single-batch processing, ReLU activation, and MAPE loss function as optimal configurations. The evaluation of regularization methods demonstrated that the default configuration without additional regularization achieved optimal performance, challenging conventional NN practices and indicating that the GRU 3x8 architecture already provided appropriate model complexity for the dataset characteristics.

The comprehensive evaluation across 403 S&P 500 stocks revealed selective model effectiveness, with substantial performance dispersion indicating suitability for only specific market segments rather than universal application. Regression metrics showed concerning variability (RMSE averaging 15.47 but ranging from 0.22 to 469.86) and classification accuracy barely exceeded random chance at 51.1%. The model demonstrated preferences for particular characteristics: lower prices, reduced volatility, and superior risk-adjusted metrics. The stark performance segmentation where only 11.7% achieve "Good" status validated this selective effectiveness, with Financial sector stocks dramatically overrepresented (42.6%) among top performers due to their alignment with the favourable metrics. Risk-return analysis further confirmed these patterns, but showed more favourable results towards higher priced stocks, since they have bigger protentional gains.

6.2 Limitations and Future Work

The research utilized a relatively small 5-year dataset which may have inherently favoured shallow architectures over deeper networks. Larger datasets spanning multiple decades with thousands of stocks would likely enable deeper, more complex models to demonstrate their superior pattern recognition capabilities.

The single-stock optimization approach, while methodologically sound for controlled experimentation, may have introduced optimization bias toward AAPL's specific characteristics and failed to capture the reality that some stocks require more complex models while others benefit from simpler architectures due to varying market dynamics and underlying business characteristics. Future work can explore multi-stock optimization strategies and adaptive architectures that can automatically adjust complexity based on individual stock characteristics and market conditions.

The variable selection approach suffered from significant constraints, excluding sentiment indicators and other types of data due to API limitations, potentially missing crucial market drivers that could improve prediction accuracy. Future research could incorporate comprehensive feature engineering approaches including feature selection or advanced dimensionality reduction techniques to transform noise-prone indicator combinations into meaningful predictive signals.

The 5-run optimization methodology, necessitated by computational, time and environmental constraints, falls short of providing robust statistical reliability compared to recommended 10-20 runs, potentially introducing uncertainty in architectural selection and performance rankings.

The deliberate focus on traditional architectures, while enabling systematic evaluation and establishing performance baselines, represents a foundation for more sophisticated approaches rather than a final solution. The optimized GRU configuration should serve as a baseline for developing more complex hybrid and alternative models.

The computational requirements (14 minutes per stock) limit practical implementation, while the ultimate goal remains real-world investment application. Since this project uses daily predictions, it would only be feasible to use a sub-set of stocks, which aligns perfectly with the performance segmentation and risk-return analysis.

6.3 AI Models Assistance Declaration

While writing this thesis, Anthropic's Claude Sonnet 4 was utilized to assist in the refinement and enhancement of the written work. Specifically, this large language model was employed for assistance with improving clarity and coherence of various sections. After using this tool, all content was thoroughly reviewed and edited, with full responsibility maintained for the final work presented.

Additionally, during the research phase of this work, Google's NotebookLM was utilized to assist in processing and extracting relevant information from academic articles and research papers. This AI tool was employed to help identify and synthesize key findings, methodologies, and insights from the literature review process. All extracted information was subsequently verified against the original sources, with full responsibility maintained for the accuracy and interpretation of all cited material.

6.4 Financial Advice Disclaimer

This thesis is conducted solely for academic research purposes and is not intended to provide financial or investment advice. Readers should not rely on the contents, findings, or conclusions of this thesis for making actual financial investment decisions. The author disclaims any responsibility for financial losses that may result from the use or misinterpretation of the information contained in this work.

Bibliography

[1] K. Alkhatib, H. Khazaleh, H. A. Alkhazaleh, A. R. Alsoud, and L. Abualigah, 'A New Stock Price Forecasting Method Using Active Deep Learning Approach', *J. Open Innov. Technol. Mark. Complex.*, vol. 8, no. 2, p. 96, June 2022, doi: 10.3390/joitmc8020096.

[2] G. Sonkavde, D. S. Dharrao, A. M. Bongale, S. T. Deokate, D. Doreswamy, and S. K. Bhat, 'Forecasting Stock Market Prices Using Machine Learning and Deep Learning Models: A Systematic Review, Performance Analysis and Discussion of Implications', *Int. J. Financ. Stud.*, vol. 11, no. 3, p. 94, Sept. 2023, doi: 10.3390/ijfs11030094.

[3] H. He and S. Dai, 'A prediction model for stock market based on the integration of independent component analysis and Multi-LSTM', *Electron. Res. Arch.*, vol. 30, no. 10, p. 1, Dec. 2022, doi: 10.3934/era.2022196.

[4] The World Bank, *Global development finance: Analysis and summary tables*, vol. 1. Oxford University Press, 2000. Accessed: Sept. 19, 2025. [Online]. Available: https://documents.worldbank.org/pt/publication/documents-reports/documentdetail/627441468175767347

[5] J. Xiao, T. Deng, and S. Bi, 'Comparative Analysis of LSTM, GRU, and Transformer Models for Stock Price Prediction', Oct. 20, 2024, *arXiv*: arXiv:2411.05790. doi: 10.48550/arXiv.2411.05790.

[6] M. Sewell, 'The Efficient Market Hypothesis: Empirical Evidence', *Int. J. Stat. Probab.*, vol. 1, no. 2, Oct. 2012, doi: 10.5539/ijsp.v1n2p164.

[7] Z. Chen, W. Li, and J. Huang, 'Investor sentiment and optimizing traditional quantitative investments', *Int. Rev. Econ. Finance*, vol. 101, p. 104227, July 2025, doi: 10.1016/j.iref.2025.104227.

[8] J. Liu, 'Navigating the Financial Landscape: The Power and Limitations of the ARIMA Model', *Highlights Sci. Eng. Technol.*, vol. 88, pp. 747–752, Mar. 2024, doi: 10.54097/9zf6kd91.

[9] J. K. Mutinda and A. K. Langat, 'Stock price prediction using combined GARCH-AI models', *Sci. Afr.*, vol. 26, p. e02374, Dec. 2024, doi: 10.1016/j.sciaf.2024.e02374.

[10] A. Razouk, M. E. M. Falloul, A. Harkati, and F. Touhami, 'Performance evaluation of technical indicators for forecasting the moroccan stock index using deep learning', *Indones. J. Electr. Eng. Comput. Sci.*, vol. 32, no. 3, pp. 1785–1794, Dec. 2023, doi: 10.11591/ijeecs.v32.i3.pp1785-1794.

[11] Grand View Research, 'Deep Learning Market Size And Share | Industry Report 2030', 2024. Accessed: Sept. 19, 2025. [Online]. Available: https://www.grandviewresearch.com/industry-analysis/deep-learning-market

[12] O. A. Montesinos López, A. Montesinos López, and J. Crossa, 'Overfitting, Model Tuning, and Evaluation of Prediction Performance', in *Multivariate Statistical*

Machine Learning Methods for Genomic Prediction, O. A. Montesinos López, A. Montesinos López, and J. Crossa, Eds, Cham: Springer International Publishing, 2022, pp. 109–139. doi: 10.1007/978-3-030-89010-0_4.

[13] R. Levine and Sara Zervos, 'Stock Markets, Banks, and Economic Growth', *Am. Econ. Rev.*, vol. 88, no. 3, pp. 537–558, 1998 26 AD.

[14] R. J. Shiller, 'From Efficient Markets Theory to Behavioral Finance', *J. Econ. Perspect.*, vol. 17, no. 1, pp. 83–104, Mar. 2003, doi: 10.1257/089533003321164967.

[15] J. J. Siegel and J. D. Schwartz, 'The Long-term Returns on the Original S&P 500 Firms', *Financ. Anal. J.*, vol. 62, no. 1, pp. 18–31, Mar. 2006, doi: 10.2469/faj.v62.n1.4055.

[16] Ebenezer Asem and Shamsul Alam, 'The Role of the S&P 500 Index Constituents in Tracking the U.S. Equity Market', *Can. Cent. Sci. Educ.*, vol. 4, no. 12, Oct. 2012, doi: 10.5539/ijef.v4n12p15.

[17] J. M. Maheu and T. H. McCurdy, 'Do high-frequency measures of volatility improve forecasts of return distributions?', *J. Econom.*, vol. 160, no. 1, pp. 69–76, Jan. 2011, doi: 10.1016/j.jeconom.2010.03.016.

[18] P. Kumar, L. Hota, V. A. Tikkiwal, and A. Kumar, 'Analysing Forecasting of Stock Prices: An Explainable AI Approach', *Procedia Comput. Sci.*, vol. 235, pp. 2009–2016, Jan. 2024, doi: 10.1016/j.procs.2024.04.190.

[19] S. J. Brown, W. Goetzmann, R. G. Ibbotson, and S. A. Ross, 'Survivorship Bias in Performance Studies', *Rev. Financ. Stud.*, vol. 5, no. 4, pp. 553–580, 1992.

[20] B. F. Tivnan *et al.*, 'Price Discovery and the Accuracy of Consolidated Data Feeds in the U.S. Equity Markets', *J. Risk Financ. Manag.*, vol. 11, no. 4, p. 73, Oct. 2018, doi: 10.48550/arXiv.1810.11091.

[21] R. J. Hyndman, *Forecasting: Principles & Practice*, 3rd edn. 2014.

[22] A. Tawakuli, B. Havers, V. Gulisano, D. Kaiser, and T. Engel, 'Survey:Time-series data preprocessing: A survey and an empirical analysis', *J. Eng. Res.*, vol. 13, no. 2, pp. 674–711, June 2025, doi: 10.1016/j.jer.2024.02.018.

[23] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, 'A review on outlier/anomaly detection in time series data', Feb. 2020, doi: 10.48550/arXiv.2002.04236.

[24] Z. Fathali, Z. Kodia, and L. Ben Said, 'Stock Market Prediction of NIFTY 50 Index Applying Machine Learning Techniques', *Appl. Artif. Intell.*, vol. 36, no. 1, p. 2111134, Dec. 2022, doi: 10.1080/08839514.2022.2111134.

[25] O. A. Montesinos López, A. Montesinos López, and J. Crossa, 'Fundamentals of Artificial Neural Networks and Deep Learning', in *Multivariate Statistical Machine Learning Methods for Genomic Prediction*, O. A. Montesinos López, A. Montesinos López, and J. Crossa, Eds, Cham: Springer International Publishing, 2022, pp. 379–425. doi: 10.1007/978-3-030-89010-0_10.

[26] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, 'Deep learning', *MIT Press*, p. 800, 2016, doi: 10.1007/s10710-017-9314-z.

[27] O. A. Montesinos López, A. Montesinos López, and J. Crossa, 'Artificial Neural Networks and Deep Learning for Genomic Prediction of Continuous Outcomes', in *Multivariate Statistical Machine Learning Methods for Genomic Prediction*, O. A. Montesinos López, A. Montesinos López, and J. Crossa, Eds, Cham: Springer International Publishing, 2022, pp. 427–476. doi: 10.1007/978-3-030-89010-0_11.

[28] O. E. Orsel and S. S. Yamada, 'Comparative Study of Machine Learning Models for Stock Price Prediction', Jan. 31, 2022, *arXiv*: arXiv:2202.03156. doi: 10.48550/arXiv.2202.03156.

[29] Mohd. R. Ab.Khalil and A. Abu Bakar, 'A Comparative Study of Deep Learning Algorithms in Univariate and Multivariate Forecasting of the Malaysian Stock Market', *Sains Malays.*, vol. 52, no. 3, pp. 993–1009, Mar. 2023, doi: 10.17576/jsm-2023-5203-22.

[30] J . W. Choi and Y. Choi, ‘A Study of Prediction of Airline Stock Price through Oil Price with Long Short-Term Memory Model’, *Int. J. Adv. Comput. Sci. Appl. IJACSA*, vol. 14, no. 5, May 2023, doi: 10.14569/IJACSA.2023.0140509.

[31] A. Ifleh and M. El Kabbouri, ‘Stock price indices prediction combining deep learning algorithms and selected technical indicators based on correlation’, *Arab Gulf J. Sci. Res.*, vol. 42, no. 4, pp. 1237–1256, Oct. 2023, doi: 10.1108/AGJSR-02-2023-0070.

[32] M. K. Paul and P. Das, ‘A Comparative Study of Deep Learning Algorithms for Forecasting Indian Stock Market Trends’, *Int. J. Adv. Comput. Sci. Appl. IJACSA*, vol. 14, no. 10, Oct. 2023, doi: 10.14569/IJACSA.2023.0141098.

[33] Kenrick, S. C. Ivancio, E. Isabellini, C. G. S. Adhi, and A. Prasetyo, ‘Multivariate stock market forecasting using lstm on Indonesian banking stock market’, *Procedia Comput. Sci.*, vol. 245, pp. 1047–1056, Jan. 2024, doi: 10.1016/j.procs.2024.10.333.

[34] S. Hartanto and A. A. S. Gunawan, ‘Temporal Fusion Transformers for Enhanced Multivariate Time Series Forecasting of Indonesian Stock Prices’, *Int. J. Adv. Comput. Sci. Appl. IJACSA*, vol. 15, no. 7, July 2024, doi: 10.14569/IJACSA.2024.0150713.

[35] A. Makinde, ‘Optimizing Time Series Forecasting: A Comparative Study of Adam and Nesterov Accelerated Gradient on LSTM and GRU networks Using Stock Market data’, Sept. 28, 2024, *arXiv*: arXiv:2410.01843. doi: 10.48550/arXiv.2410.01843.

[36] D. M. Teixeira and R. S. Barbosa, ‘Stock Price Prediction in the Financial Market Using Machine Learning Models’, *Computation*, vol. 13, no. 1, p. 3, Jan. 2025, doi: 10.3390/computation13010003.

[37] B. Y. Dwiandiyanta, R. Hartanto, and R. Ferdiana, ‘Harnessing Deep Learning and Technical Indicators for Enhanced Stock Predictions of Blue-Chip Stocks on the Indonesia Stock Exchange (IDX)’, *Eng. Technol. Appl. Sci. Res.*, vol. 15, no. 1, pp. 20348–20357, Feb. 2025, doi: 10.48084/etasr.9850.

[38] V. Chang, Q. A. Xu, A. Chidozie, and H. Wang, ‘Predicting Economic Trends and Stock Market Prices with Deep Learning and Advanced Machine Learning Techniques’, *Electronics*, vol. 13, no. 17, p. 3396, Jan. 2024, doi: 10.3390/electronics13173396.