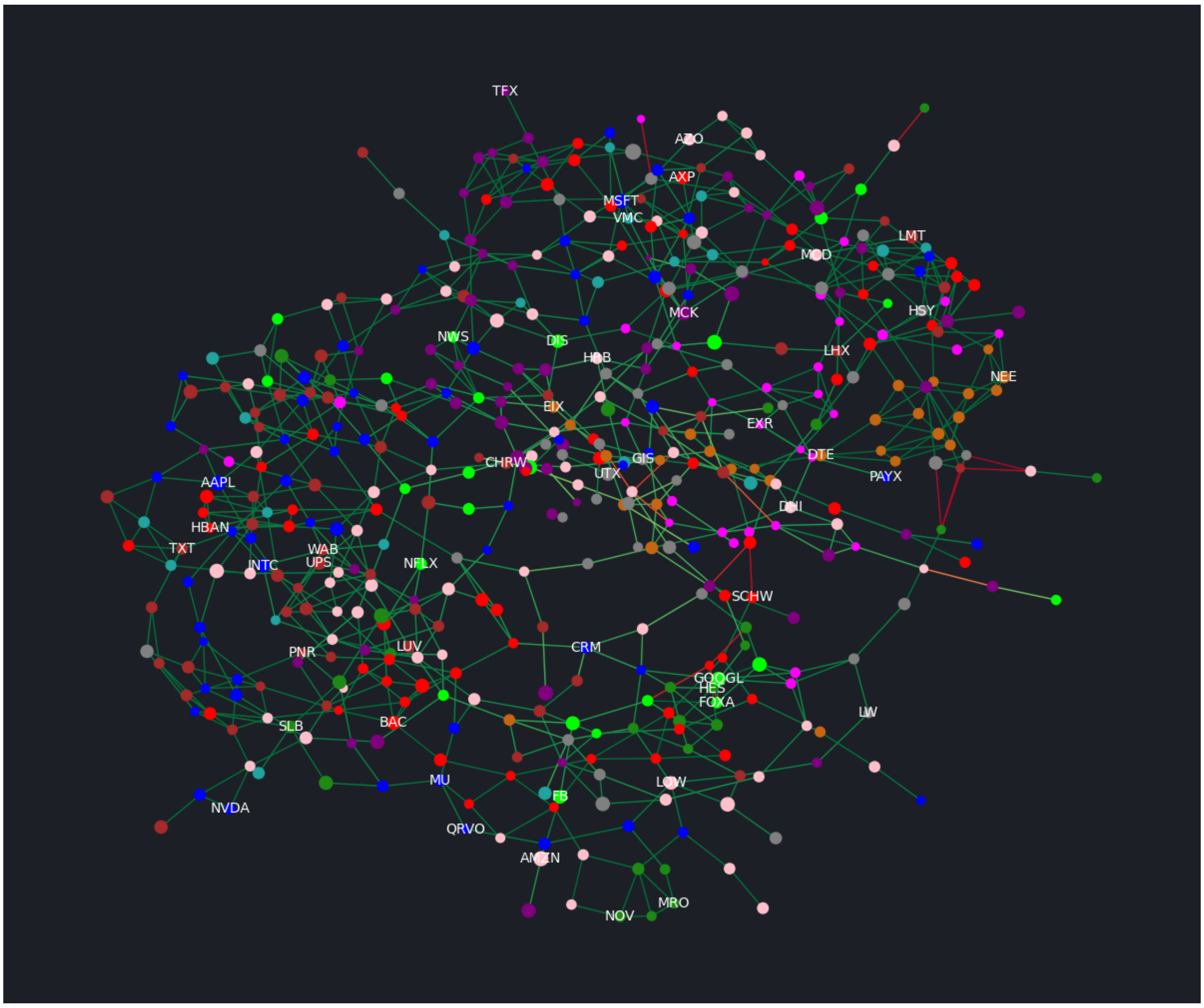
A comparative study of AT-GCN and LSTM for Portfolio Management

Datascience Msc.
Thesis - 20-12-2024

Author/Student:
Christian Emil Haldan
cemh@itu.dk

Supervisor:
Maria Sinziiana Astefanoaei
msia@itu.dk



Abstract

This thesis explores the performance advantages of framing stock market data as a graph problem, in contrast to treating individual stocks as independent input entries. By employing an AT-GCN, (a spatio-temporal graph neural network), and contrasting its performance with an LSTM, the study aims to evaluate the efficacy of graph-based representations in portfolio management. The dataset spans 28 years and includes 1,092 stocks, all of which were part of the S&P 500 at some point during the datasets period. Extensive feature engineering was conducted on price and fundamental data to emulate the depth of information typically considered by institutional investors.

Both models demonstrated a moderate edge over the S&P 500 during the test period. However, the results of the AT-GCN did not indicate a clear advantage over the LSTM, suggesting that a holistic market-wide graph structure, does not provide a significant improvements, over a model independently evaluating stock data. The code base is available here - Emil Haldan - GitHub.

1 Introduction

Portfolio management is the process of selecting, prioritizing, and managing investments to achieve specific financial objectives, typically balancing risk and return to optimize performance over time (Bodie et al., 2014). It plays a critical role in wealth preservation and growth, directly influencing the financial security of individuals and institutions. Predicting financial market behavior accurately remains an enduring challenge due to the highly dynamic nature of markets, their complex inter dependencies, and the multitude of factors that influence stock prices, including macroeconomic events, geopolitics, and investor sentiment. Furthermore, the sheer volume, variety, and velocity of financial data add layers of complexity to deriving actionable insights.

Advancements in machine learning (ML) have introduced new possibilities in financial markets, offering methods to process vast datasets, uncover intricate patterns, and make data-driven decisions. ML techniques have shown to improve forecasting accuracy, optimize portfolio allocation, and assist decision making for automated trading strategies. Methods also involve anomaly detection, sentiment analysis, and risk assessment, enhancing overall market efficiency (Heaton et al., 2017).

This thesis explores the application of deep learning models within the domain of portfolio management optimization, by assessing the benefits of modeling financial data as a graph problem. To uncover this, we will be comparing the predictive capabilities of an AT-GCN - a time series graph neural network -, and an LSTM - a time series neural network using tabular data.

The task for each model is to define a monthly portfolio of stocks from the S&P 500, by rating available stocks based on market data and quarterly reports. The AT-CGN will utilize a node-static graph structure of the S&P 500, where each stock is represented as a node with features derived from price data, and quarterly reports. The LSTM will only consider the features provided for each window of data for each stock, remaining unexposed to the specific ticker, or it's associations with other stocks. The inspiration of comparing the two input data structures was inspired by the praise of GNNs for portfolio management mentioned in prior research for portfolio management (Soleymani and Paquet, 2021) (Sun et al., 2024).

Multiple evaluation methods are employed to assess the models' predictive capabilities. Beyond standard deep learning metrics, a backtesting framework evaluates model constructed portfolios against a ”Buy and hold the S&P 500” strategy, focusing on metrics such as annualized returns and risk.

This study provides an unbiased comparison of temporal GNN-based models and LSTM models to assess whether framing portfolio management as a graph problem enhances predictive accuracy and portfolio performance. The comparison focuses on the impact of providing models with holistic market-wide information (AT-GCN) versus input data limited to individual assets (LSTM).

2 Background and Literature Review

This section provides the foundational context and theoretical underpinnings for the research later in this thesis, by unraveling both the domain of portfolio management and research within graph neural networks (GNNs). As this thesis and project is data science oriented, the section of portfolio management will briefly go into details of current landscape of the field, alongside introducing terminology, bridging the gap from financial data to computer science. The latter sections will introduce prior research applied specifically for this field.

2.1 Portfolio Management

Before going into the details of the paper, prior information and norms of the domain are to be established, in order to appreciate the domain specific evaluation in later sections of this paper.

Portfolio management summarized, boils down to the art of strategic asset allocation to achieve specific financial objectives - balancing risk, returns, and market impact. To contextualize its relevance, consider the following narrative: suppose you have some money saved up and would like it to grow. There are various options available, each with different incentives, benefits, and risks associated. Apart from real-estate investments, one of the most popular methods of investing is the stock market. You could either invest broadly by purchasing an index like the SPY (which tracks the S&P 500) or selectively choose individual stocks that align with your preferences or beliefs. If you choose to select individual stocks, there might be some strategy or rationalization that defines the validity of an investment.

Regardless of one's investment strategy or familiarity with the stock market, every decision is ultimately guided by a rationale—an evaluation of the inherent risk-reward trade-off. For instance, the individual selecting specific stocks for their portfolio may be prepared to accept greater risk while anticipating higher returns. This specific example introduces considerations often addressed by principles derived from Modern Portfolio Theory (Markowitz, 1952).

1. **Portfolio**: A portfolio represents a collection of financial assets, such as stocks, assembled to achieve specific financial objectives.
2. **Efficient Portfolios**: At any given time, certain stocks yield higher returns than others. An efficient portfolio aims to capitalize on these opportunities by selecting a combination of assets that maximize returns for a given level of risk. An Efficient portfolio is the hypothetical scenario of the perfect investments taken at any given time.
3. **Diversification Reduces Risk**: Expanding a portfolio to include a variety of stocks can significantly mitigate risk by spreading exposure across different assets. However, this often comes at the cost of slightly lower potential returns.
4. **Risk-Return Trade-Off**: Managing risk typically involves sacrificing potential rewards. Investors must consider the potential risk against the potential rewards.
5. **Long-Term Focus**: A portfolio's performance should be evaluated over extended time horizons, accounting for both favorable and unfavorable market conditions. This perspective aggregates measures of returns and risks over time.

To asses these considerations, data-driven insights offer a systematic approach to rationalize investment decisions. To quantify the performance of ones historical portfolio composition, metrics based on the returns, and the risk are often utilized. Specifically, the average annualized returns Equation 1, the Sharpe ratio Equation 2 is the formula for expressing the risk of ones portfolio over a duration.

$$\text{Average Annualized Returns} = \left(\frac{V_f}{V_i}\right)^{\frac{1}{n}} - 1 \quad (1)$$

Where V_f is the final portfolio value

V_i is the initial portfolio value

n is the number of years

$$\text{Sharpe Ratio} = \frac{E[R_p - R_f]}{\sigma_p} \tag{2}$$

Where R_p: Portfolio return

R_f: Risk-free rate return

$E[R_p - R_f]$: Excess return of the portfolio

σ_p = Standard deviation of the portfolio's return

$$\text{MDD} = \max_{t\in[1,T]} \left(\frac{P_{\text{peak}}(t) - P_{\text{trough}}(t)}{P_{\text{peak}}(t)} \right) \tag{3}$$

Where P is the Portfolios value at t. MDD, represents the largest negative difference in future portfolio value.

Portfolio management strategies can range from passive approaches, such as index tracking, to actively managed methods employed by hedge funds and institutional investors. Hedge funds, in particular, play a critical role in modern financial markets, employing diverse strategies such as arbitrage, quantitative analysis, and multi-strategy approaches. As of the first half of 2024 / year to date (YTD), hedge funds reported an average net return of 6.1%. Among those strategies, quantitative approaches were the best performers (+8.7%), followed by Equity Long-Short stateges at 8.17%, in contrast to the performance of the S&P 500 index gaining 15.14% (Aurum Research, 2024).

The current state of portfolio management reflects a growing reliance on data-driven techniques. Quantitative strategies, leveraging advanced statistical and machine learning methods, have also shown consistent performance, with five-year average annualized returns of 8.7% and a Sharpe ratio of 1.3. (Aurum Research, 2024).

These methods stand in contrast to traditional strategies that rely heavily on human judgment and qualitative analysis. However, the increasing complexity and interconnectedness of financial markets have highlighted the limitations of purely human-centric approaches, driving the adoption of towards more sophisticated tools.

Despite these advancements, challenges remain. For instance, market volatility and concentration, particularly in sectors like technology, have led to uneven returns, underscoring the need for robust, unbiased, and adaptable portfolio management strategies.

2.2 Deep learning in Portfolio Management

The application of deep learning in portfolio management has gained significant attention in recent years, driven by the increasing availability of financial data and advancements in machine learning techniques. This section bridges the domain of portfolio management with related research of applied deep learning on financial data.

One of the earliest research papers recording the application of neural networks in financial modeling can be found in the work of Atsalakis and Valavanis (2009) (Atsalakis and Valavanis, 2009), however, it's presumed that previous non-public research has been performed earlier. Their study leveraged a neuro-fuzzy methodology to forecast short-term stock market trends, illustrating the potential of machine learning in financial forecasting. Further supporting the claim of the inherent challenge of stock market prediction is supported by the work of (Cao et al., 2019), applying LSTM's for predicting daily close prices.

The use of Long Short-Term Memory (LSTM) networks, a type of recurrent neural network, has been previously been applied for time-series forecasting in finance. The research of (Siami Namin and Siami Namin, 2018) compare LSTM models with traditional economic models like ARIMA for financial time series forecasting, demonstrating the superiority of LSTMs in reducing error rates and capturing nonlinear dependencies.

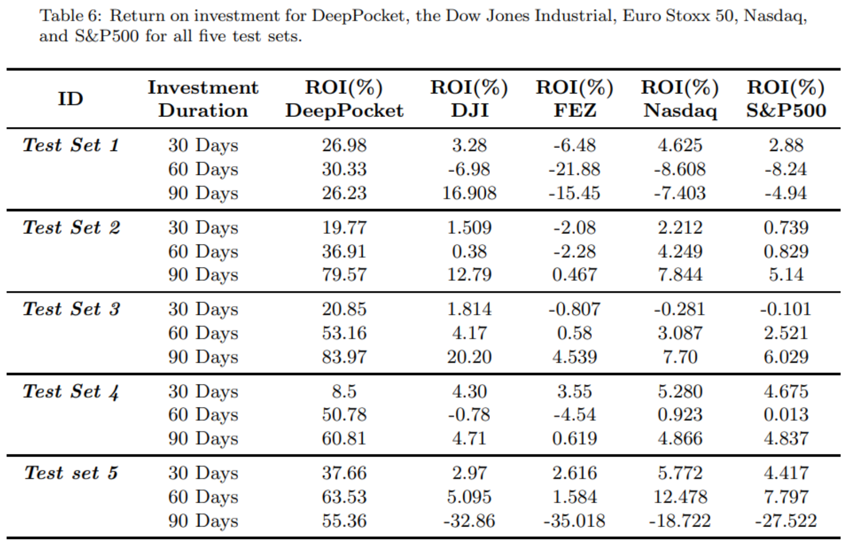


Table 6: Return on investment for DeepPocket, the Dow Jones Industrial, Euro Stoxx 50, Nasdaq, and S&P500 for all five test sets.

ID	Investment Duration	ROI(%) DeepPocket	ROI(%) DJI	ROI(%) FEZ	ROI(%) Nasdaq	ROI(%) S&P500
Test Set 1	30 Days	26.98	3.28	-6.48	4.625	2.88
	60 Days	30.33	-6.98	-21.88	-8.608	-8.24
	90 Days	26.23	16.908	-15.45	-7.403	-4.94
Test Set 2	30 Days	19.77	1.509	-2.08	2.212	0.739
	60 Days	36.91	0.38	-2.28	4.249	0.829
	90 Days	79.57	12.79	0.467	7.844	5.14
Test Set 3	30 Days	20.85	1.814	-0.807	-0.281	-0.101
	60 Days	53.16	4.17	0.58	3.087	2.521
	90 Days	83.97	20.20	4.539	7.70	6.029
Test Set 4	30 Days	8.5	4.30	3.55	5.280	4.675
	60 Days	50.78	-0.78	-4.54	0.923	0.013
	90 Days	60.81	4.71	0.619	4.866	4.837
Test set 5	30 Days	37.66	2.97	2.616	5.772	4.417
	60 Days	63.53	5.095	1.584	12.478	7.797
	90 Days	55.36	-32.86	-35.018	-18.722	-27.522

Figure 1: (Soleymani and Paquet, 2021) Backtest results

Recent studies have also demonstrated the use of GNNs for portfolio management (Soleymani and Paquet, 2021). The paper demonstrates the application of the DeepPocket framework, employing GNNs and Reninforcement learning, to rebalance a predefined portfolio of assets. They demonstrate incredible results showing the return on investment of 5 testing periods of 90 days between 2010 and 2020, using a pre-defined pool of stocks to consider consisting of 28 stocks figure 1.

Their results, particularly during the COVID-19 market crash, highlight the effectiveness of GNNs in managing financial portfolios under volatile conditions (Soleymani and Paquet, 2021). However, their work has notable limitations, such as the omission of comparisons to an equally weighted portfolio of the 28 stocks, seen in figure 1. Furthermore, the study neglects to account for trading costs during portfolio rebalancing, an important factor in real-world applications.

Similarly, (Sun et al., 2024) applied GNNs combined with a reinforcement learning agent for portfolio optimization, using their another predefined portfolio of 10 stocks, also demonstrating incredible results in comparison to the S&P 500. Differing from (Soleymani and Paquet, 2021), (Sun et al., 2024) highlight their performance relative to a uniformly weighted portfolio of the same assets, as the red line in figure 2 and figure 3. Despite these achievements, their study shares similar limitations: the space of potential candidates is predefined. When considering the figures, the overall performance increase of employing their deep learning rebalancing model is

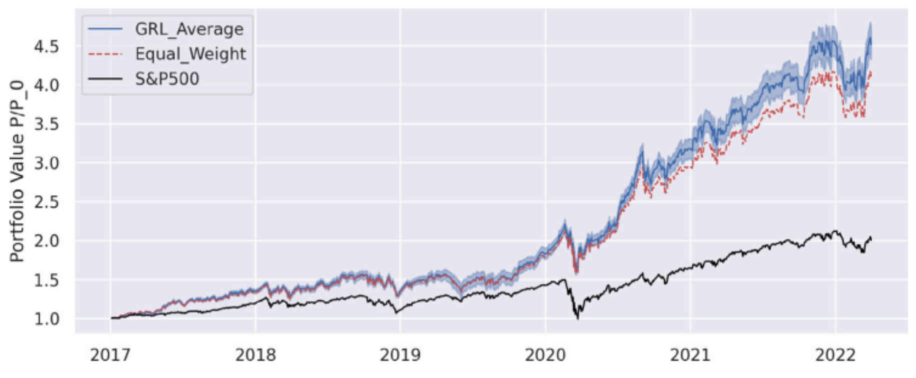


Fig. 6. Portfolio values on Top10 test dataset by GRL model (blue) and Equal Weight strategy (red), the black line indicates *S&P*.

Figure 2: (Sun et al., 2024) Backtest results (Top-10)

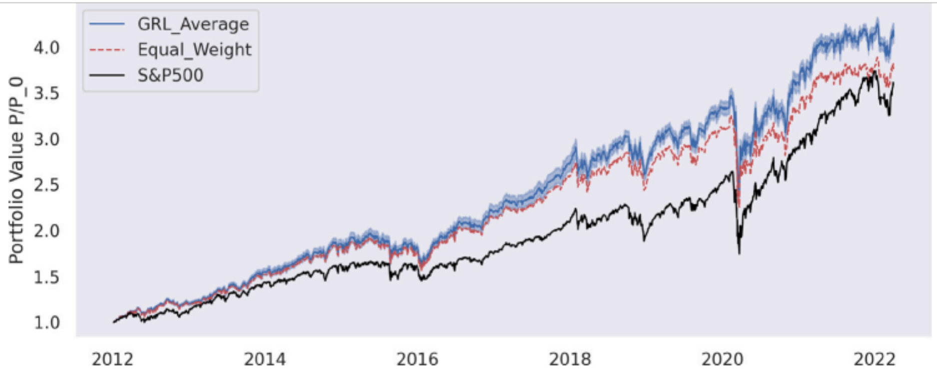


Fig. 7. Portfolio values on 28-stocks test dataset from 2012-01-02 to 2022-04-01 by GRL model (blue) and Equal Weight strategy (red), the black line indicates *S&P*.

Figure 3: (Sun et al., 2024) Backtest results (28-Stocks from (Soleymani and Paquet, 2021)

minimal, in contrast to an equal-weighted portfolio.

While prior research demonstrates the potential of GNNs and LSTMs in portfolio management, the neglect of trading costs, and use of predefined candidate spaces, leaves room for the comparative assessment of the benefits by defining portfolio management as a graph problem.

2.3 Graph Neural Networks (GNN's)

Introduced in the late 2000's (Scarselli et al., 2009) formalized the concept of GNNs, which are specifically constructed to handle non-Euclidean data structures, and take advantage of the spatial and relational structure of graphs. Graph Neural Networks (GNNs) are a class of neural networks designed to process or/and construct graph like data.

GNNs have found promising results in a wide range of applications, including molecular property prediction, recommendation engines, and traffic predictions. (Bongini et al., 2024) demonstrate the effectiveness of Composite Graph Neural Networks (CGNNs) for predicting molecular properties. Another unrelated paper demonstrate improved recommendation accuracy by effectively capturing the strengths of social ties and user preferences using a bipartite network of items and people (Fan et al., 2019). Lastly, GNNs have shown promising results within traffic prediction (Zhu et al., 2020), where they demonstrate the third iteration of their model showing improved accuracy in forecasting traffic speeds for different prediction horizons. This last paper in particular was eye catching, as the motivation for the architecture is focused specifically for temporal data, which is at the core of the problem regarding deep learning in portfolio management.

(Zhu et al., 2020) demonstrate the effectiveness of combining an attention mechanism (Vaswani et al., 2017), gated recurrent neural networks (GRU), with graph-convolutional neural networks (GCN) for timeseries graph problems. The paper showcases their third iteration of their Attention Temporal Graph Convolutional Network model (AT-GCN) used for traffic forcasting (Zhu et al., 2020) with an overall improvement to current models. The key advantages of this model is the design specifically for spatial temporal data, which was considered suitable to apply for this project.

2.3.1 The A3TGCN Model

The Attention Temporal Graph Convolutional Network (AT-GCN) represents a significant evolution in GNN architecture, combining three types on network properties: Graph Convolutional Networks (GCNs), Gated Recurrent Units (GRUs) and an attention mechanism (Vaswani et al., 2017). While it was originally proposed for traffic forecasting (Zhu et al., 2020), it is likely that the AT-GCN is capable of modeling other complex spatial and temporal dependencies, such as a graph representation of financial data. A representation of the model components can be seen in figure 4

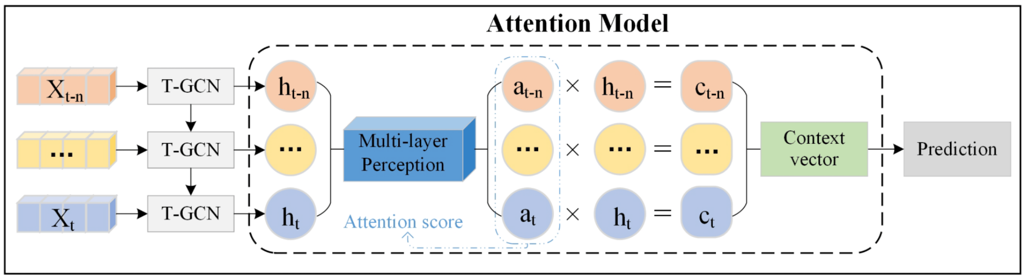


Figure 4: AT-GCN model architechture (Zhu et al., 2020)

Graph Convolutional Networks (GCNs) capture spatial dependencies by leveraging graph structures to propagate information between connected nodes. In the A3TGCN, GCN modules are used to extract topological features that represent relationships between nodes. Gated Recurrent Units (GRUs) capture temporal dependencies by modeling sequential data, identifying trends over time while maintaining computational efficiency compared to more complex recurrent models like LSTMs. The attention layer allows the model to weigh the importance of different time points, capturing global temporal variations that might otherwise be overlooked.

The AT-GCN architecture combine these components, making it particularly well-suited for time-series financial data. The GCN component enables the model to encode inter-stock relationships, such as correlations and dependencies, while the gated recurrent units (GRU) captures temporal trends of price data and financial reports. The attention mechanism further enhances this capability by highlighting significant historical patterns. By leveraging the A3TGCN's strengths, providing carefully tailored data to this model can enhance portfolio management by selecting attractive assets at the correct time, for a dynamic and profit-optimized portfolio.

2.4 The LSTM Model

Recurrent Neural Networks (RNNs) have been pivotal in modeling sequential data, with Long Short-Term Memory (LSTM) networks being one of the most widely adopted architectures for time series analysis. Although the first introduction of the LSTM is nearly 2 decades old (Hochreiter and Schmidhuber, 1997), the basis of this model architecture address the vanishing gradient problem commonly associated with traditional RNNs. This in turn enables the effective learning of long-term dependencies in time-series data. This capability makes LSTMs particularly well-suited for financial modeling, where historical trends play a significant role in forecasting.

A comparative study (Siami Namin and Siami Namin, 2018), demonstrates the LSTM model's efficacy compared to traditional statistical methods, such as ARIMA (Autoregressive integrated moving average). The comparative analysis of LSTMs and ARIMA models found that LSTMs reduced error

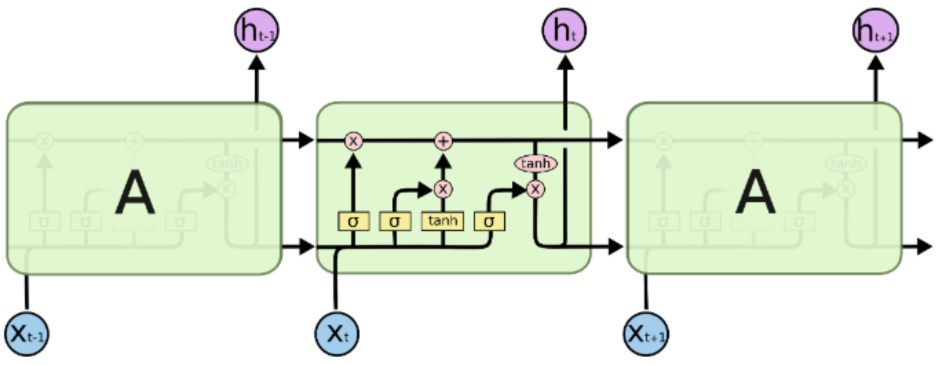


Figure 5: LSTM cell Colah's Blog , (Olah, 2015)

rates by 84–87% compared to ARIMA, showcasing the potential of deep learning models in capturing nonlinear dependencies and dynamic patterns in financial data. These results highlight the potential of LSTMs as suitable comparison for this particular task adressed in this research.

The LSTM model was chosen for this study as a benchmark for comparison with the GNN-based approach. Unlike GNNs, which incorporate the relational structure of the data, LSTMs treat each input independently, without considering the inter dependencies between stocks. This homogeneous treatment of input data enables us to isolate the added value of representing stocks as nodes, and connectivity based on stocks price correlation.

LSTMs achieve their unique capabilities through three core components: input gates, forget gates, and output gates, illustrated in figure 5. These gates control the flow of information through the network. The intuition of this structure is to retain, update, or discard information as needed.

The forget gate ensures that irrelevant information does not clutter the memory, while the input and output gates ensure that critical patterns are stored and utilized at the right times. While the detailed inner workings of LSTMs are beyond the scope of this discussion, the combination of gates and cell states allow LSTMs to effectively learn both short- and long-term dependencies in sequential data, hence it's name.

3 Data

In this section, we present the various datasets used in this project, outlining their roles and how they inter-connect to support the analysis. At the center of this study is the S&P 500 index, a benchmark consisting of the 500 largest publicly traded companies in the U.S. stock market. Any stock on the US stock market contains a substantial amount of information with various granularity.

Each company used in this study has meta-data such as the company's name, sector, age, listing date, and, if applicable, de-listing date, etc.

Complementing each stock, we have market data, which is the market price of each share of a stock at any point in time defined by real time trades. As larger companies often have their shares traded frequently, a measure defined as the volume, it's common practice to aggregated the price data into four key prices—Open, High, Low, and Close (OHLC)—over predefined intervals such as weekly (1W), daily (1D) or hourly (1H) periods.

The third type of data in this project is fundamental data, derived from quarterly reports that companies are legally obligated to publish. These reports provide insights of a company's financial health, detailing metrics that are highly relevant to investors and play a significant role in understanding a company's achievements or issues in the future.

Finally, we explain how this diverse set of data is collected, processed and stored. Using the EODHD API Python wrapper and a SQLite database, we efficiently access and extract the required information, finalized by combining data from multiple sources into a unified local database for future ease of access.

3.1 The S&P 500

The S&P 500, also know as the Standard & Poor's 500, is a stock market index that tracks the performance of 500 large publicly traded companies listed on stock exchanges in the United States. It is not only regarded as a benchmark for the overall performance of the U.S. equity market and the broader economy, but also a suitable benchmark for portfolio managers to compare their strategy against (Sun et al., 2024), making it one of the most followed indices in globally.

3.1.1 Historical Performance

The S&P 500 has historically demonstrated robust long-term growth, making it a cornerstone for

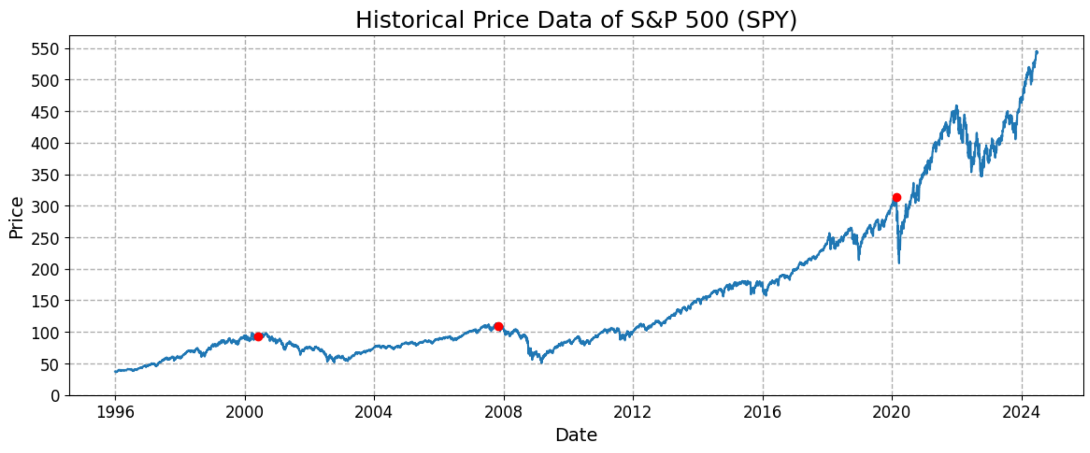


Figure 6: Adjusted Close Price data for the S&P 500 ETF (SPY)

investors seeking to track the performance of the U.S. equity market. Over several decades, the index has delivered an average annual return of approximately 11,04%, including dividends, and a standard deviation of the annual return at 17.93% table 1. Although very impressive, individual years can exhibit significant volatility. This performance reflects the index's role as a barometer of the U.S. economy, driven by its diverse representation of sectors and industries.

Statistic	**Anualized Return**
Mean	11.048 %
Standard Deviation	17.928 %
Minimum	-35.032 %
Median	15.149 %
Maximum	34.572 %
Start Date	1996-01-02
End Date	2024-07-01

Table 1: Summary Statistics of S&P 500

Market dynamics have evolved over time, with various periods dominated by specific sectors. For instance, the technology sector has played an increasingly prominent role in the index's performance in past 10 years, with companies such as Meta, Apple, Microsoft, Amazon, Alphabet most and recently NVIDIA, contributing significantly to overall returns. At the same time, economic downturns and external shocks, such as the dot-com bubble, the global financial crisis of 2008 and the COVID-19 pandemic, highlight the index's sensitivity to broader market conditions. figure 6 illustrates the historical price movement of the SPY ETF, which follows the S&P 500 index, along with some notable periods (red dots) highlighting turmoil in the financial market. For future reference as a bench mark of daily returns, the daily returns of the S&P 500 can be seen in figure 7.

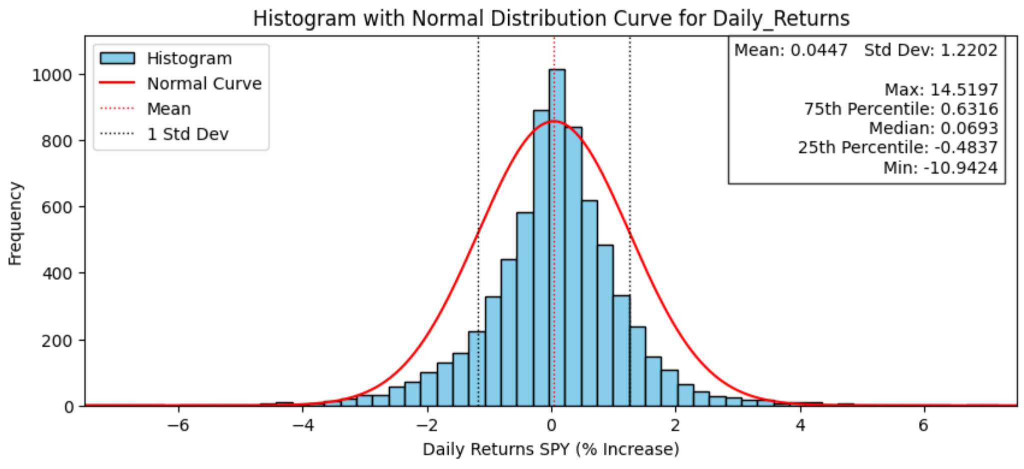


Figure 7: Daily Returns of S&P 500 (SPY)

3.1.2 Significance in Portfolio Management

The S&P 500's status as a benchmark stated by (Sun et al., 2024), is further underscored by its resilience and ability to recover from market disruptions. For instance, despite periodic downturns, the index has consistently rebounded to reach new all-time highs, demonstrating the strength of the U.S. equity market over the long term. This resilience, combined with its broad diversification in a wide variety of sectors, provide a comprehensive snapshot of the U.S. economy. Consequently, many mutual funds, exchange-traded funds (ETFs), and portfolio managers, use the S&P 500 as a benchmark for evaluating investment performance.

3.1.3 Composition and Selection Criteria

The S&P 500 is a dynamic list of 500 large companies (size defined by market capitalization), curated to represent a diverse cross-section of industries and sectors. The index undergoes a scheduled review and re-balancing by the S&P Index Committee. This occurs four times a year, in March, June, September, and December. Adjustments also occur between the scheduled re-balancing periods in response to corporate actions such as mergers, acquisitions, bankruptcies, or other events that impact a company's eligibility. Companies must meet specific eligibility criteria to be included in the index (S&P Dow Jones Indices, 2024).

1. **Market Capitalization**: Companies must have a market cap of at least $14.6 billion (as of 2024, subject to periodic adjustments by S&P Dow Jones Indices).

2. **Liquidity**: Stocks must demonstrate high trading volume and active market participation.

3. **U.S. Domicile**: Companies must be headquartered in the United States.

4. **Sector Diversification**: The index aims to cover a broad range of industries, ensuring that no single sector dominates the composition.

3.1.4 Source of historical composition

For this project, we emplployed an open source S&P 500 GitHub directory, which has a curated dataframe of the historical composition of the S&P 500 dating back from 1996-01-02, to 2024-07-08. This is used throughout the project to determine which stocks are eligible to trade with at a specific point in time.

3.2 The Stock Data

The stock data forms the backbone of this study, providing the essential information for the model and further analysis. The dataset includes 1092 stocks from the S&P 500 index, with data spanning a period from 1996-01-02 to 2024-07-01. This extensive time range of more than 28 years is ideal as a robust foundation for the training data set, while also allowing a significant portion of the dataset to be used for validation and testing purposes.

Each publicly traded company on the stock market is associated with a unique identifier known as a stock ticker, which will be referred to as the **ticker** in future sections. The ticker serves as a shorthand representation of the company on a specific stock exchange, enabling quick identification and access to its trading data. It is important to note, that tickers are unique only within a specific exchange. In this case, we are only dealing with stocks from the NASDAQ. Each company is accompanied by a variety of metadata that can include details such as the company name, listing date, headquarter address, the amount of employees, and much more, however, this project only makes use of the sector and the stock ticker in terms of non temporal data.

3.3 Price Data (OHLC)

When discussing stocks, price data is often the first and most prominent type of information considered. As introduced in section 3, price data is typically structured in the form of Open-High-Low-Close (OHLC) values, which represent aggregated trading prices of a stock within a specified time interval. For this project, we used OHLC data with 1-day intervals, as it was deemed appropriate considering extensive time span and inclusion of stocks in the dataset.

Date	**Open**	**High**	**Low**	**Close**	**Adjusted Close**
2024-06-06	1240.48	1255.87	1183.20	1209.98	120.98
2024-06-07	1197.70	1216.92	1180.22	1208.88	120.87
2024-06-10	120.37	123.10	117.01	121.79	121.77
2024-06-11	121.77	122.87	118.74	120.91	120.89
2024-06-12	123.06	126.88	122.57	125.20	125.19

Table 2: Example of OHLC, Adjusted Close Price before and after NVIDIA's 10 for 1 stock split

In this study, the adjusted close price was used to transform the OHLC data. The adjusted close price is commonly regarded as the most accurate representation of a stock's value at a given time as it accounts for corporate actions like stock splits and dividends, ensuring consistency over time. However, it is worth noting that while the adjusted close price corrects for these factors, it does not account for inflation. The choice to use adjusted close prices to modify OHLC values, align with standard practice in financial analysis, ensuring that the historical price data is accurate. An example of the adjusted close price along with raw OHLC values can be seen in table 2, where ignoring this pre-processing step would make it seem as if NVIDIA fell by 90% over night.

Companies perform stock splits and pay dividends for strategic reasons that directly influence their stock price and shareholder relationships. Stock splits make shares more affordable by reducing the share price, thereby improving accessibility for smaller investors and enhancing market liquidity. Dividends, on the other hand, are a way for companies to distribute a portion of their profits to shareholders, signaling financial health and rewarding investors with income for their trust. These corporate actions impact stock prices and are reflected in adjusted close prices, ensuring a consistent view of historical performance.

The dataset includes over 5.1 million rows of OHLC entries, with each row uniquely identified by a combination of the ticker and the date. Larger companies, such as Bank of America (ticker: BAC), are included in the S&P 500 throughout the entire duration of the dataset, and are responsible for 7,172 entries. The price data serves as a foundational element for feature engineering, providing the basis for constructing technical indicators, which are discussed in further detail in section 4.4.

3.4 Fundamental data

Fundamental data refers to the financial and operational information about a company that provides insights into its overall performance, financial health, and company value. This data is extracted from a company's quarterly statements, which are typically filed with regulatory authorities a few days to a few weeks after the end of each quarter (known as the **filing date**). The financial performance detailed in these statements corresponds to the previous quarter, known as the **effective date**. The effective date differs from companies based on their financial calendar, but tend to align with the quarters of the year, meaning that most companies have their effective dates for their reports at {31st of March, 30th of June, 30th of September, 31st of December}. Released four times a year, these statements provide a comprehensive overview of a company's profitability, assets, liabilities, and cash flow. In contrast, price data is influenced by market sentiment and short-term fluctuations. Fundamental data offers an alternative, and more stable assessment of the company's value.

3.4.1 Why is fundamental data important?

At its core, fundamental data helps investors and analysts evaluate the long-term potential of a company. It answers critical questions such as: ”How profitable is the company?”, ”Can it sustain its operations?”, ”How much debt does it have?”, and so on. These insights are crucial for assessing whether a stock is overvalued, undervalued, or fairly priced relative to the companies financial health. This type of data differs from price data which is influenced by

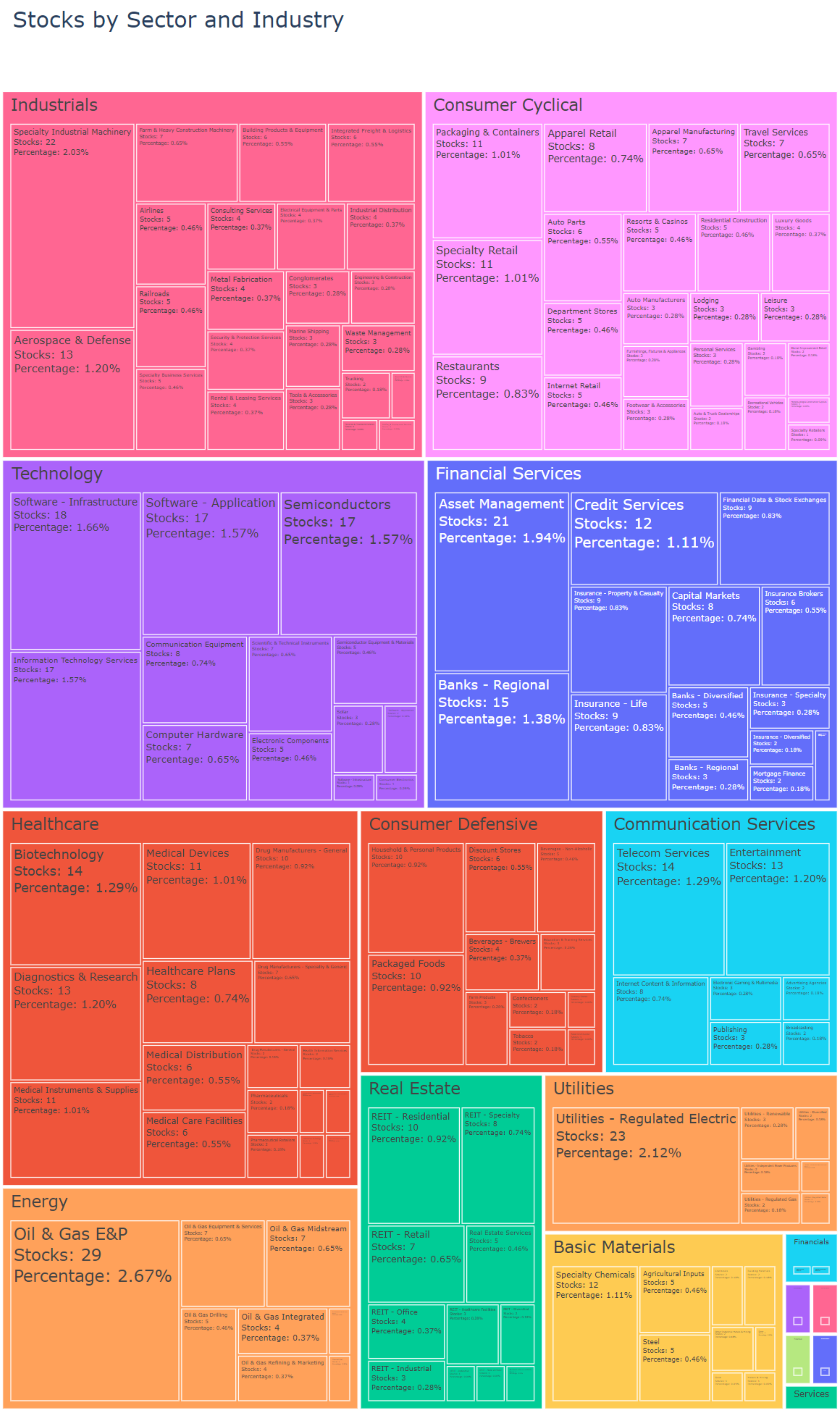


Figure 8: Sectors and Industries of all companies in the S&P 500 since 1996

short-term market sentiment and external factors like news or economic events.

3.4.2 The components of fundamental data

Fundamental data is derived from 3 primary components. Balance sheets, income statements, and cash flow statements. Together, they offer insights into different aspects of a company's operations and are essential for investors to evaluate a company's financial health and growth potential. Below, we explore some of the components, their meaning, and their importance to investors. In this report, only the attributes used in the machine learning pipeline are defined, however, each report contains vastly more information than what is provided below.

3.4.3 Income Statement

The income statement, is like a financial report card for a company over a specific period. Imagine it as a monthly household budget: the money you earn represents your revenue, the bills you pay are your expenses, and the amount left over is your profit. After considering all expenses, taxes, and allowances, etc. you are left with the net income. Similarly, the income statement reports a company's ability to generate revenue, manage costs, and achieve profitability. Below are definitions and examples of the metrics used in the project that are associated with the income statement:

1. **Total Revenue**

   Definition: Total revenue is the income a company generates from its primary operations, such as selling goods or services, during a specific period.

   Purpose: It reflects the company's scale of operations and market demand for its products or services.

   Example: If a retail store sells $1 million worth of products in a quarter, this amount would be recorded as total revenue.

2. **Gross Profit**

   Definition: Gross profit is the difference between total revenue and the cost of goods sold (COGS), which includes direct expenses like materials and labor.

   Purpose: It measures the profitability of a company's core operations, excluding indirect costs like marketing or administrative expenses.

   Example: If a company has $1 million in revenue and $600,000 in COGS, the gross profit is $400,000.

3. **EBIT** (Earnings Before Interest and Taxes)

   Definition: EBIT represents a company's operating profit before accounting for financing costs (interest) and taxes.

   Purpose: It provides a view of how efficiently the core business generates profit without being influenced by tax strategies or capital structure.

   Example: A company with a gross profit of $400,000 and $200,000 in operating expenses has an EBIT of $200,000.

4. **EBITDA** (Earnings Before Interest, Taxes, Depreciation, and Amortization)

   Definition: EBITDA expands on EBIT by excluding non-cash expenses like depreciation and amortization.

   Purpose: It provides a clearer picture of cash flow from operations, useful for comparing companies with different asset bases or capital structures.

   Example: If a company's EBIT is $200,000 and incurs $20,000 in depreciation and $10,000 in amortization, its EBITDA is $230,000.

5. **Income Before Tax**

   Definition: Income before tax (or pretax income) represents the company's profit after all expenses except taxes.

   Purpose: This metric shows profitability before government deductions, isolating the impact of operational and financing decisions.

   Example: If a company's EBIT is $200,000 and it pays $50,000 in interest, its income before tax is $150,000.

6. **Net Income** Definition: Also known as the ”bottom line,” net income is the profit remaining after all expenses, including taxes and interest, are deducted from total revenue.

   Purpose: Measures the overall profitability of the company.

   Example: If the income before tax is \$150,000 and taxes amount to \$30,000, the net income is \$120,000.

Balance Sheet The balance sheet can be thought of as a snapshot of a company’s financial health at a specific point in time. Imagine it as a personal financial statement: Your assets (cash, car, house) represent what you own, your liabilities (mortgage, credit card debt) are what you owe, and the difference between the two is your net worth. Similarly, a company’s balance sheet provides a detailed view of its resources, obligations, and shareholder equity. The balance sheet is structured around the accounting Equation 4:

$$\text{Equity} = \text{Total Assets - Liabilities} \tag{4}$$

The following are the key metrics used in this project associated with the balance sheet.

1. **Total Assets** Definition: Total assets represent everything a company owns that has monetary value. Assets are categorized into two main types, namely ”Current Assets” and ”Non-Current Assets”. As there are numerous types of assets within these categories, we used the combined value (Total Assets).

   Purpose: Total assets provide a measure of the resources available to the company to generate revenue and maintain operations.

   Example: A company with \$1 million cash, \$500,000 inventory, and \$3 million property has total assets of \$4.5 million.

1. **Total Liabilities** Definition: Total liabilities represent all financial obligations the company owes to others. Again there are numerous types of liabilities which wont be covered, as we only use the total liabilities for this project.

   Purpose: Total liabilities indicate the extent of the company’s debt and financial risk. This metric is typically considered as a measurement of the companies ability to meet it’s obligations.

   Example: If a company owes \$500,000 in short-term loans and \$1.5 million in long-term bonds, its total liabilities are \$2 million.

3.4.4 Cash Flow

The cash flow statement tracks the inflow and outflow of cash within a company over a specific period, offering insights into its liquidity and ability to sustain operations. Unlike the income statement, which includes noncash items like depreciation, the cash flow statement focuses solely on actual cash movements. The metrics applied for this project are:

1. **Capital Expenditure (CapEx)**

   Definition: Capital Expenditure represents the funds a company spends to acquire, maintain, or upgrade its physical assets, such as buildings, machinery, or technology. CapEx is a key indicator of investment in the company’s long-term growth.High CapEx may indicate aggressive expansion, while low CapEx may suggest a focus on cost containment or mature operations.

   Purpose: FCF is often used by investors to determine whether a company has the capacity to fund growth initiatives or pay dividends.

   Example:

2. **Free Cash Flow (FCF)**

   Definition: Free Cash Flow is the cash remaining after a company has covered its operational expenses and capital expenditures (CapEx). It represents the funds available for discretionary use, such as reinvestment, debt repayment, or distribution to shareholders $\text{FCF} = \text{Operating Cash Flow} - \text{CapEx Expenditures}$.

   Purpose: FCF indicates how much cash a company generates beyond what is needed to maintain or grow its asset base. Positive FCF suggests the company is self-sustaining and has flexibility to pursue opportunities or return value to shareholders.

Example: If a company has \$500,000 in operating cash flow and spends \$200,000 on CapEx, its FCF is \$300,000. This amount represents the discretionary funds available for reinvestment or shareholder returns.

Fundamental data forms the foundation for fundamental analysis, a widely used investment strategy aimed at identifying the intrinsic value of a company. By examining this data, investors can make informed decisions about buying, holding, or selling a stock, complementing the short-term signals provided by price data.

3.5 Obtaining the data with EODHD

The data for this project was sourced using the Python wrapper API provided by EOD Historical Data (EOD-HD). This platform offers a comprehensive suite of historical financial data, including the various datasets discussed earlier. The Python wrapper allows seamless access to the data, allowing for retrieval and integration into the analysis pipeline.

While some of the quarterly reports retrieved through EOD-HD were incomplete, the platform's pricing of \$99 per month made it a cost-effective choice compared to industry-leading alternatives like the Bloomberg Terminal or Refinitiv Eikon, which can cost between \$20,000 and \$25,000 annually (pricing, 2023a), (pricing, 2023b). This affordability, combined with its broad data coverage, made EOD-HD a suitable option for the scope and student budget of this project.

4 Methodology

This section outlines the methodological framework employed in this study, detailing the process of transforming raw financial data into actionable insights using the two different types of neural networks. The methodology is organized into distinct stages, where each step has been carefully designed to align with best practices in data science. The overall goal of this methodology is to fine tune the data and models for the task of composing a portfolio once a month, intended to outperform a buy-and-hold strategy of the S&P 500. Exploring the machine learning specific results, and the portfolio performances created of the two model types, we aim to uncover the increased performance of graph structure vs. tabular data.

4.1 Outline of the Methodology:

Data Acquisition and Data Cleaning The first step involved creating a database using SQLite and populating it with financial data retrieved via the EOD Historical Data API. Both price data and fundamental data underwent extensive cleaning and pre-processing to address inconsistencies, information leakage, and ensure accurate results.

Creating Technical Indicators and Valuation Metrics: Technical indicators and valuation metrics were calculated to provide additional features derived from price data and fundamental data, enriching the dataset for model training.

Target Definition: The target definition was established to group stocks into 5 ordinal categories, based on near future price fluctuations. Instead of using regression for the percentage change, it was considered more insightful to correctly predict growth potentials.

Feature Engineering: Comprehensive feature engineering was performed to extract and encode meaningful information from the attributes of the data. As the project contains various types of data, the transformations were carefully considered, assisted by prior research, statistics, and data visualizations.

Graph Creation: As a core part of this project is defining input data in a graph representation. We describe each trading day over a series of years as a graph. The prior mentioned features define the node data for each stock and edges were defined based on the highest price correlations between each of the stocks.

Train, Validation, and Test Splits To ensure robust model evaluation, the dataset was split into training, validation, and test sets while preserving the temporal structure of the data.

Model Selection and Configuration: The primary models used in this project was the AT3-GCN and the LSTM using Pytorch. This subsection describes the rationale behind the this model selection, along with methods and considerations for hyperparameter tuning during training.

Backtesting and Evaluation: The evaluation framework for the models are comprehensive, incorporating both model-specific metrics and techniques for interpreting and simulating its performance on the validation and test data. The goal of the model is not to achieve perfection but to demonstrate some ability to outperform the overall composition of the S&P 500 index, by selecting favorable stocks.

4.2 Data Acquisition and Data Cleaning

The process of acquiring and cleaning the data was done simultaneously, ensuring that the data loaded into the SQLite database was clean, consistent, and ready for future analysis. The acquisition process involved three types datasets in the following order: price data, stock metadata, and fundamentals. Each type of data underwent specific cleaning steps to address inconsistencies.

4.2.1 Price Data

The process of acquiring the data began by loading all relevant tickers from the dataset referenced in section 3.1.4, which provided the complete list of stock tickers included in this project. Using these tickers, the OHLC (Open, High, Low, Close) price data was downloaded for each stock. The decision to start with price data stems from its foundational role for many attributes in the feature engineering process. It is also used to define the target, further emphasizing it's importance.

If the price data for a stock was missing or incomplete, that stock was excluded from the project.

As discussed in section 3.3, the adjusted close price is considered the most accurate representation of a stock's value at any given time, as it accounts for stock splits and dividends. To ensure consistency across the dataset, the adjusted close price was used to transform the respective open, high, low, and close prices for each day.

Following this, a validation step was implemented to check for anomalies in the price data. Stocks rarely exhibit extreme daily price fluctuations, and to account for potential outliers, a threshold was defined based on the most extreme historical case of price volatility: the GME short squeeze incident (Forbes, 2024). The threshold was based on GME data, ensuring that the difference in the closing price from one day to the next could not exceed -90%. This step effectively filtered out random, unrealistic, and faulty price spikes or dips in the dataset.

4.2.2 Stock Metadata

After acquiring and processing the price data, the associated metadata for each stock was extracted. This metadata included attributes such as the stock ticker, sector, and other static details. Only a subset of attributes was retained for use in subsequent analyses, as no additional cleaning was required at this stage.

4.2.3 Fundamentals

The final type of data included the fundamental data, originating from earnings calls, balance sheets, income statements, and cash flow statements. These datasets presented two key dates for each entry:

1. **Filing Date**: The date the report was submitted to regulatory authorities.
2. **Effective Date**: The date representing the reporting period's end.

To ensure a realistic and consistent representation of the filing date, a series of cleaning steps were applied. If the filing date was missing or occurred earlier than the effective date — a clear inconsistency — it was adjusted to be exactly two months after the effective date. For cases where the filing date appeared correct, it was shifted forward by one day to simulate the practical availability of the data from external sources.

Sector	P/E Ratio	P/S Ratio	P/B Ratio	Stocks
Technology	66.2 ± 44.1	13.5 ± 5.0	4.2 ± 3.3	103
Healthcare	60.5 ± 57.5	13.6 ± 12.5	4.3 ± 10.8	90
Real Estate	59.9 ± 49.2	17.2 ± 6.6	1.8 ± 12.3	44
Consumer Defensive	57.0 ± 34.9	5.8 ± 6.2	5.0 ± 14.8	46
Consumer Cyclical	53.0 ± 57.7	5.2 ± 9.9	3.2 ± 5.5	107
Industrials	51.3 ± 37.1	5.3 ± 10.9	3.4 ± 10.0	108
Communication Services	49.4 ± 52.0	8.4 ± 19.5	2.4 ± 12.4	42
Other	49.0 ± 40.7	6.1 ± 6.1	2.8 ± 11.7	34
Basic Materials	48.5 ± 51.3	5.2 ± 6.5	2.5 ± 7.4	30
Financial Services	47.0 ± 69.4	9.7 ± 16.6	2.0 ± 3.2	99
Energy	44.3 ± 50.4	7.1 ± 11.7	1.7 ± 13.3	54
Utilities	41.0 ± 34.0	4.5 ± 4.3	1.1 ± 0.8	32

Table 3: Valuation metrics across sectors, showing $\mu \pm \sigma$ for P/E, P/S, and P/B ratios.

This adjustment reflect the typical delay in the release of quarterly reports, which are rarely made public immediately after midnight. The primary rationale behind these transformations was to prevent information leakage, as stock prices often exhibit unusual volatility following the release of quarterly reports, due to institutional investors responding to the new information. These adjustments intend to emulate realistic filing dates, and attempts to negate any advantage obtained from unintentional information leakage, ensuring a more reliable analysis.

4.3 Valuation Metrics

Financial ratios are widely used in fundamental analysis to assess a company's relative value and financial performance. These metrics provide a standardized way to compare companies across industries, allowing investors to make informed decisions. Ratios derived from financial statements such as earnings, revenue, and book value offer insights into a company's profitability, market valuation, and financial health.

For this project, three valuation metrics were used: Price-to-Earnings (P/E) Ratio, Price-to-Sales (P/S) Ratio, and Price-to-Book (P/B) Ratio, each offering unique insights into a company's valuation:

$$\text{P/E Ratio} = \frac{\text{Stock Price}}{\text{Earnings Per Share}} \tag{5}$$

$$\text{P/B Ratio} = \frac{\text{Stock Price}}{\text{Revenue Per Share}} \tag{6}$$

$$\text{P/S Ratio} = \frac{\text{Stock Price}}{\text{Book Value Per Share}} \tag{7}$$

Financial ratios can occasionally have extreme values due to anomalies in the underlying data, such as very low earnings or revenue. To ensure the stability and interpret ability of the metrics, the ratios were clipped to fall within the range [-200, 200]. This decision was informed by data visualization, which revealed that once a metric reaches a certain value, the ratio can be considered "high". Beyond this range it is unlikely that the data would provide meaningful insights. Clipping these ratios ensure that the values can be normalized in the future, without outliers dominating the transformation. There were also instances of missing values, which were simply filled with 0. table 3 shows some statistics of the valuation metrics excluding outliers and missing values (-200, 0 and 200).

4.4 Technical Indicators

Technical indicators are category of tools within financial analysis, derived from a window of price data. Some consider the indicators alone to be able to identify trends, momentum, and potential reversal points in stock prices. However, their predictive capabilities are undoubtedly controversial. Critics argue that technical indicators rely on arbitrary formulas that lack fundamental justification, often appearing more intuitive on charts than providing predictive value. Despite criticisms, technical indicators serve as a method of encoding historical data into a single time frames, which proved effective in aiding deep learning model's predictive capabilities in prior research (Haldan, 2023).

Technical indicators are utilized as features designed to capture historical price dynamics and momentum patterns. When combined with fundamental data and valuation metrics, should provide a diverse perspective on a stocks behavior.

The chosen indicators: AROON, MACD, Bollinger Bands, EMA, ROC, RSI, Max High, and Min Low—are created for two different window sizes (10 and 20 days). This approach ensures that the features capture both immediate market fluctuations and longer-term patterns, potentially enhancing the model's ability to make robust predictions.

In the following subsections, O_t, H_t, L_t, and C_t, will denote the Open, High, Low, and Close values at time t.

4.4.1 AROON

The AROON is an indicator which states the amount time there has been since the previous highest or lowest price in a sequence of the past n candles, as stated on (Investopedia, 2024a)

$$\text{Aroon Up}_{n,t} = \frac{n - \text{Days since high}}{n} \cdot 100$$

$$\text{Aroon Down}_{n,t} = \frac{n - \text{Days since low}}{n} \cdot 100$$

If Aroon_Up = 100, the current price is the highest it has been looking n days back. Likewise, if Aroon_Down = 100, the current price is the lowest it has been looking n days back.

4.4.2 Bollinger Bands

Bollinger Bands consist of a middle band being an n-period simple moving average (SMA), which defines the basis for the upper and lower bands using the standard deviation for n-candles of the close price, which are added or subtracted from the SMA. (Investopedia, 2024b)

$$SMA_{n,t} = \sum_{t-n}^{t} \frac{C_t}{n}$$

$$\text{Upper Bollinger Band}_{n,t} = SMAn,t + (k \cdot \sigma_{n,t})$$

$$\text{Lower Bollinger Band}_{n,t} = SMAn,t - (k \cdot \sigma_{n,t})$$

$$\sigma_{n,t} = \sqrt{\frac{\sum_{i=t-n}^{t} (x_i - \bar{X}_{n,t})^2}{n}}$$

4.4.3 EMA

The Exponential Moving Average (EMA) is a variant of a moving average, which weighs the current price higher, and does infact not have any exponential components in the formula (Investopedia, 2024c).

$$\text{EMA}_{n,t} = (C_t \times \alpha) + (\text{EMA}_{n,t-1} \times (1 - \alpha)) \quad (8)$$

where:

- $\text{EMA}_{n,t-1}$ is the EMA value for the previous day (day $t - 1$).
- α is the smoothing factor, typically given by:

$$\alpha = \frac{2}{n+1} \quad (9)$$

  where n is the number of candles in the span of the EMA.

4.4.4 MACD

The Moving Average Convergence Divergence (MACD), consists of three features. MACD, the Signal Line, and the Histogram. The indicator has shown some correlation to market movements (Chio, 2022).

The MACD is derived from two Exponential Moving Averages (EMAs:

$$\text{MACD} = \text{EMA}_{12} - \text{EMA}_{26} \tag{10}$$

The Signal line is a 9-period EMA of the MACD, is given by:

$$\text{Signal}_t = (\text{MACD}_t \cdot \alpha) + (\text{Signal}_{t-1} \cdot (1-\alpha)) \tag{11}$$

The Histogram is the difference between the MACD and the Signal line, where the functionality of the indicator is when these lines intersect. On the example of 2022's data it appears to be valid for long term investments.

In the code, the parameter n is used as a scaling factor for the previous three mentioned values (12, 26 and 9), such that the values become ($\frac{12\cdot n}{9}$, and $\frac{26\cdot n}{9}$, n)

$$\text{Histogram}_t = \text{MACD}_t - \text{Signal}_t \tag{12}$$

4.4.5 ROC

The Rate of Change (ROC) (Investopedia, 2024d) calculates the percentage difference between the current price and the price n-candles ago. Scaling the value by 100, appears to be a common practice, also done for section 4.4.1 and section 4.4.6.

$$\text{ROC}_{n,t} = \frac{\text{C}_t - \text{C}_{t-n}}{\text{C}_{t-n}} \cdot 100$$

4.4.6 RSI

The Relative Strength Index (RSI) (Investopedia, 2024e) is a momentum oscillator that measures the speed and change of price movements. The RSI oscillates between 0 and 100, where these maximum are seen if the ROC_1 is consecutively negative/positive throughout the entire period.

1. Calculate the gains and losses using close prices. A gain for any given time t is calculated when $C_t - C_{t-1} > 0$ Conversely, a loss occurs when $C_t - C_{t-1} < 0$.

   The Average Gain over n periods is then:

   $$\text{Average Gain}_{n,t} = \frac{1}{n}\sum_{t-n}^{t} \max(C_t - C_{t-1}, 0)$$

   $$\text{Average Loss}_{n,t} = \frac{1}{n}\sum_{t-n}^{t} \min(C_{t-1} - C_t, 0)$$

   where $\max(C_t - C_{t-1}, 0)$ ensures we only consider positive differences for gains.

2. Calculate the RSI for the first n steps:

   $$\text{RSI}_{n,t} = 100 - \frac{100}{1 + \frac{\text{Average Gain}_{n,t}}{\text{Average Loss}_{n,t}}}$$

3. Calculate the RSI for the next steps:

   $$\text{RSI}_{n,t} = 100 - \frac{100}{1 + \frac{\text{Avg Gain}_{n-1,t-1} + \text{Cur Gain}_t}{\text{Avg Loss}_{n-1,t-1} + \text{Cur Loss}_t}}$$

4.4.7 Max_High and Min_Low

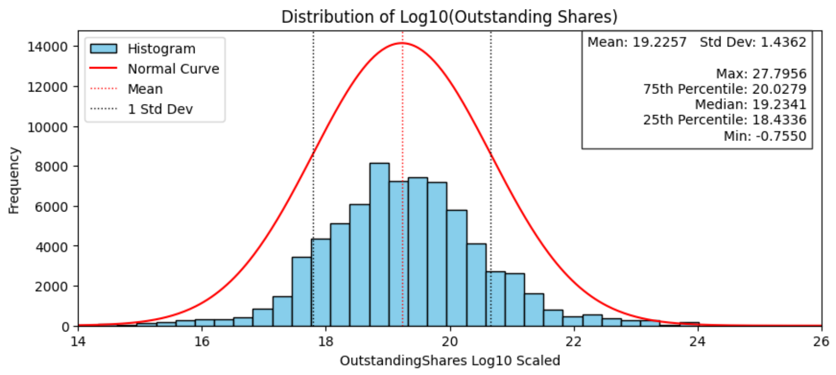
The Max_High and Min_Low are the only indicators which are not strictly technical indicators, but rather caches of maximums and minimums over a period of n rolling back.

$$\text{Max\_High}_{n,t} = \max_{t-n \le k \le t}(\text{High}_k)$$

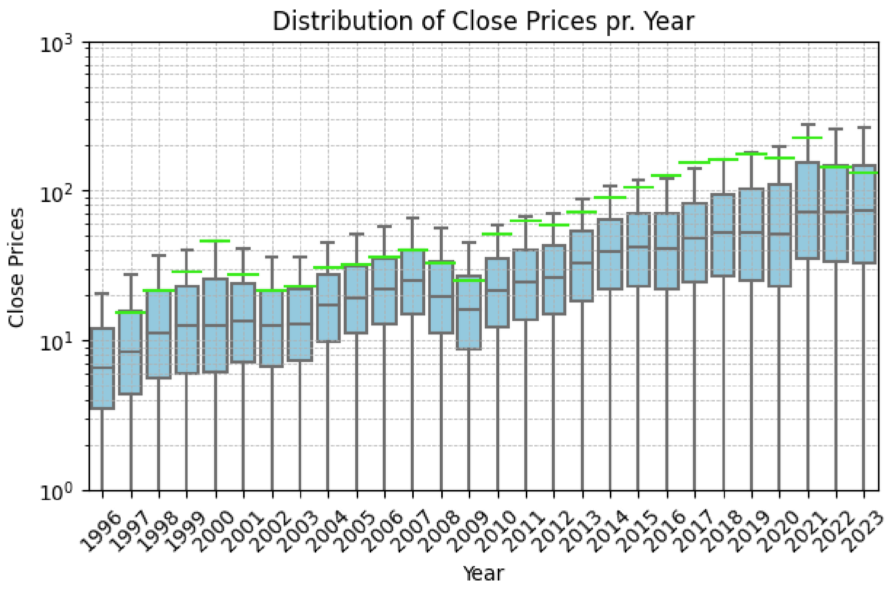
$$\text{Min\_Low}_{n,t} = \min_{t-n \le k \le t}(\text{Low}_k)$$

4.5 Feature engineering

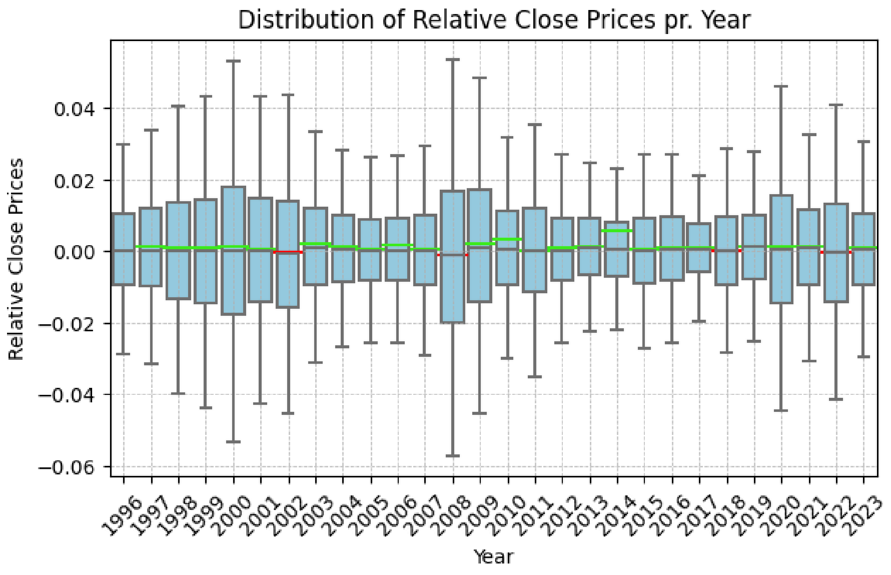
Feature engineering is a critical step in the data preparation process, as the raw data is transformed into interpretable features specific to machine learning, aiming to enhance the performance of models learning process. The feature engineering in this project was guided by an exploratory data analysis (EDA) of the acquired datasets. The primary objective of these data transformations is to capture the information of the stock prices and quarterly reports through various perspectives, emulating the type of information which might be interpreted by an investor looking to buy a new stock. In some instances, the absolute value of a feature is near meaningless, such as the value of the closing price or the EMA section 4.4.3, as they are both relative to previous values for the same



(a) Distribution of outstanding shares



(b) Distribution of close prices (the whiskers define IQR, colored line is mean).



(c) Distribution of close prices after relative transformation (the whiskers define the IQR).

Figure 9: Comparisons of close price distributions before and after relative transformations.

stock. In other instances, both the magnitude and development is interesting, but the range and scale of it often displayed in log10 scales, such as the Total Revenue section 3.4.

Given the diversity of the data, the transformation and normalization processes differed depending on the type of attribute. To maintain clarity and ensure reproducibility, this section is divided into five key subsections, each addressing the unique handling of specific data types:

4.5.1 Price Data normalization

The goal of transforming and normalizing the price data is to convert raw stock prices into features that are interpretable and meaningful for a deep learning model. Stock prices, particularly the adjusted close price, tend to increase over time due to market growth and inflation figure 9b. Moreover, the absolute value of a company is inherently tied to its market capitalization and the number of total shares, making the share price as a predictor irrelevant. figure 9a illustrate the distribution of outstanding shares for all companies included in this analysis.

Investors often focus on a stocks relative price changes over time. For example, an investor evaluating a potential stock for their portfolio would probably look at historical trends, such as consistent growth over several months or years. To replicate this perspective, the adjusted close price was transformed into its relative daily change, calculated as $\frac{\text{Close}_t}{\text{Close}_{t-1}} - 1$, ensuring that the resulting feature has a near-constant mean across the entire time span and is also comparable across different stocks figure 9c.

Following this transformation, the data was standardized in order to scale the range of the data and prepare it for time-series modeling. Given the daily granularity of the data, a rolling normalization approach was applied. For each time step t, the mean and standard deviation were calculated using only the historical data up to that point. This ensures that no future information influences the current normalization, a crucial step in preventing information leakage.

$$\mu_t = \frac{1}{t \cdot N} \sum_{i=1}^{t} \sum_{n=1}^{N} x_{i,n} \tag{13}$$

$$\sigma_t = \sqrt{\frac{1}{N \cdot t} \sum_{i=1}^{t} \sum_{n=1}^{N} (x_{i,n} - \mu_t)^2} \tag{14}$$

Using the derived series of means and standard deviations, each value at time step t for a given stock's attribute $x_{t,n}$ (e.g., the adjusted close price $C_{t,n}$) is standardized accordingly. This iterative time-series normalization process allows the data to maintain its sequential integrity while ensuring consistency across all stocks and time steps Equation 14:

$$x'_{t,n} = \frac{x'_{t,n} - \mu_t}{\sigma_t} \tag{15}$$

4.5.2 Technical Indicators normalization

Technical indicators, derived from price data, provide additional insights into trends, momentum, and volatility. As each indicator's value range differs, they require tailored transformations prior to normalization to ensure consistency and meaningful input for the model. The main objective for this transformation, is to represent each of the technical indicators values, relative to their specific stock, similarly to how the information is provided on a stock chart. With this consideration, indicators used in this project were divided into three categories based on their value ranges, an trailing mean, with specific transformations applied to each.

Indicators with Values Similar to the Closing Price

This category includes indicators such as Bollinger Bands (BB), Exponential Moving Averages (EMA), Max High, and Min Low, which are indicators who's values are are closely tied to the stock's market price,

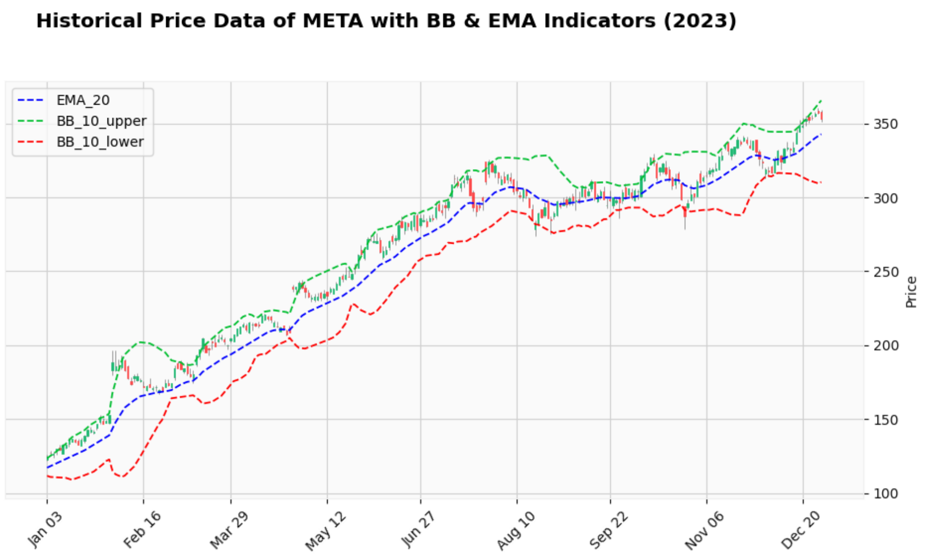


Figure 10: Illustration of EMA, BB, and Price data of META 2023

as seen in figure 10. Similarly to the transformation in section 4.5.1, we wish to express their values relative to the untransformed closing price. If we consider figure 10, we can express any point of interest of the 2 illustrated technical indicators, as the relative distance to the market price Equation 16.

$$x'_t = \frac{x_t - C_t}{C_t} \tag{16}$$

Where x_t is the technical indicator at time t, and C_t is the adjusted (non transformed figure 9b) close price at time t.

Indicators fluctuating around zero without bounds

This category includes the Rate of Change indicator (ROC) and Moving Average Convergence Divergence (MACD). Starting with the ROC, we already have some idea of the distribution, as the formula for ROC stated in section 4.4.5 is very similar to the transformation done in section 4.5.1, making the values for ROC similar across all stocks, as the value is independent of the share price. MACD on the other hand is dependent on the closing price, and requires an aditional transformation prior to normalization.

If we consider figure 11 in section 4.4.4, the specific value of the MACD, the Signal Line, and the Histogram are dependent on the volatility of an asset, and the share price. In order to create a homogeneous representation of the 3 parts of the MACD, we

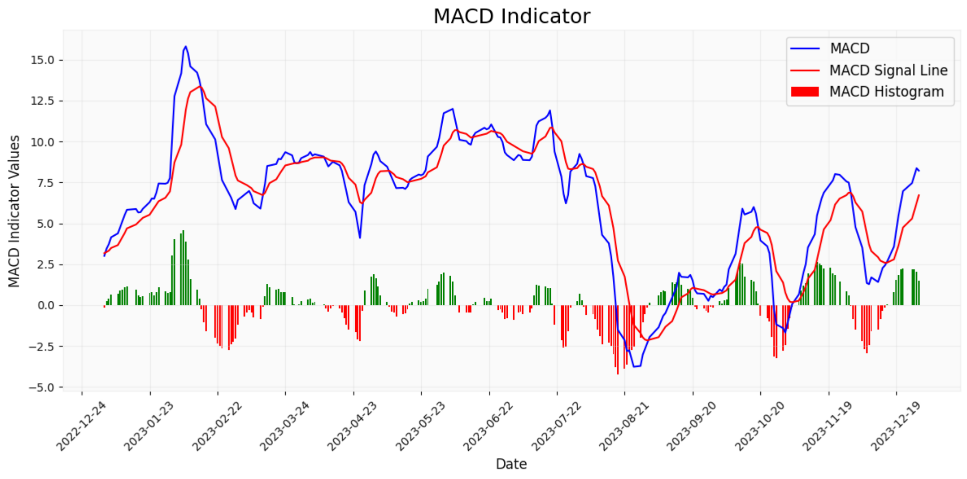


Figure 11: Illustration of MACD indicator of META 2023

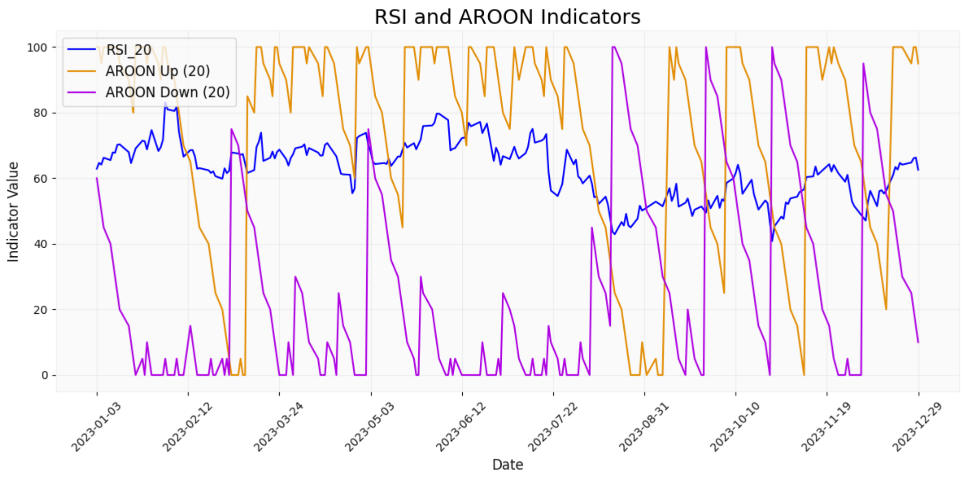


Figure 12: Illustration of RSI and AROON of META 2023

construct the two EMA lines in a price normalized manner, as defined in figure 4.5.2. Following through with the definition of the MACD, Signal line, and Histogram from section 4.4.4, using the price normalized EMA's, yields us a price normalized MACD indicator comparable across all assets.

Indicators fluctuating around Zero with bounds

This category includes indicators such as Relative Strength Index (RSI) and AROON, which are both constrained within specific range from 0 to 100 figure 12. Both the minimum and maximum values are said to be informative for decision making, with the middle value of 50 often not considered interesting. This transformation is rather simple, as we simply scale the values and shift the mean, making the new range between -1 and 1.

$$x'_t = \frac{2 \cdot (x_t - min(x))}{max(x) - min(x)} \tag{17}$$

Time-Series Window Normalization

After applying the category-specific transformations, all indicators (except RSI and AROON) underwent time-series window normalization as described in Equation 14. This process ensures that the mean and standard deviation of each feature are calculated based only on prior values, preserving the temporal structure and avoiding information leakage.

RSI and AROON, which are already bounded and scaled, no additional normalization was applied, as they by definition are already aligned across time and stock, and posses values suitable for deeplearning.

4.5.3 Fundamental data normalization

The fundamental data, reported at a quarterly intervals come with their own unique challenge due to their statistical features. The data is highly exponentially distributed, with an very large range of values. With the presence of both positive and negative values in certain metrics, such as EBITDA and Free Cash Flow, removes the ability to use log transformations. The primary goal for the transformations is to express the magnitude and change of the attributes, while constructing normally distributed data allowing us to normalize it. The below transformations were not applied to the earnings_data as the EPS is already presented as the earnings relative to the amount of outstanding shares.

Through an iterative and exploratory approach, the solution to describe the magnitude of the features was to cube root the data twice, as seen in figure 13, yielding a normal distribution for attributes with non negative values, and two distributions for those containing some negative values. The gap seen in figure 13 stems from the magnitude of the values found in the quarterly reports, ranging from millions to billions of USD.

To capture the temporal dynamics of fundamental data, a new feature was created to express the relative

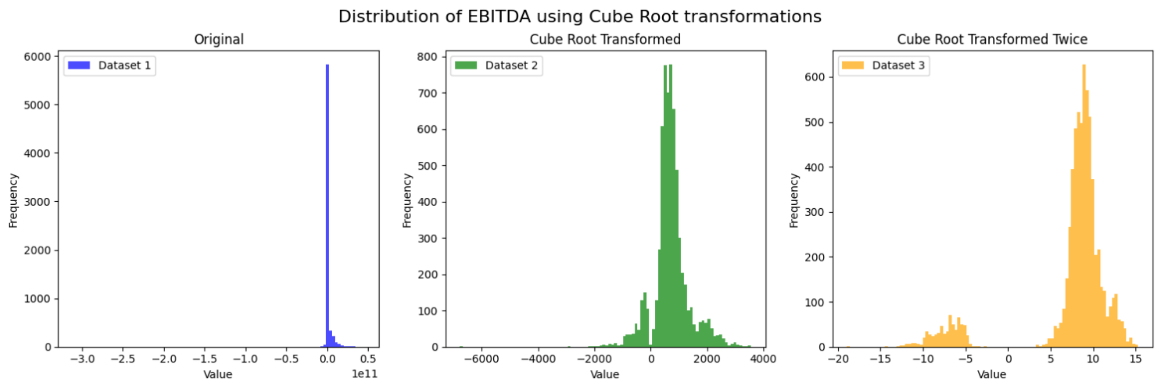


Figure 13: Illustration of cube root transformations on EBITDA

change in each attribute compared to its previous value Equation 18.

$$\Delta x = \frac{x_t - x_{t-1}}{|x_{t-1}| + 1} \tag{18}$$

Once the two features were created, time-series window normalization was employed to normalize the data. The financial values in quarterly reports have increased significantly over the years due to inflation, market growth, and other factors. To combat this, we utilized a 4-year window for the window normalization process, to account for the time differences, and ensuring that each company's values are interpreted relative to its peers within a comparable time frame. This time-series window normalization process was also performed on the Earnings data.

4.5.4 Valuation Metrics normalization

Valuation metrics, such as the Price-to-Earnings (P/E), Price-to-Book (P/B), and Price-to-Sales (P/S) ratios, are defined and discussed in section 4.3, with their statistical characteristics summarized in table 3. Valuation metrics are often used as a reference to justify whether a stocks price is overvalued or undervalued. A high P/E, P/B, or P/S ratio may indicate that a stock is overvalued or that investors have high expectations for the company's future growth. However, high values can also suggest speculative behavior or unsustainable valuations, such as NVIDIA's valuation metrics seen in figure 14.

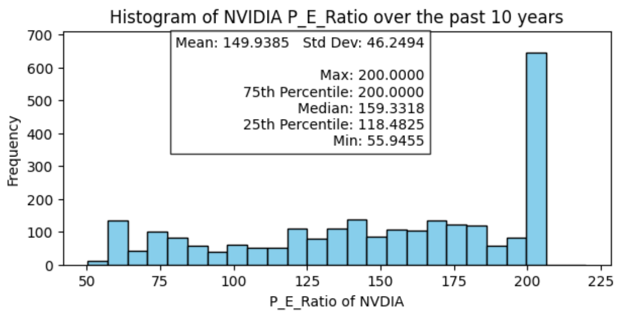


Figure 14: NVIDIA P/E Ratio between 2014 and 2024

Conversely, low values can imply that a stock is undervalued, which might indicate a potential investment opportunity. Persistently low values could also signal issues with the company, such as declining revenue, earnings, or reputation. Low values can also signal lack of interest or a consensus of no growth potential.

To address the challenge of variability and better align the metrics with intuitive interpretations, the decision was made to transform each valuation metric into its reciprocal form $\frac{1}{Ratio}$. By doing so, a higher value represent potentially undervalued companies.

After applying the reciprocal transformation, the valuation metrics were normalized using the same method as described for price data in Equation 15.

4.5.5 One-Hot Encoding Sectors and Active State

Financial norms and practices vary widely between sectors, as illustrated in the valuation metrics table table 3. Quarterly reports are also sector specific.

An example being technology companies, often prioritizing growth over profitability in their early stages. This is reflected in their tendency to minimize profits by reinvesting heavily in research and development (R&D) or other growth initiatives. For instance, a profitable tech company might allocate most of its earnings to R&D, infrastructure, or marketing, showing relatively low profits despite significant revenue growth. This strategy allows tech firms to scale rapidly, but it also means that traditional metrics like P/E ratios might be less informative for these companies.

Another example are companies which work with financial services, such as banks. Their quarterly reports differ significantly from other sectors due to their focus on cash flow and interest-related income. This leads to banks often having unusually high cash flow values due to the nature of their business (e.g., loans, deposits, and trading activities). Their income statements also differ, as it is primarily derived from interest, fees, and revenue from realized investments, which makes their income statement structure unique compared to companies in manufacturing or retail.

To incorporate the sector information into the model, a one-hot encoding approach was implemented where each sector was represented by a binary feature, where a value of 1 indicates that a stock belongs to the sector, and 0 indicates otherwise.

In addition to sector encoding, another key feature introduced was the Active State. The dataset includes a total of 1092 stocks, but not all of them are active throughout the dataset's entire duration. To represent this, a feature dubbed ”Active” was created, with values -1 and 1, defining whether the stock is listed on the S&P 500 at the specific time.

4.6 The Feature Data Set

The final state of the tabular dataset contains 71 features and 5.3 million entries. The dataframe is stored in a SQL database under the name ML_Feature_Data. This table serves as the foundation for node attributes in the graph and contains all previously defined features for each stock (node) for every trading day in the dataset. All features are stored in a standardized format, ensuring consistency and compatibility for downstream processing. To obtain a better understanding of the value range for each feature, the violin plot below illustrates said distributions after clipping outliers beyond 5% on each end of the extremes figure 15.

4.7 Defining the target

The target creation process is a critical component of the methodology, as it numerically defines the problem that the model seeks to solve. In this project, the ultimate objective of the model is to select attractive stocks based on their current features, and construct a portfolio where the average returns of the selected assets increase. This aligns with the fundamental goal of portfolio management: to maximize returns while managing risks. In an ideal, and purely hypothetical scenario, the model would predict the future returns of all of the assets in the dataset at any given time, for any distance in the future. This is however this is an impossible feat, as even predicting future price changes is notoriously challenging due to the chaotic and stochastic nature of the stock market as described by (Cao et al., 2019).

Given the inherent difficulty of accurately predicting future price changes, an ordinal categorical approach was adopted for the target creation. Specifically, an attribute denoting the percentage change over a 5-day future window was used to categorize stocks into five ordinal classes:

1. 2: Significant positive change (e.g., best investments).
2. 1: Moderate positive change.
3. 0: Minimal or no change / Inactive Stock
4. -1: Moderate negative change.
5. -2: Significant negative change (e.g., worst investments).

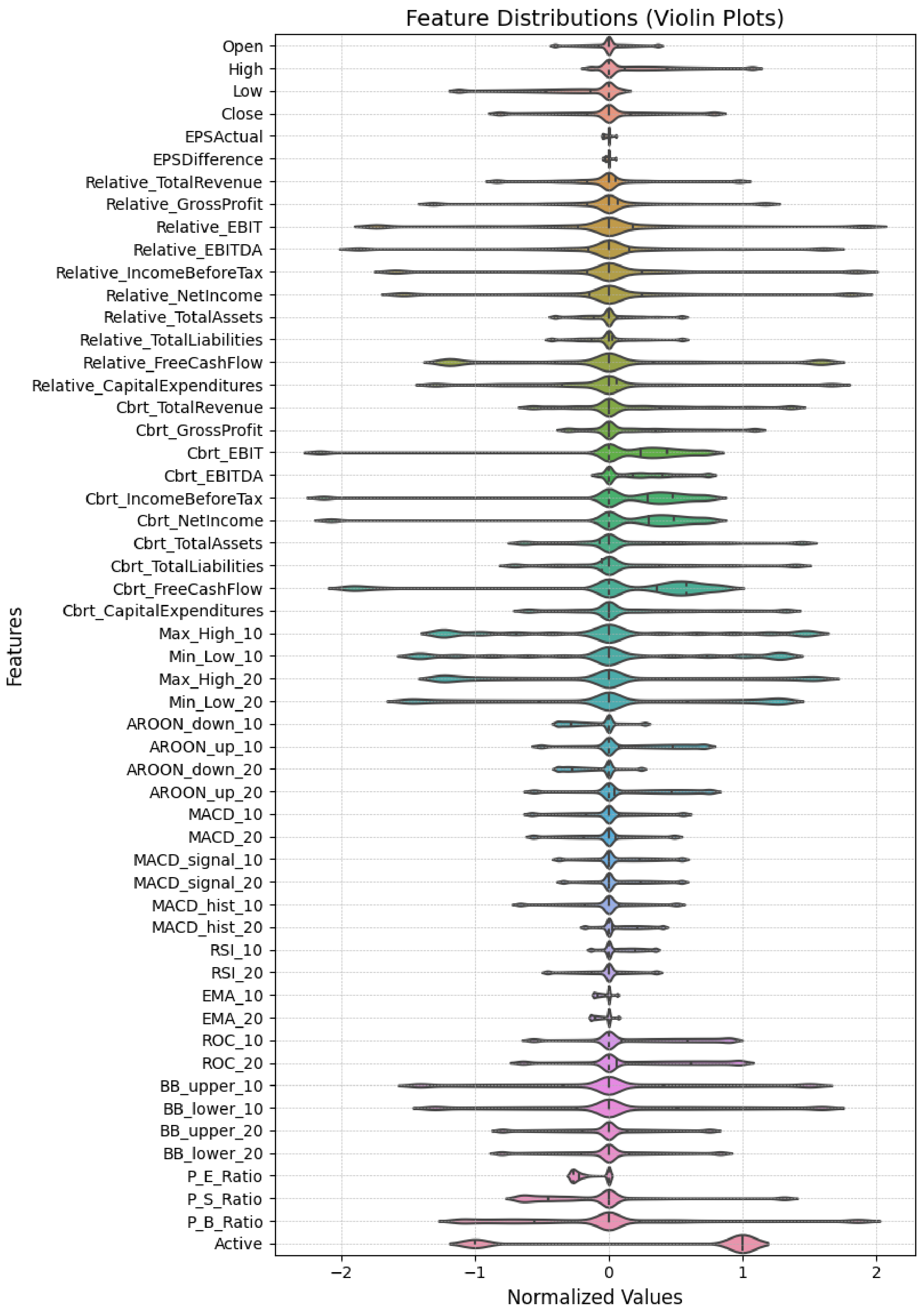


Figure 15: Distribution of features in the final dataset

Label	Bin	**Percentage (%)**	**Frequency**
2	$\mu + (\sigma \cdot 0.7) \, , \infty$	14.44 %	602,136
1	$\mu + (\sigma \cdot 0.2) \, , \mu + (\sigma \cdot 0.7)$	23.14 %	964,451
0	$\mu - (\sigma \cdot 0.2) \, , \mu + (\sigma \cdot 0.2)$	30.08 %	1,252,968
-1	$\mu - (\sigma \cdot 0.7) \, , \mu - (\sigma \cdot 0.2)$	23.64 %	984,882
-2	$-\infty \, , \mu - (\sigma \cdot 0.7)$	15.19 %	633,074

Table 4: Frequency and percentage distribution of target classes in the dataset for active stocks.

There are multiple reasons arguing for the benefits of using predefined bins for the targets:

1. **Evaluation Flexibility** The ordinal nature of the labels allow us to treat them as regression values during training, enabling the model to regress the attractiveness of an asset on a continuous scale. Conversely, for evaluation, the predictions can be converted back into class labels, facilitating the investigation of performance across different types of movements.

2. **Distribution Design** To reflect market dynamics, the target labels were designed with a desired distribution of 13%, 22%, 30%, 22%, and 13% for the classes (-2, -1, 0, 1, 2), respectively. This ensures that extreme classes (great and horrible investments) are relatively scarce, reflecting real-world conditions where most stocks exhibit minimal or moderate changes in value.

3. **Backtesting and Portfolio Construction** The regression-like predictions can be directly utilized for portfolio optimization. By sorting stocks based on their predicted values, the model can rank investments and assign weights proportional to the predicted attractiveness.

The specific bins levels, proportions, and frequency are shown in table 4 defined by over the entire dataset, and only represent the labels of active stocks.

4.8 Graph Structures

The graph structures in this project serves to unveil the benefits of representing the relationships and properties of stocks over time. Each graph captures both the individual attributes of stocks (nodes) and their dynamic interactions (edges) within the market, creating a comprehensive data representation of the stock market. This approach transforms tabular stock market data into a graph-based structure, enabling the use of Graph Neural Networks (GNNs), which predictions are used for the backtester.

Node data represents a specific stock and is enriched with a combination of daily and quarterly attributes. The edge data encodes relationships between stocks using price correlations calculated over a sliding time window, emphasizing strong connections while allowing the network to adapt to evolving market dynamics.

Daily graphs are then constructed, and are further organized into sequential ”chunks” to incorporate temporal information. Chunks in this project varied in size, between 3 and 10 sequential graphs, hopefully enabling the AT-GCN to model short-term market trends and inter-stock relationships.

The following section outlines the detailed process of defining the graph structures, including the creation of node and edge data, the construction of daily graphs.

4.8.1 Nodes

The nodes in the graph represent individual stocks, each with a set of features derived from daily price data and quarterly financial reports. The preparation of node data involves several steps to ensure consistency and relevance across the dataset, and was performed iteratively for each specific stock:

A base tabular dataset for a specific stock is constructed with a 1-day granularity, representing each trading day within the dataset’s time span. An auxiliary dataset containing all trading dates was used as the basis for the dates, ensuring no gaps for bankrupt companies, or relatively young companies.

Next, data quarterly financial report is merged with the daily dataset, through a left merge, using the filing date as the date key, as the filing date represents the date of which the report was available, only providing the information available up to the specified date, preventing data leakage and aligning with real-world scenarios.

The temporal resolution of the price data and the quarterly reports differ, hence, some data-processing was required. The fundamental data was filled in a forward manner, and the early entries with missing values were filled with 0's. The resulting dataset contains features for each day, regardless whether the stock existed at the specific point in time.

4.8.2 Edges

The edges in the graph are intended to represent the relationships between stocks, specifically their price correlations over a relatively short time window of 43 entries (roughly 2 months). This period was selected based on the observation in (Soleymani and Paquet, 2021) that correlations are more effective in a short time frame. Specifically stated in (Soleymani and Paquet, 2021): "This period should be relatively short as the correlation tends to disappear rapidly due to randomization".

Due to the size of the dataset and the computational overhead of processing fully connected graphs, the number of edges per node is deliberately limited, subsequently ensures that the edges included are the most meaningful for the GNN. The edge prioritization and inclusion process is as follows:

1. **Ordering of Edge Weights** The computed correlations for each day (both positive and negative) are sorted in descending order of absolute value. This prioritization ensures that the strongest relationships, whether positive or negative, are considered first, when adding the edges to the graph. The potential edge pairs, and their weight, were iteratively consumed to define edges during the

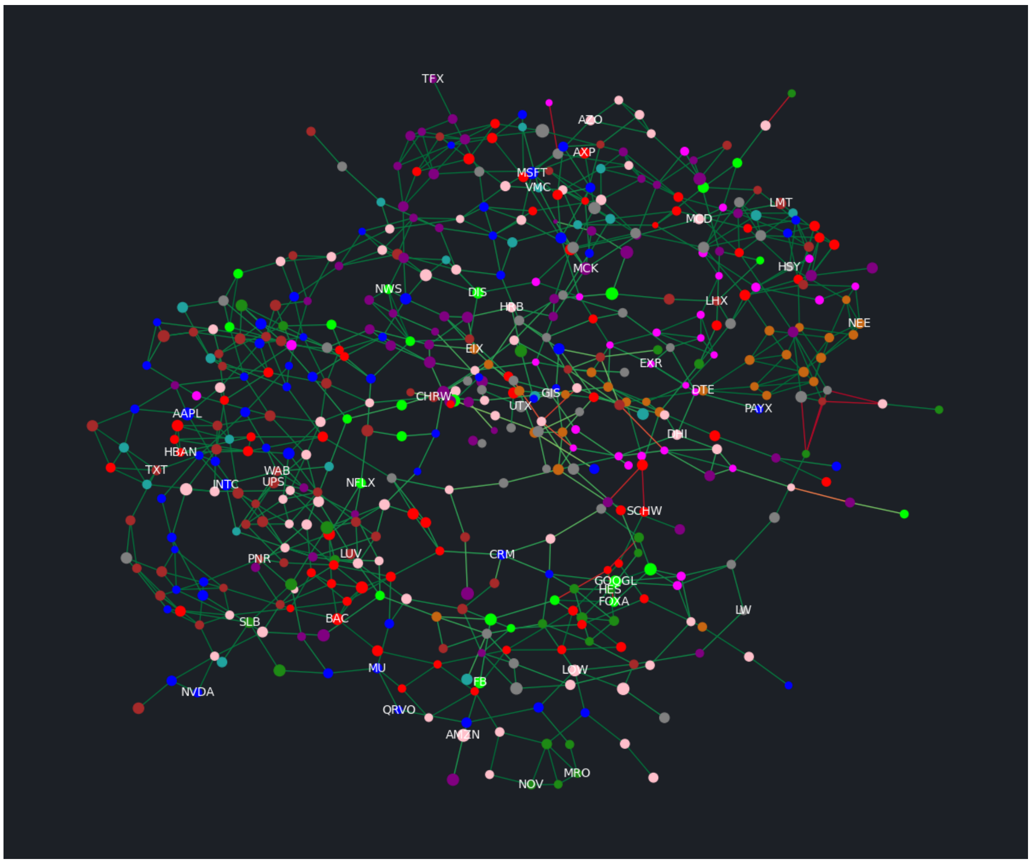


Figure 16: Sample graph of data from 2019-07-01

next steps:

2. **Limiting Node Degree** An edge is added between two nodes only if the node degree of both stocks is below a predefined maximum of 5 edges per node. Adding this constraint limited the connectivity of each stock, ensuring sparsity while retaining the most relevant relationships.

3. **Transformation** As the AT-GCN is unable to handle negative edge weights, some transformations were required for the negative correlations. To handle this, the values were shifted to range from 0 to 1.

The main two reason why it was considered appropriate to limit the connectivity, was due to the computational overhead associated, along with ambitions of providing the most valuable information for the GNN based model.

4.8.3 Final Graph format

The final graph format is designed to encapsulate the temporal and structural dynamics of the stock market, preparing the data to be utilized for the AT-GCN. This involved constructing individual daily graphs figure 16, and organizing them into sequential chunks to provide a window of 3,5, and 10 days.

The visualization figure 16 shows a graph describing the data from 2019-07-01. The size of the node represents the squished Revenue, the color represents the sector, and the edges represent positive (green) correlations, and negative (red) correlations.

The final summary statistics can be seen in table 5. The graph size of 734 nodes is a is a noticeable reduction from the initial dataset of 1092 ticker, which is the products of the cleaning processes outlined in section 3.2.

Statistic	**Value**
Number of Daily Graphs	7,109
Number of Graph Chunks	7,105
Tickers Included per Graph	734
Node Attributes per Graph	71
Mean Edge Count per Graph	1,745
Standard Deviation of Edge Count	198

Table 5: Summary of the final graph format and dataset statistics.

4.9 Train, Validation, Test split

The dataset is divided into train, validation, and test sets following a 70/10/20 split, a widely accepted practice in data science. This split ensures a variety of scenarios within the stock market during the training and testing period. The validation set is during a relatively calm period in terms of the financial market. Given the temporal structure of the data, the split is performed sequentially to maintain the chronological order of events. This approach prevents information leakage by ensuring that future market data does not inadvertently influence the training process. The periods for each split are as follows:

Train Period:
Start Date: 1996-04-08
End Date: 2016-01-07
The training set spans two major stock market crashes, the dot-com bubble and the 2008 housing crisis, aiming to provide the model some exposure towards challenging market conditions.

Validation Period:
Start Date: 2016-01-08
End Date: 2018-10-31
The validation set does not include any significant market crashes, offering a relatively stable period for fitting the model.

Test Period:
Start Date: 2018-11-01
End Date: 2024-06-28
The test set includes the COVID-19 crash, and the tech-scare of 2022, presenting a unique opportunity to evaluate the model's performance under sudden and extreme market downturn.

4.10 Model Training

The AT-GCN pytorch geometric temporal and LSTM pytorch LSTM models were imported from PyTorch, leveraging prebuilt model objects that were configured with a specific output layer to align the model outputs with the defined target labels.

Both models were trained using identical datasets as described in section 4.9, and follow consistent evaluation protocols to enable a fair comparison of their predictive capabilities. The LSTM's outputs were transformed to match the GNN's node-level prediction format, enabling a direct evaluation of their performance on the same targets for any given date.

The training pipeline was an iterative process of defining model parameters, followed by training and evaluating the models. A comprehensive set of evaluation metrics were employed to assess model performance, including:

1. Prediction Statistics: Mean, standard deviation, minimum, and maximum of the predictions, to compare the distribution of the predictions to the target data.

2. Loss Metrics: Training, validation and Test Mean squared error were evaluated to asses whether to models were generalizing.

3. Confusion Matrix: Initially designed to uncover the accuracy and precision of predicted labels 1 and 2, lead to the discovery of prediction values being relatively small compared to the target labels.

4. Normalized Confusion Matrix: Accounting for the scaling difference in the target distribution the prediction distribution, the normalized confusion matrix categorizes predictions into the five defined labels (-2, -1, 0, 1, 2).

5. Backtesting Metrics: A custom built framework to assess the predictively capiabilities on the validation and test data. Specifics will be elaborate in section 4.11

For this particular task, one metric holds particular significance: the precision of labels 1 and 2. Precision, defined in Equation 19...

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{19}$$

...serves as a direct measure of how accurately the model identifies attractive investment opportunities. The emphasis on precision, rather than recall, aligns with the ultimate goal of the model: constructing a portfolio of high-performing assets. In this context, even if only a fraction of the stocks in the S&P 500 are deemed attractive at a given time, the model only needs to correctly identify a subset of these opportunities to form a successful portfolio.

4.11 Backtesting

Backtesting is a critical tool in financial modeling that simulates the performance of a strategy, portfolio, or model against historical data. It allows researchers and practitioners to assess the effectiveness of their approach in a controlled environment before deploying it in real-world scenarios. By evaluating cumulative returns, annualized returns, and Sharpe ratios, backtesting provides a quantitative framework to compare strategies and determine their potential profitability and risk against a desired benchmark, such as the US interest rates, or the ”Buy the S&P 500” strategy.

In the context of this project, backtesting serves as the most realistic measure of the models' capabilities in portfolio management. Machine learning metrics like Mean Squared Error (MSE) and confusion matrices offer insights into a model's predictive accuracy, but fall short of reflecting the models applicability. Combining simulations, strategies, and models, the possibility for optimizing the model's predictions and how they are interpreted, is similar to that of machine learning models.

Key metrics used in the backtesting framework:

1. **Cumulative Returns**: A measure of the total growth of an investment over time.

2. **Annualized Returns**: The geometric average of

yearly returns, reflecting the consistent growth rate of an investment.

3. **Sharpe Ratio**: A risk-adjusted return metric that evaluates how much excess return is achieved for each unit of risk taken.

4. **Maximum Drawdown**: The largest peak-to-trough decline in portfolio value during the testing period, expressed as a percentage of the peak value. It measures the worst observed loss from a historical high, providing insight into the portfolio's downside risk.

4.11.1 Backtester Implementation

The backtester is designed to evaluate the practical utility of the predictions made by the AT-GCN and LSTM models. It operates by simulating the performance of a portfolio based on the model's predictions using the validation and test data, applying a monthly rebalancing of the portfolios composition.

The backtester begins by receiving two primary inputs: Predictions, and Price Data. The outputs of both the AT-GCN and LSTM model are originally structured as a tensor corresponding to the batch size, the window size, the amount of nodes. During the final steps of training, each deep learning model store the validation and test predictions in an easily applicable data structure, for the backtester.

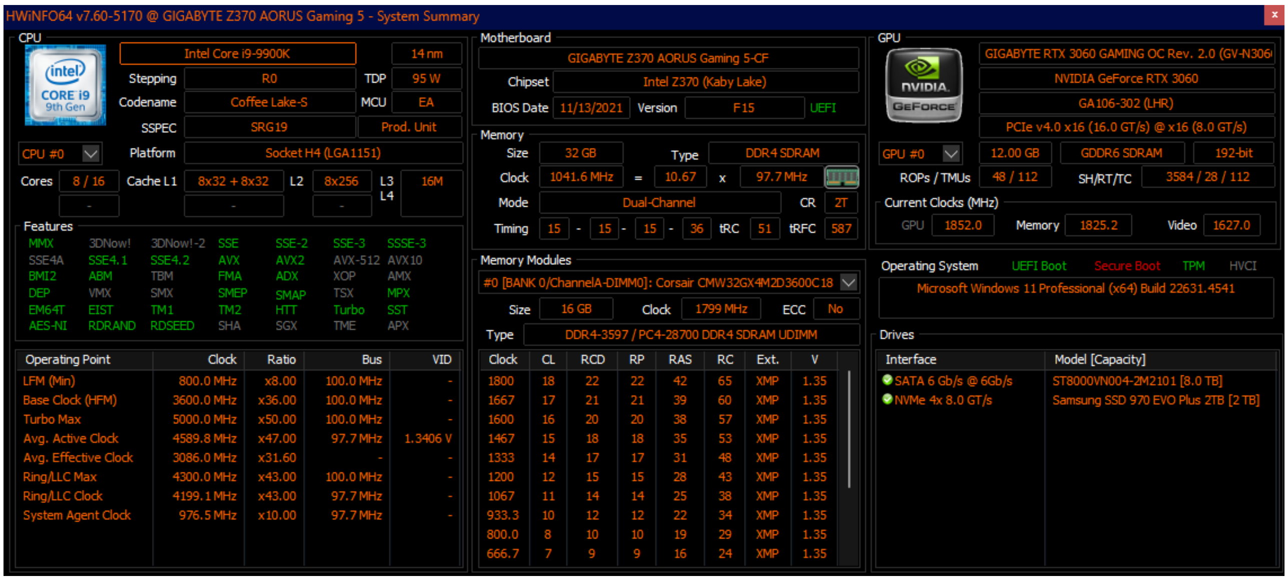
The predictions from the test set are used to create a portfolio data-structure, where the proportion invested in a given asset corresponds to the assets perceived value for the given date. As the distribution of the models' predictions vary, the weights associated are scaled relative to said distributions. There is also a maximum weight pr. stock, ensuring one asset in a specific portfolio does not dominate the portfolio.

Algorithm 1: Backtester Algorithm

```
   Data: Predictions $P$, Price Data $D$, Initial Cash $C_0$
 1 Attributes: Score Vectors $S$, Stock and Score $s$, Portfolio $\varphi$, Portfolio Value $V$, Number of stocks selected $N$
   Result: Performance Metrics: Cumulative Returns, Annualized Returns, Sharpe Ratio, Maximum Drawdown
 2 Initialize: Portfolio state $S \leftarrow \{\text{cash: } C_0, \text{weights: } \emptyset\}$,
 3 Group trading dates into monthly intervals $\mathcal{M}_D$;
 4 foreach Month $M_t \in \mathcal{M}_D$ do
 5     Calculate Score_Vector for prior 2 weeks of $M_t$;
 6     $S_t = \sum_{i=1}^{10} P_(t, i) \cdot \frac{i}{10}$
       $\varphi_t = \text{Top}_N (S_t)$
       foreach Trading Day $d \in M_t$ do
 7         Calculate daily returns $R_d$;
 8         $R_d = \sum_{s \in \varphi_t} w(s) \cdot \Delta P_d(s)$
           Update portfolio value $V_d = V_{d-1} \cdot (1 + R_d)$;
 9     end
10     Store portfolio history and net value;
11 end
```

Algorithm 1 represents the core flow of the algorithm, however the following text includes specific details of the implementation which is not included in the algorithm.

Once a month, the portfolio selects the 20 best stocks at the selected period. The event of which the portfolio changes using the models predictions is called rebalancing. During this event, stocks are bought and sold, and to illustrate a realistic environment, each acquisition and dispositions will



HWiNFO64 v7.60-5170 @ GIGABYTE Z370 AORUS Gaming 5 - System Summary

CPU

	Intel Core i9-9900K		14 nm
Stepping	R0	TDP	95 W
Codename	Coffee Lake-S	MCU	EA
SSPEC	SRG19		Prod. Unit
CPU #0	Platform	Socket H4 (LGA1151)	

Cores	8 / 16	Cache L1	8x32 + 8x32	L2	8x256	L3	16M
	-		-		-	L4	

Features

MMX	3DNow!	3DNow!-2	SSE	SSE-2	SSE-3	SSSE-3
SSE4A	SSE4.1	SSE4.2	AVX	AVX2	AVX-512	AVX10
BMI2	ABM	TBM	FMA	ADX	XOP	AMX
DEP	VMX	SMX	SMEP	SMAP	TSX	MPX
EM64T	EIST	TM1	TM2	HTT	Turbo	SST
AES-NI	RDRAND	RDSEED	SHA	SGX	TME	APX

Operating Point	Clock	Ratio	Bus	VID
LFM (Min)	800.0 MHz	x8.00	100.0 MHz	-
Base Clock (HFM)	3600.0 MHz	x36.00	100.0 MHz	-
Turbo Max	5000.0 MHz	x50.00	100.0 MHz	-
Avg. Active Clock	4589.8 MHz	x47.00	97.7 MHz	1.3406 V
Avg. Effective Clock	3086.0 MHz	x31.60	-	-
Ring/LLC Max	4300.0 MHz	x43.00	100.0 MHz	-
Ring/LLC Clock	4199.1 MHz	x43.00	97.7 MHz	-
System Agent Clock	976.5 MHz	x10.00	97.7 MHz	-

Motherboard

	GIGABYTE Z370 AORUS Gaming 5-CF			
Chipset	Intel Z370 (Kaby Lake)			
BIOS Date	11/13/2021	Version	F15	UEFI

Memory

Size	32 GB	Type	DDR4 SDRAM				
Clock	1041.6 MHz	=	10.67	x	97.7 MHz		
Mode	Dual-Channel	CR	2T				
Timing	15 - 15 - 15 - 36	tRC	51	tRFC	587		

Memory Modules

#0 [BANK 0/ChannelA-DIMM0]: Corsair CMW32GX4M2D3600C18

Size	16 GB	Clock	1799 MHz	ECC	No
Type	DDR4-3597 / PC4-28700 DDR4 SDRAM UDIMM				

Clock	CL	RCD	RP	RAS	RC	Ext.	V
1800	18	22	22	42	65	XMP	1.35
1667	17	21	21	39	60	XMP	1.35
1600	16	20	20	38	57	XMP	1.35
1467	15	18	18	35	53	XMP	1.35
1333	14	17	17	31	48	XMP	1.35
1200	12	15	15	28	43	XMP	1.35
1067	11	14	14	25	38	XMP	1.35
933.3	10	12	12	22	34	XMP	1.35
800.0	8	10	10	19	29	XMP	1.35
666.7	7	9	9	16	24	XMP	1.35

GPU

GIGABYTE RTX 3060 GAMING OC Rev. 2.0 (GV-N306

NVIDIA GeForce RTX 3060

GA106-302 (LHR)

PCIe v4.0 x16 (16.0 GT/s) @ x16 (8.0 GT/s)

GPU #0	12.00 GB	GDDR6 SDRAM	192-bit
ROPs / TMUs	48 / 112	SH/RT/TC	3584 / 28 / 112

Current Clocks (MHz)

GPU	1852.0	Memory	1825.2	Video	1627.0

Operating System UEFI Boot Secure Boot TPM HVCI

Microsoft Windows 11 Professional (x64) Build 22631.4541

Drives

Interface	Model [Capacity]
SATA 6 Gb/s @ 6Gb/s	ST8000VN004-2M2101 [8.0 TB]
NVMe 4x 8.0 GT/s	Samsung SSD 970 EVO Plus 2TB [2 TB]

Figure 17: Hardware used for running the project

have cost of trade fees associated with it. While this value varies from broker to broker, this project will use the reported trading fees from one of the largest Copenhagen based stock trading brokers; SAXO BANK, at **0.08% per. trade** (Bank, 2024) of the value of the investment. The backtester allows the portfolio to hold cash, which occurs if the sum of the most attractive predictions for the specific date are lower than usual. Stocks are ranked according to the predictions for the prior 2 weeks, such that each asset has a score Score_Vector $= \sum_{i=1}^{10} \hat{Y}_i \frac{i}{10}$, where the 10'th iteration corresponds to the day of the reblancing. A subset of predicted top-performing stocks is selected for the portfolio. Portfolio weights are allocated proportionally based on the model's predicted values or evenly across selected stocks. The price data serves as the foundation for calculating the daily returns and prior mentioned portfolio performance metrics.

The backtester evaluates portfolio performance using the previously defined metrics in section 4.11.1.

5 Results

5.1 Hardware

The hardware used in this project consists of a RTX 3060 (12GB VRAM), and Intel Core i9-9900k 8 cores - 16 threads, and 32GB of DDR4 RAM. A table with further details is included for the curious readers. figure 17.

5.2 Hyperparameters of Best Models

This section summarizes the configurations for the top models of both the AT-GCN and LSTM architectures, allowing us to observe the trade-offs between training time, complexity, and predictive accuracy.

table 6 presents the key hyperparameters, training configurations, and metrics for the three best-performing AT-GCN models (A3TGCN2: 5, 6, 7) and LSTM models (LSTM: 2, 3, 4). The models were tuned iteratively, with small adjustments being made to the hyperparameters to observe differences. Prior models are not included in the results to maintain readability.

Learning Rate and Optimization:

All models utilized the Adam optimizer, and the range for the optimal learning rates were obtained through some initial trial and error. The models use learning rates between $10^5 - 10^7$. Additionally, a

Table 6: Results for Selected Models: A3TGCN2 and LSTM

Parameter	**A3TGCN2_5**	**A3TGCN2_6**	**A3TGCN2_7**	**LSTM_2**	**LSTM_3**	**LSTM_4**
Hidden Dim	512	512	512	64	64	128
Periods	5	5	5	5	5	5
Edge Weights	Normal	Normal	Self-loops only	N/A	N/A	N/A
Batch Size	1	1	1	32	32	16
Optimizer	Adam	Adam	Adam	Adam	Adam	Adam
Learning Rate	10^6	10^5	10^5	10^6	10^6	10^7
Forced Epochs	50	100	10	15	5	10
Epochs	59	119	13	22	12	39
Patience	10	20	5	4	4	4
Num Layers	N/A	N/A	N/A	2	2	3
Dropout	0	0	0	0.2	0.2	0.5
Gamma (LR Decay)	0	0	0	0.9	0.9	0.9
Train Time	8:51:00	19:52:00	2:34:00	5:03:29	5:44:59	22:22:12
Loss Function	Weighted MSE	Weighted MSE	Weighted MSE	Weighted MSE	Weighted MSE	Weighted MSE
Loss Function Weights	Weights 1	Weights 1	Weights 1	Weights 2	Weights 3	Weights 1
Train Loss	4.3337	4.2749	4.3174	6.9677	6.9104	6.0312
Validation Loss	3.7280	3.7603	3.7337	5.0634	4.8818	4.5863
Test Loss	4.7993	4.8265	4.7905	8.8369	8.8253	7.7557
Validation Unweighted Loss	1.2113	1.2405	1.2198	1.8250	1.8263	1.8750
Test Unweighted Loss	1.4871	1.5258	1.4868	2.7099	2.7095	2.7649
Mean Test Pred	0.1948	0.2348	0.1901	0.0067	0.0042	0.0007
Std Test Pred	0.1158	0.2278	0.1412	0.0523	0.0534	0.0259

learning rate decay was applied for the LSTM models, attempting to help with long-term convergence.

Epochs and Patience:

Both models experienced the same issues, with convergence happening much earlier than expected. Because of this, the patience was set relatively high, and there was introduced a hyper parameter called ”forced epochs”, which would force the model to train for N amount of epochs. The most extreme case of application, A3TGCN2_6, trained for 100 epochs before patience was considered. What can be observed on is that the validation loss did not improve much, and the training loss only improved slightly. The LSTM models took much longer time to train pr. epoch, hence the patience and forced epochs were reduced.

Dropout

Dropout rates of 0.2–0.5 were critical for preventing overfitting in the LSTM models, however, the AT-GCN models had no direct way of implementing dropout, except if it were to be on the input layer or the output layer.

Weighted MSE Loss Function:

Both model types utilized a weighted Mean Squared Error (MSE) loss function, with higher penalties for extreme labels (-2 and 2). This weighting was an attempt to improved predictions for critical classes, aligning with the models task. Specific weights can be seen here table 7.

Table 7: Loss Function Weights

Weights Set	-2	-1	0	1	2
Weights 1	3.0	2.0	0.5	2.0	5.0
Weights 2	4.0	2.0	1.0	2.0	6.0
Weights 3	2.0	0.75	0.25	3.0	8.0

Low Standard Deviation

Both models appeared to have a relatively low standard deviation of their predictions. Ideally, the variance of the predictions were expected to match that of the target, especially with the weighted loss seen in table 7. The standard deviation of the target in the test set was 0.9208, and consists of values $\in \{-2, -1, 0, 1, 2\}$. The AT-GCN models ranged between 0.1158 to 0.2278, while the LSTM modes ranged between 0.0259 to 0.0523, significantly lower than the targets distribution.

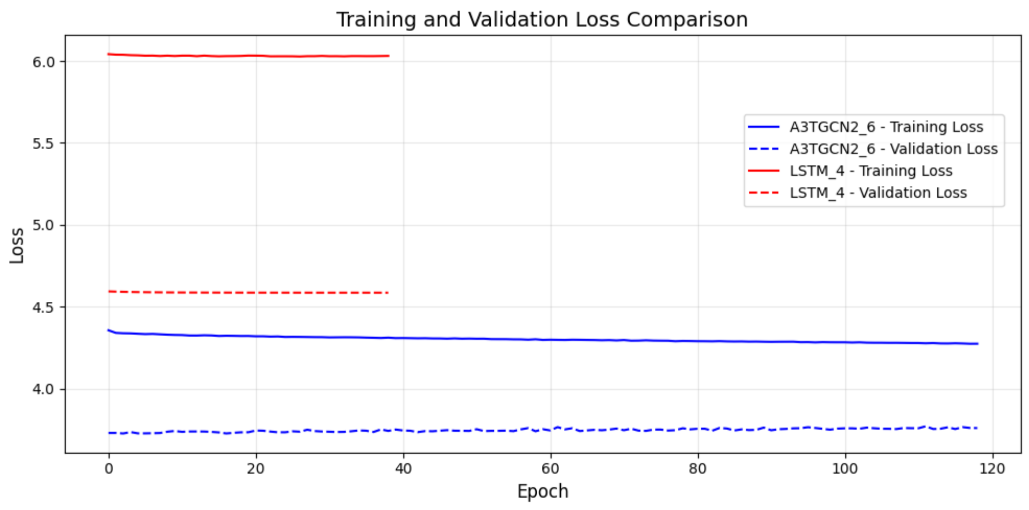


Figure 18: Best Training losses of the best performing LSTM and A3TGCN models trained

Performance:

Validation and test losses were consistently better on the AT-GCN models. However, removing the edge weights completely, as done with A3TGCN2_7, did not appear to influence the unweighted loss score. This indicates that the correlation edge weights were perhaps not as informative features as anticipated.

5.3 Machine Learning Metrics

Evaluating the machine learning models in this project begins with traditional metrics that assess their predictive accuracy on the validation and test datasets. The AT-GCN and LSTM models are compared using training and validation loss curves, confusion matrices, and precision. These metrics provide insight into how well the models predicts the target labels, specifying their performance for practical application in portfolio management. The following subsections detail the results and analysis of these evaluations.

5.3.1 Validation and Training Loss Curves

Typically, deep learning model's training and validation improve for each epochs, until the validation score converges at some point, typically dictating the point of which the model has reached its maximum potential using the training data. With the dataset used in this project, the validation loss never decreased significantly, as it would typically converge within the first epoch, indicating that the data provided for the model failed to grant the model the ability to generalize figure 18.

5.3.2 Confusion Matrices

Due to the low standard deviations of the predictions, the unscaled confusion matrices are uninformative, as most of the models have a low prediction variance, leading to a confusion matrix with all predictions at label 0.

Table 8: Precision Metrics by Class for Each Model

Class	AT3-GCN_6	LSTM_4
-2	0.1448	0.1026
-1	0.2148	0.2270
0	0.3738	0.7969
1	0.2118	0.2242
2	0.2102	0.2247

Table 9: Recall Metrics by Class for Each Model

Class	AT3-GCN_6	LSTM_4
-2	0.0851	0.1227
-1	0.2696	0.2852
0	0.4872	0.7984
1	0.1601	0.1844
2	0.1265	0.1436

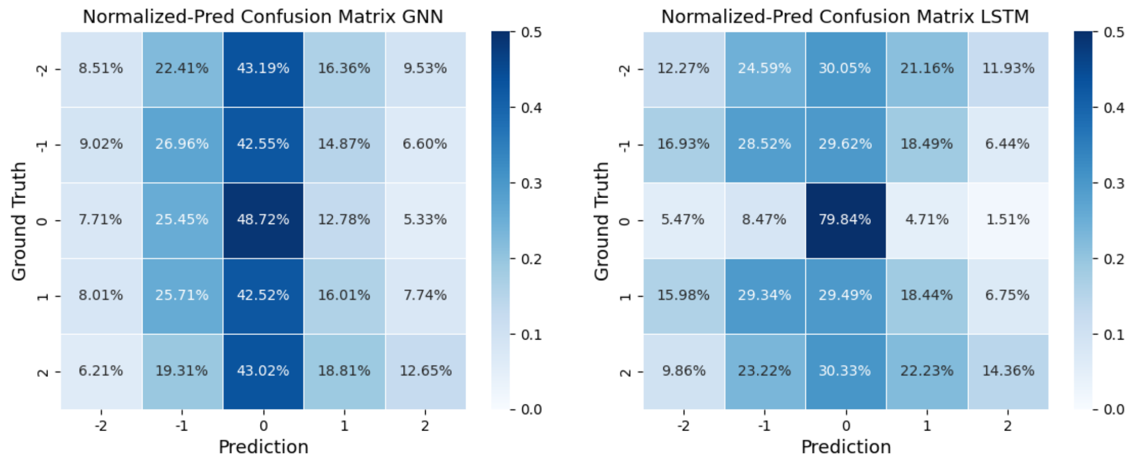


Figure 19: AT-GCN_6 and LSTM_4 normalized confusion matrix (Recall)

To obtain a useful confusion matrix, we can scale the predictions using the distribution of the predictions and the distribution of the target, thus, infer the class labels. Two heatmaps of the best AT-GCN model, and the best LSTM model can be seen in figure 19.

Whats interesting with these matrices, is that the LSTM appears to outperform the GNN in terms identifying stocks with no interest (label 0), the LSTM has a recall of 79.84%. The precision is also slightly higher when considering the more important classes, such as label 2 and 1. In terms of inferred classification, the LSTM appears to be slightly better than the GNN, however the differences in the metrics are not substantial.

We can further analyze the models precision by defining a binary classification of the predictions. Consider top K stocks, defined by the highest predictions for each day in the dataset. In the entire test duration, all stocks used from the top K predictions are considered True, predictions with a ground truth target label of 1 or 2, are considered ”True positives”, while ground truth target labels -2, -1 and 0 are considered ”False negatives”. The table below illustrates the binary precision of each model, for various values of K figure 20.

Once again, the results are mixed, with the 7th iteration of the AT-GCN models having a seemingly higher binary precision on labels 1&2 across most values of K. The proportion of 1&2 set the benchmark

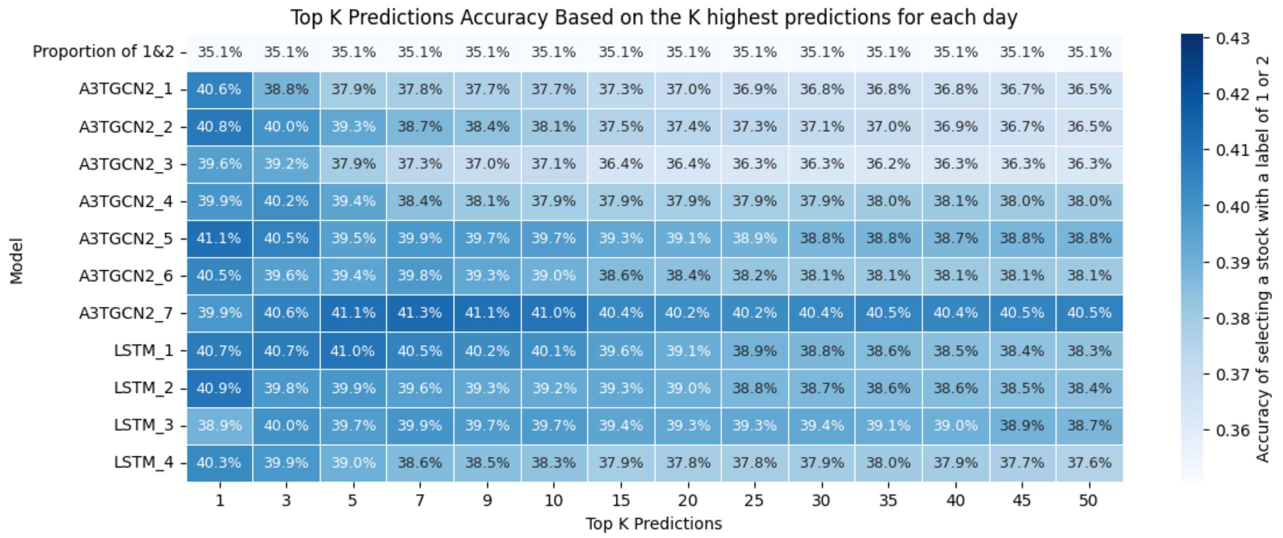
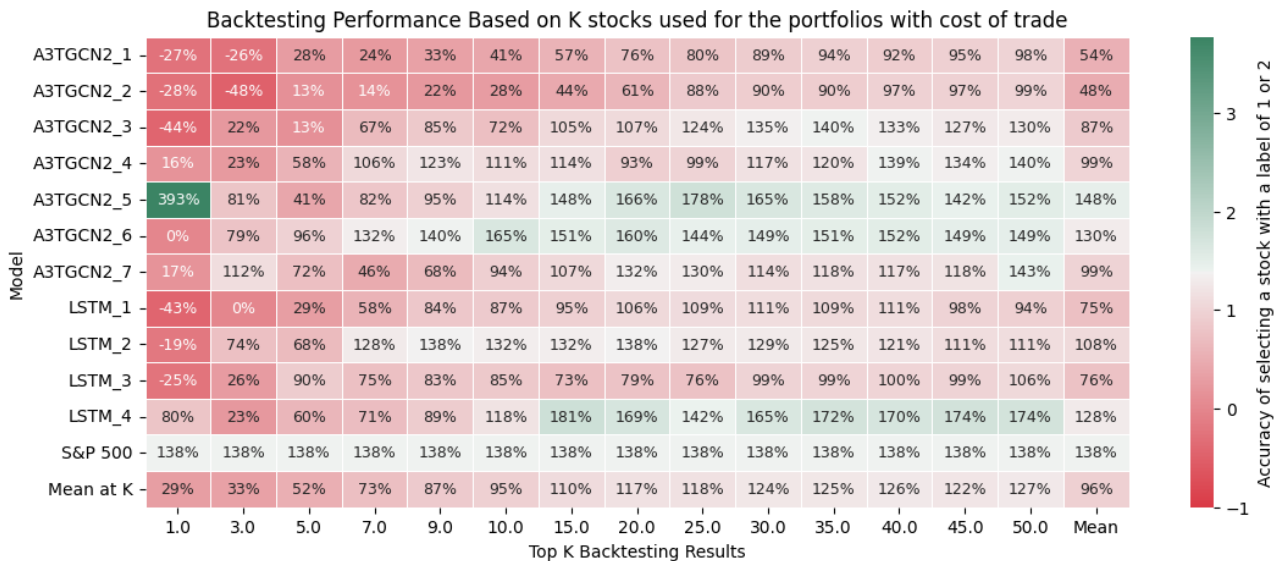


Figure 20: Evaluation of inferred precision of each model



Model	1.0	3.0	5.0	7.0	9.0	10.0	15.0	20.0	25.0	30.0	35.0	40.0	45.0	50.0	Mean
A3TGCN2_1	-27%	-26%	28%	24%	33%	41%	57%	76%	80%	89%	94%	92%	95%	98%	54%
A3TGCN2_2	-28%	-48%	13%	14%	22%	28%	44%	61%	88%	90%	90%	97%	97%	99%	48%
A3TGCN2_3	-44%	22%	13%	67%	85%	72%	105%	107%	124%	135%	140%	133%	127%	130%	87%
A3TGCN2_4	16%	23%	58%	106%	123%	111%	114%	93%	99%	117%	120%	139%	134%	140%	99%
A3TGCN2_5	393%	81%	41%	82%	95%	114%	148%	166%	178%	165%	158%	152%	142%	152%	148%
A3TGCN2_6	0%	79%	96%	132%	140%	165%	151%	160%	144%	149%	151%	152%	149%	149%	130%
A3TGCN2_7	17%	112%	72%	46%	68%	94%	107%	132%	130%	114%	118%	117%	118%	143%	99%
LSTM_1	-43%	0%	29%	58%	84%	87%	95%	106%	109%	111%	109%	111%	98%	94%	75%
LSTM_2	-19%	74%	68%	128%	138%	132%	132%	138%	127%	129%	125%	121%	111%	111%	108%
LSTM_3	-25%	26%	90%	75%	83%	85%	73%	79%	76%	99%	99%	100%	99%	106%	76%
LSTM_4	80%	23%	60%	71%	89%	118%	181%	169%	142%	165%	172%	170%	174%	174%	128%
S&P 500	138%	138%	138%	138%	138%	138%	138%	138%	138%	138%	138%	138%	138%	138%	138%
Mean at K	29%	33%	52%	73%	87%	95%	110%	117%	118%	124%	125%	126%	122%	127%	96%

Figure 21: Backtest performance for different values of K stocks in the portfolio - Including Cost of Trade at 0.08%

for the precision, showing the proportion of available labels with 1&2 in the test set.

5.4 Backtesting results

With mixed signals from both types of deep learning models, we can obtain a clarifying answer of the models performance using the backtester. Originally, the validation predictions were to be used, to optimize and find the optimal backtesting parameters for each model. These parameters were to be used on a backtest using the test predictions, enabling the possibility to show results on an unseen dataset.

Unfortunately, the GNN model's validation dataset were not saved in an ideal format. Attempts were made to reconstruct the validation predictions, however this could unfortunately not be done without re-training all of the models again.

To evaluate the models performance in a simulated environment, we run the backtester for each model, using different values for K, representing the amount of stocks in the portfolio each month.

Using the simulated results from the backtesting performance figure 21, we find that the best composition for the AT-GCN models appear to be model 5, using $K = 1$ stocks pr. month and for the LSTM models the LSTM_4 using $K = 15$ stocks in its portfolio. In a realistic setting, using 1 stock in a portfolio discards all prior knowledge gained of risk management. We will consider the results of the AT-GCN at $K = 1$ as an outlier, and use the next best available value $K = 25$. The color scheme is defined based on the buy and hold performance of the S&P 500 being neutral, easily displaying the relative performance to the benchmark.

Comparing the final results from each type of model, we see that they each perform slightly better than a buy and hold strategy of the S&P 500 in terms of the cumulative returns, however the LSTM model is

Table 10: Performance Metrics with 0.08% Cost of Trade

Metric	**A3TGCN2_5,** $K = 25$	**LSTM_4,** $K = 15$	**S&P 500**
Cumulative Return (%)	177.24	179.42	138.48
Annualized Return (%)	20.45	20.62	17.19
Annualized Std (%)	23.93	30.61	20.28
Maximum Drawdown (%)	-24.65	-36.59	-33.72
Sharpe Ratio	0.729	0.576	0.699

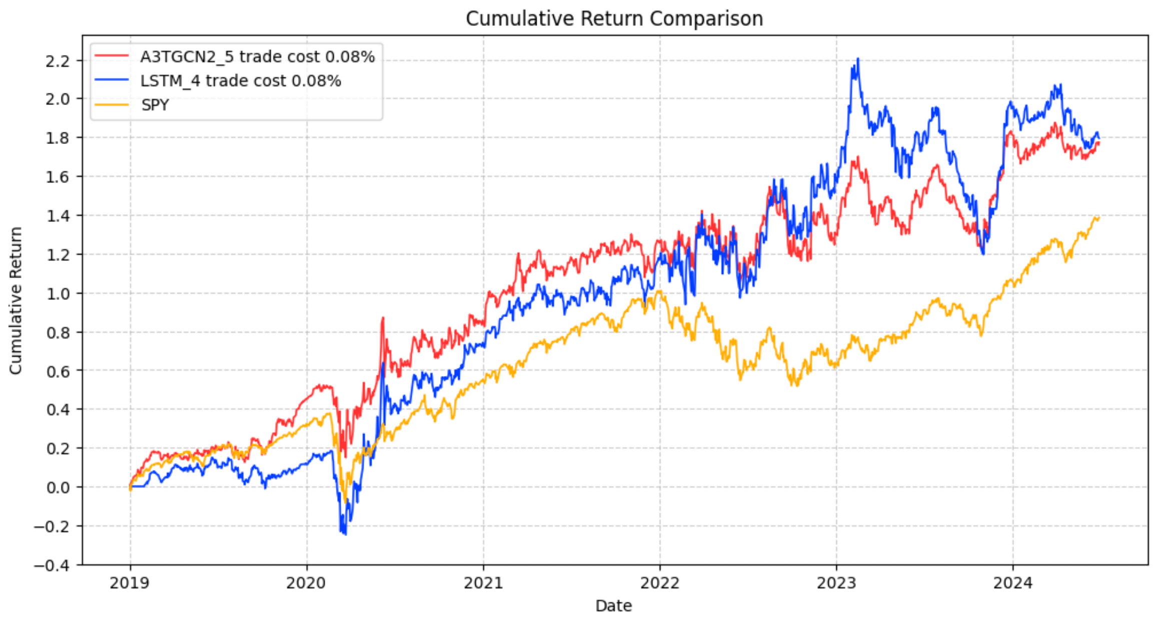


Figure 22: Cumulative returns comparison of AT-GCN, LSTM, and SPY, including cost of trade.

more volatile than the AT-GCN table 10. The direct comparison of the daily returns history of the 2 models can be seen at figure 22.

6 Discussion

The goal of this project was to develop and evaluate models capable of predicting stock performance within the S&P 500. Specifically, focusing on comparing the efficacy of incorporating price correlation data as edge features, as proposed in (Soleymani and Paquet, 2021). Significant efforts were put into constructing meaningful features to capture the complexities of the financial data. Despite these efforts, the project faced challenges in achieving substantial improvements in validation performance during training. These challenges highlight the importance of thoroughly evaluating not just the models but also the data and methodologies employed. This discussion aims to reflect on the successes, limitations, and insights gained from the project while addressing methodological flaws, biases, and potential areas for future improvement. Furthermore, the discussion aims to contextualize the project's findings and propose actionable recommendations for advancing predictive models in financial applications.

6.1 Analysis of challenges

The following sections highlight the key challenges encountered during this project, focusing on feature engineering, model limitations, and their impact on overall performance.

6.1.1 Feature Engineering

The feature engineering process in this project involved significant efforts to create a comprehensive set of features, intended to emulate the observations made by human counterparts. One limitation was the lack of evaluating the relative importance of these features before and after pre-processing. Without identifying the most impactful features, the models may have been constrained by the inclusion of irrelevant or redundant inputs. This oversight was unfortunately the product of poor time management, and likely contributed to suboptimal feature utilization, limiting the ability of the models to extract meaningful patterns from the data.

6.1.2 Model Limitations

Both the GNN and LSTM models exhibited near immediate stagnation in their validation scores, indicating limited improvements in predictive performance despite adjustments to hyperparameters and loss the function. One significant factor was the computational cost associated with the high dimensionality of the feature set. This constraint unfortunately ruled out the possibility of exploring longer input windows, which are critical for capturing temporal dependencies over extended periods. Consequently, the models may have struggled to fully leverage temporal patterns that could have enhanced their predictions.

A notable issue was the delayed focus on model interpretability. The interest in understanding the decision-making processes of the models arose late in the project. Unfortunately, integrating tools such as SHAP values and Integrated Gradients into the pipeline of the existing models was not possible to implement within the time constraints of the project.

6.2 Model Behavior Observations

To gain a deeper understanding of the trained models' behavior, a post-analysis was conducted on the portfolios generated during the backtesting simulation. The focus was placed on the best-performing models: AT-GCN 5 and LSTM 4. This analysis aimed to uncover any patterns or biases in stock selection.

6.2.1 AT-GCN Model's preference for American Airlines

A notable observation was the AT-GCN model's clear preference for American Airlines (AAL), as illustrated in Figure 23. A plausible explanation for this behavior could be that the target values for ALL during the training period, were skewed in contrast to other stocks, potentially leading the model to favor it as a consistently profitable option. However, upon closer inspection, this explanation weakens as the proportion of 1&2 labels in the training set is not substantially greater than the others. Additionally, similar behavior was not observed for the other stocks in 23.

To underline the magnitude of this bias for the frequent inclusion of AAL in the portfolio, it appears in 55.9% of the monthly portfolios during the test period ($\frac{38}{68}$ months). With the additional consideration of 631 active stocks being present in the graph during the test period, this highlights a case of plausible node-level over-fitting.

6.2.2 Unexpected Node Inclusion

Another unusual observation was the inclusion of CSRA, ranked as the 7th most popular stock in the AT-GCN portfolios. Despite CSRA's node features and target values being entirely zero, with the exception of an inactive flag (set to -1, as defined in 4.5.5), CSRA was still included in 25% of the monthly portfolios. This anomaly suggests that the AT-GCN model's feature importance is specified at the node level, rather than uniformly across the graph, further complicating the goal of achieving model generalization.

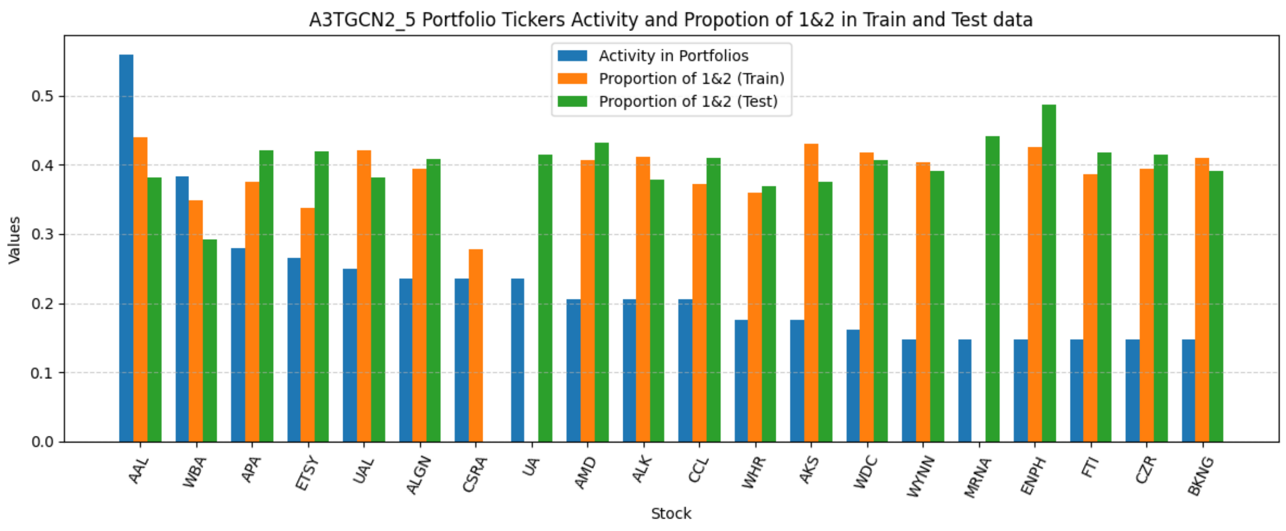


Figure 23: AT-GCN 5 post portfolio analysis

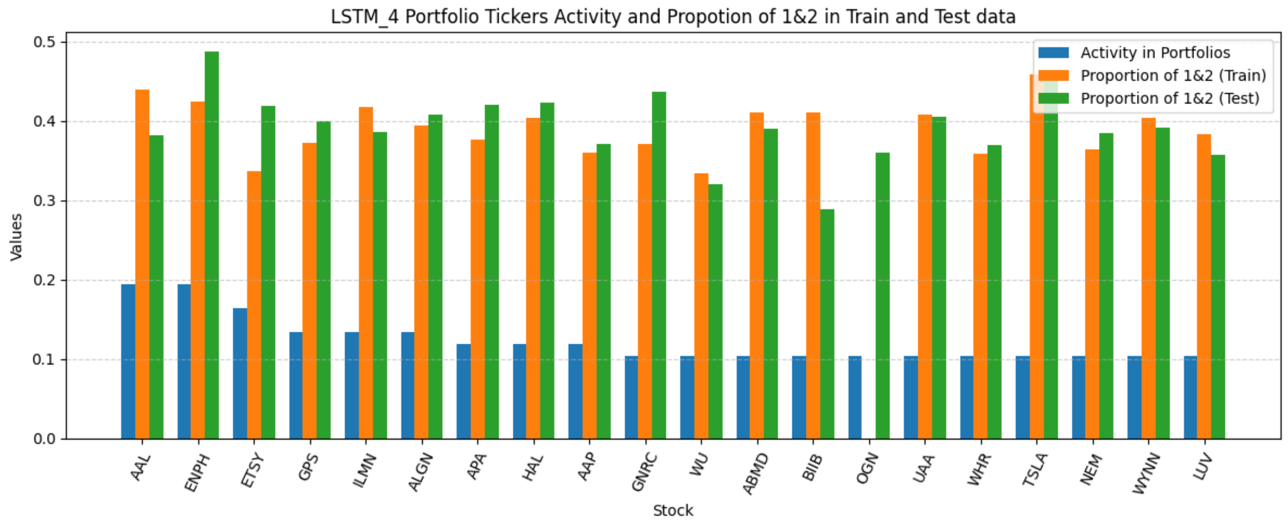


Figure 24: LSTM 4 post portfolio analysis

6.2.3 LSTM Model Behavior

In contrast, the LSTM model exhibited a more dispersed preference for specific stocks, as shown in 24. While American Airlines (AAL) remained the most frequently selected stock, the stock's dominance was significantly reduced compared to the AT-GCN portfolios. However, the proportion of the most popular stocks typically ranged between 10% and 20%, suggesting that the LSTM's decision-making process was more varied but, perhaps slightly random.

The apparent randomness in stock selection becomes particularly evident when comparing all stocks within each portfolio, to the 20 best-performing stocks during the test period 11. The best-performing stocks were identified based on their buy & hold returns over the test period, while each model's stock preference were quantified as the normalized sum of portfolio weights. The weights wwere averaged over the test period (68 months) and scaled by $K = 25$ for the AT-GCN model and $K = 15$ for the LSTM model.

To create a meaningful comparison of the values, an auxiliary metric called "Influence" was defined as

$$\textbf{Influence} = \frac{\textbf{Sum}}{K}$$

where the numerator represents the sum of normalized weights of the 20 best-performing stocks and the denominator (K) corresponds to the amount of stocks in each models portfolio. Influence measures

Table 11: Top 20 stocks during the test period, and the normalized portfolio weigths of AT-GCN and LSTM

Ticker	**AT-GCN**	**LSTM**	**Returns (%)**
SMCI	0.102	0	5841.62
NVDA	0.055	0.031	2185.97
ENPH	0.167	0.215	1930.75
BLDR	0.011	0	958.18
ANF	0.013	0	888.33
KLAC	0	0	851.66
LLY	0.013	0.042	817.42
TSLA	0.058	0.117	762.15
AVGO	0.019	0	739.16
AMD	0.209	0.069	702.23
PWR	0	0	673.23
LRCX	0.013	0	662.48
FICO	0.029	0	650.26
DECK	0.014	0	634.86
DDS	0.046	0	615.57
AMAT	0.046	0	613.29
MPWR	0	0.025	592.42
GME	0.026	0	584.22
CDNS	0.096	0.050	579.21
CMG	0	0.013	562.79
Sum	0.917	0.562	**Base Case**
Influence	3.668 %	3.747 %	3.169 %

the extent to which these high-performing stocks contributed to the portfolio's overall performance. The specific time of a stocks inclusion does impact the performance contribution of the stock in the portfolio, however, excluding the time dependent returns provides a better overview of the intended insights from the table.

For comparative purposes, a Base Case benchmark was created, defined as the proportion of the 20 best-performing stocks relative to the total number of active stocks during the test period $\frac{20}{631}$.

The key observation lies in comparing the Influence value against the Base Case. Ideally, the models' influence scores should exceed the Base Case, indicating that the models successfully prioritized high-performing stocks. However, while both models outperformed the Base Case, the observed difference was marginal and substantially lower than expected.

6.2.4 Summary of Model Behavior

These findings aim to illustrate the differing behaviors of the AT-GCN and LSTM. The AT-GCN demonstrated clear biases, particularly for specific stocks like AAL and CSRA, suggesting strong reliance on node-level features. In contrast, the LSTM model showed diverse stock preferences, with a degree of apparent randomness. Both models' limited ability to prioritize high-performing stocks above the Base Case benchmark highlights opportunities for improving feature selection to obtain generalization in future research.

6.3 Feature Insights

In an attempt to further understand the behavior of the models' predictions required further investigation into the importance of the input features. Given the unusual stock preferences observed in the portfolios generated by both the GNN (AT-GCN) and LSTM models (as discussed in section 6.2), analyzing feature contributions was a logical next step to gain insights for improving future iterations.

To evaluate feature importance, an XGBoost model was implemented as a substitute model, as both

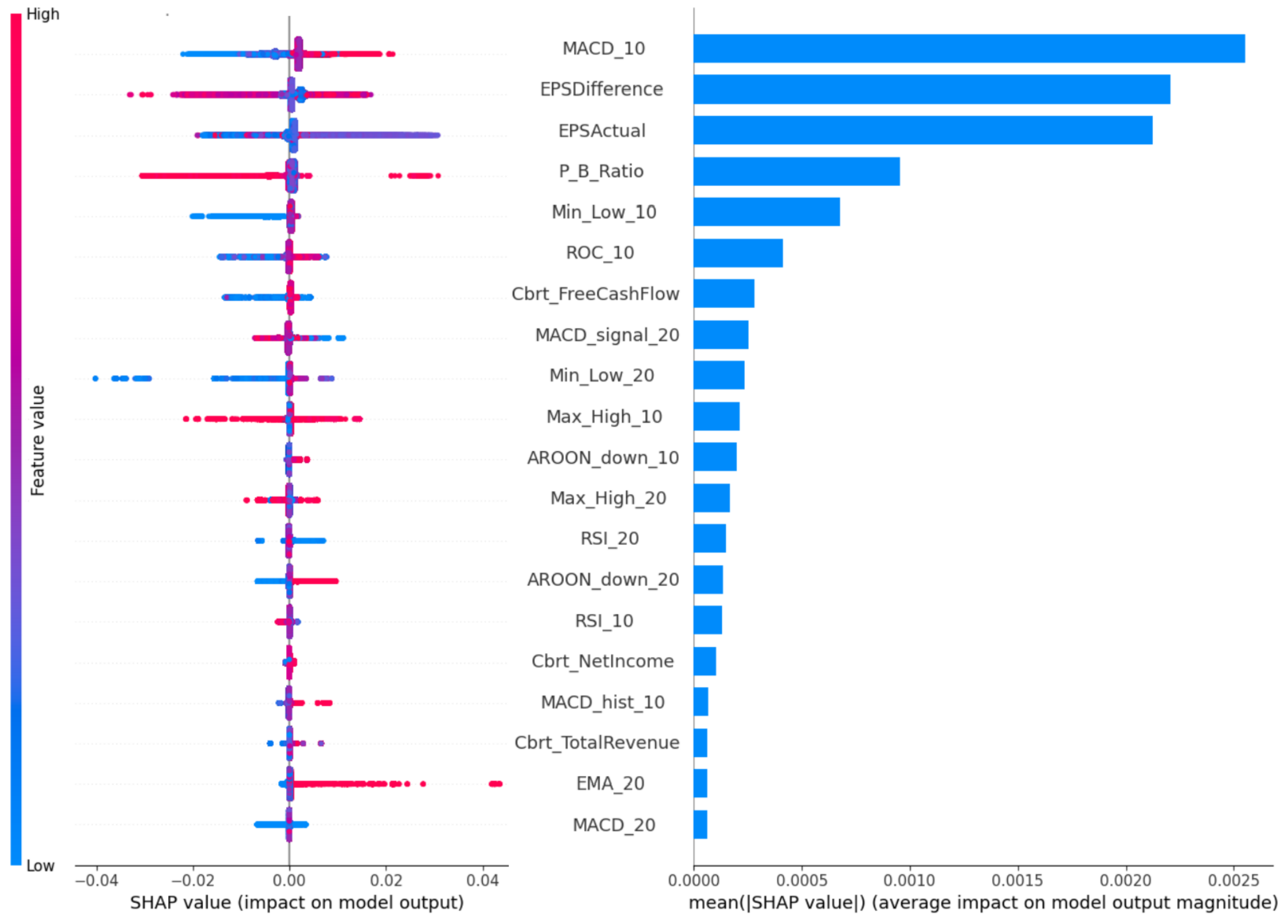


Figure 25: SHAP value summary on test data

AT-GCN and LSTM have properties making post model training feature analysis more challenging than a model which does not use windowed data, or graph structures as input. SHAP values enable a detailed understanding of the contribution of each feature to the model's predictions, identifying which features had the most influence and which contributed minimal value. The choice of using XGBoost comes from a practical stand point, as tree-based models inherently produce SHAP values in the tree node creation process, while gradient boosting approximates the non-linear relationships more complex models are able to create.

When inspecting the SHAP values figure 25, certain features clearly dominated the decision-making process. Interestingly, the MACD_10 appear to dominate the decision process, alongside the EPSDifference and EPSActual. What's interesting with the EPS metrics (earnings per share), is that these metrics are release on a quarterly interval, and notoriously effect the stock prices of companies. Needless to say, it's unsurprising that these metrics appear in the top, however, the current implementation has duplicated these values for the entire future quarter for the given stock, without any encoding of how recently these features have appeared. Suppose that the EPSDifference is high, meaning that the company's revenue has surpassed the expectations of investors. While this is arguably great information to base an investment on, this information is time sensitive, and only relevant in a brief period before the stock price has reached a new equilibrium due to portfolio rebalancing of investors.

Another notable observation is the EMA_20. After the transformation and normalization processes, low values of EMA correspond to an acute increase in stock price Equation 16. The left summary plot in figure 25 illustrates that high values of EMA impacts the predicted values positively. This can be rationalized by ”just buy low and sell high”, while a counter argument could be ”why invest in a sinking ship?”.

An additional observation emerged from the SHAP decision plot based of input data with the target 2. figure 26. The inclusion of all features might have caused the models' prediction values to remain low and centered around the target mean, suggesting that the other models also struggled to effectively utilize the large feature set, potentially leading to an over-smoothing effect. Although the decision plot is only based on 5000 samples, it highlights the plausibility of features diluting the models' ability to focus on stronger, more predictive signals. The top area of the plot show the model's regression values for each instance which for all instances is significantly lower than 2, even for the high predictions (red detached lines).

The inspection of SHAP values from the XGBoost model solidifies the evidence that feature redundancy is likely a contributing factor to the underperformance and low standard deviation of predictions observed in both the AT-GCN and LSTM models. While the feature importance results apply directly to XGBoost and not to the GNN or LSTM architectures, the findings support a consideration: many features could have been discarded in favor of improving temporal depth (e.g., larger time windows) and increasing the graph connectivity (e.g., adding more meaningful edges to the GNN).

This insight is particularly relevant for the GNN, where node-level bias was observed section 6.2, where the inclusion of unnecessary features likely amplified node-level over fitting, leading the model to favor specific stocks (e.g., AAL, and CSRA). The lack of feature reduction could likely enhance the models' ability to generalize and identify profitable opportunities across a broader range of stocks.

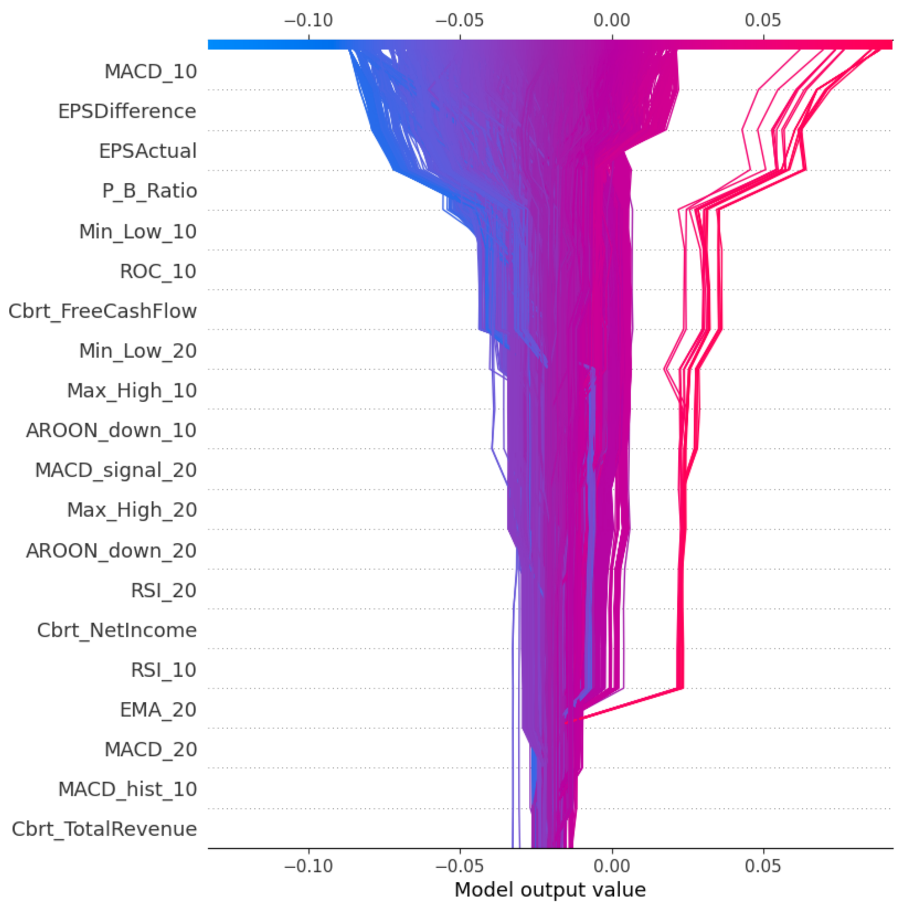


Figure 26: SHAP decision plot (5000 samples of label 2)

6.4 Biases and Methodological Challenges

While deliberate measures were taken to ensure fairness and transparency in the analysis, certain aspects of the dataset, feature engineering, and evaluation methodologies posed biases and methodological limitations which must be acknowledged.

6.4.1 Dataset Issues

One of the primary issues encountered was missing values from the quarterly reports. Filing dates and certain fundamental features were occasionally absent or incorrect, necessitating imputation strategies. Forward-filling previously available values and filling missing entries with neutral values ensured continuity but may have introduced biases, especially for smaller firms with less consistent reporting practices. Few scenarios were discovered in the exploratory data analysis, where a collection of 4 quarterly reports for a year would consist of 1 annual report, and 3 empty reports. Examples of missing filing data (essential information to avoid time leakage in financial time series analysis) lead to extreme measures of disadvantageous value replacements, such as defining the filing date 2 months after the effective date.

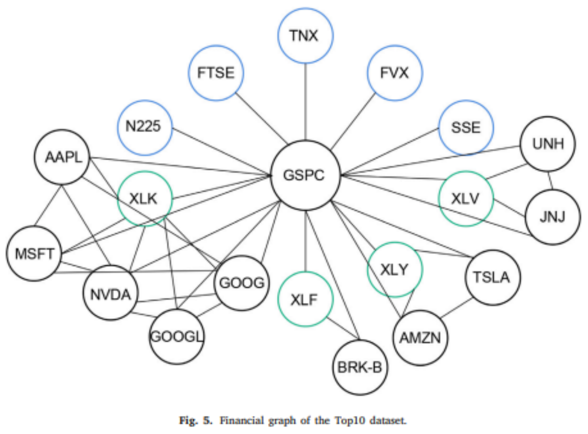


Figure 27: (Sun et al., 2024) portfolio composition

Another post-analytic clarification involved the selection pool of stocks. This was intentionally implemented to mitigate survivorship bias and favoritism, a critical consideration motivated by prior methodologies observed in (Soleymani and Paquet, 2021) and (Sun et al., 2024).

The composition illustrated in figure 27 highlights various color dimensions of nodes. Blue nodes represent global indices, structurally comparable to the S&P 500, while green nodes correspond to sector-specific Exchange Traded Funds (ETFs). The black nodes surrounding the graph are of particular interest, as they represent the selection of the top 10 stocks defining the selection pool in the referenced research. Notably, the portfolio described in (Sun et al., 2024) consists of seven leading technology companies, a prominent financial institution (Berkshire Hathaway, BRK-B), and two major healthcare stocks (Johnson & Johnson and UnitedHealth Group). Furthermore, the test period, spanning 2017 to 2024, aligns with the period of highest average growth for the seven selected technology companies, underscoring the favorable conditions for their performance during this time frame. Needless to say, contrasting this portfolio to the S&P 500 is unarguably biased, as the selection strongly favors the test period, and was influenced by current information.

To avoid replicating prior stated bias, the selection pool was based on the S&P 500 stocks at any given time over the 28-year period. While this approach of complete inclusion allowed for a broader and unbiased analysis, the approach backfired as there was a substantial amount of increased noise in the data due to the inclusion of smaller firms with less robust financial reporting.

Larger, more established firms typically have higher-quality accounting and reporting standards. A narrower focus on the top 100 stocks, such as the NASDAQ-100, could have improved data integrity while reducing the graph's size and computational complexity. This refinement might have also aligned better with the project's goal of identifying actionable patterns in high-quality data.

6.4.2 Feature Engineering

Most of the feature engineering and normalization techniques were effective, although in hindsight, certain choices are subject for reconsideration. Apart from the insights of redudant features in section 6.3, one overlooked opportunity was the inclusion of temporal encoding capturing the delay between a quarterly report's filing date and the specific date of an entry. This hypothetical feature could have provided additional context, helping the model recognize the timing and relevance of the fundamental data.

Another promising avenue, which was considered but not implemented, involved using multiple time sequences within a single window. For instance, a 5-day window could have included both daily price data and five quarters of fundamental data, allowing the model to simultaneously learn from short-term trends and long-term financial performance. This approach might have aided the model's understanding of a stocks long term performance, and contrast the information to current price information.

6.4.3 Backtesting Bias

While this study critically evaluated related research in section 6.4.1, it is important to acknowledge that the methodology in this project is not free from biases, with the backtesting warranting particular attention.

One notable bias stems from the tuning of backtester parameters. Ideally, these adjustments should have been performed exclusively on the validation dataset, ensuring that the test data remained entirely unseen. However, due to time constraints and logistical challenges discussed in section 5.4, this principle was not strictly adhered to. The iterative implementation of saved model parameters and predictions for the AT-GCN models encountered technical issues, with correct storage of results and model weights only corrected after the sixth iteration. This oversight advertently introduces exposure to test data during the backtester tuning process in the area dictating the portfolio size, however, the portfolio weights, thus the order of the portfolio composition for each rebalancing period was not influenced in this manner.

Initial considerations for the backtester included incorporating additional strategies, weighting schemes, and functionalities such as stop-loss and take-profit mechanisms. However, fully aware that the optimal parameters for these metrics would pose biased results, the decision to prioritize a clear narrative and fairness in model comparisons, led to the exclusion of these enhancements. Discarding this additional layer of complexity still permits focus on the comparative research objective of the input data structures.

While this methodological limitation undermines the authenticity of the reported best-performing K values for the AT-GCN and LSTM models, figure 21 provides a comprehensive overview of all permutations of models and K values considered. This broader presentation of results serves to mitigate the potential bias of the findings.

Most importantly, the analysis reveals that cumulative returns were relatively stable for $K > 15$. The reported results based on the best-performing configurations (AT-GCN at $K = 25$ and LSTM at $K = 15$) introduces a limitation that must be considered when deriving conclusions of the research.

6.5 Comparison with Prior Work

The research objective of this thesis aims to explore the advantages of employing Graph Neural Networks (GNNs) for portfolio management and to offer broader insights into the applicability of modeling a pool of available stocks as a graph representation.

Discussed in section 2.2, (Soleymani and Paquet, 2021) and (Sun et al., 2024) set an important benchmark for applications of GNN's for portfolio management. While the methodology defining the available stocks for the model to select differs from this project, the reported results of (Sun et al., 2024) can be compared framework to this study.

The framework of (Sun et al., 2024) was tested on two datasets: one consisting of 10 stocks and another of 28 stocks defined by (Soleymani and Paquet, 2021). Both papers focused on generating portfolio weights for all stocks in the pool, which contrasts with the broader approach in this project, where the portfolio size varied ($1 \leq K \leq 50$) and was drawn from a considerably larger universe of assets, including all stocks in the S&P 500 index throughout the datasets period. The extended dataset used in this study posed greater challenges but also provided an opportunity to evaluate the robustness and scalability of the GNN and LSTM models by letting the portfolio be a variable to optimize.

The findings of (Sun et al., 2024) reveal marginal improvements in portfolio performance compared to a uniform-weighted strategy of same portfolio compositions. These findings align with the outcomes of this report, where the best-performing GNN and LSTM models exhibited only a modest edge over the S&P 500, as seen in figure 22. While there where numerous interesting artifacts found in the methodology, potentially limiting the results, it emphasizes the difficulty of significantly surpassing market benchmarks, even when applying advanced deep learning techniques.

Despite differences in features and datasets, both studies underline that GNNs are a powerful tool for their predictive advantages in portfolio management. Comparing the methodologies also reveals complementary insights. GraphSAGE relied heavily on a static graph representation and SHAP-based feature selection to ensure model robustness. Differing from (Sun et al., 2024), this reserach paper intentionally included all features, and incorporated a temporal focus by using the AT-GCN, and LSTM model.

The findings from both studies suggest that optimizing feature engineering and expanding datasets to include richer and more relevant financial signals may be critical for realizing the full potential of a GNN-based portfolio management system, further emphasizing the decision of feature inclusion.

6.6 Reflections and Future work

Throughout this research, various opportunities for refinements of the pipeline were identified. Some were implemented during the project, while others emerged too late in the process to be incorporated. The following subsections outline potential directions for further exploration, informed by the findings and limitations encountered in this study.

6.6.1 Incorporating Feature Selection Techniques

Feature selection was found to be a critical step in reducing noise and enhancing the predictive capabilities of machine learning models. While this study intentionally included an extensive feature set, the addition of feature selection techniques earlier in the pipeline could have helped the quality of the input data for model training.

Further expanding the basis of features could also prove valuable cyclical information. Two auxiliary metrics from (Sun et al., 2024) where found particularly useful in their SHAP analaysis, namely the ”Smoothed U.S. Recession Probabilities”, and ”Sticky Price Consumer Price Index”, which are macro economic indicators of the U.S consumer market. Directly translated, these metrics act as indicators for financial seasonality which could provide useful context for the task. Although initially considered for this project, macro economic factors were unfortunately discarded due to the sheer amount of moving parts in the project.

Further expanding the feature set to include cyclical information could provide valuable insights into financial seasonality. Notably, two auxiliary metrics from (Sun et al., 2024), the ”Smoothed U.S. Recession Probabilities” and the ”Sticky Price Consumer Price Index,” were identified as particularly effective in their SHAP analysis. These macroeconomic indicators of the U.S. consumer market, offer critical context for understanding financial trends and seasonality. Although these factors were initially considered for inclusion in this project, they were ultimately omitted due to the complexity and scope of the study, which involved numerous moving components.

6.6.2 Refinement of Stock Selection Pool

The current study utilized the entire S&P 500 as the selection pool of stocks, which, while comprehensive, introduced potential data quality issues. As suggested in section 6.4.1, reducing the selection pool to improve the probability of high-quality data could improve the consistency of the training data, indirectly reducing computational and memory limitations discussed before, allowing for deeper exploration of model architectures and hyperparameter tuning.

7 Conclusion

This thesis has explored the comparative performance of the AT-GCN model, and the LSTM model applied in the domain of portfolio management. By framing the stock market as a graph problem, this study aimed to uncover the benefits of representing financial data as a holistic graph structure of the entire stock market versus independently assessing assets.

Both methodologies yielded results that moderately outperform the S&P 500 during the test period. The best performing LSTM achieved a cumulative return of 179.42%, slightly outperforming the best performing AT-GCN, which demonstrated a cumulative return of 177.24%, both surpassing the S&P 500's return of 138.48%. Despite this, the AT-GCN exhibited a slight edge in risk-adjusted returns, achieving a Sharpe ratio of 0.729 compared to the LSTM's Sharpe ratio of 0.576. In comparison, the S%P 500's Sharpe ratio for the given duration was 0.699. This edge in the Sharpe ratio can be justified by various factors, one of which is the inclusions of 25 stocks for the AT-GCN portfolios, versus the portfolio size of 15 stocks used for the LSTM, underscoring the value of diversification stated in section 2.1.

The analysis highlighted the impact of feature selection on model performance, as the inclusion of all features appeared to have clouded the judgment of both models, as suggested by the SHAP plot figure 25 in section 6.3. This finding indicates the potential benefit of a more targeted feature selection approach to enhance predictive accuracy for future research.

Additionally, the models demonstrated distinct behavioral tendencies in their stock preferences. The AT-GCN model exhibited node-level biases, favoring specific stocks such as American Airlines (figure 23). In contrast, the LSTM appeared less discriminatory, with no clear preference for specific stocks, as observed in figure 24. While the LSTM showed slightly higher precision and recall scores across all target labels, it's lack of confidence in stock selection appeared to have diminished the portfolio performance.

A significant insight aiding the conclusion of the research question lies in the minor performance decrease of the AT-GCN_7 iteration compared to its counterparts (figure 21). Notably, this model intentionally excluded edge weights, to uncover the information gain of the included edges, which attributed to a performance decrease. However, the observed performance difference of the LSTM and the AT-GCN remains negligible.

While the performance of both models were modest, the study underscores future work potentially aiding in higher performance, such as feature selection, backtester optimization, and limiting the selection pool of available stocks to increase data quality.

In conclusion, the findings of this research does not indicate an advantage of using graph-based representations for the stock market, in favor of assessing individual stock data independently.

References

George S. Atsalakis and Kimon P. Valavanis. 2009. Forecasting stock market short-term trends using a neuro-fuzzy based methodology. *Expert Systems with Applications*, 36(7):10696–10707.

Aurum Research. 2024. Hedge fund industry performance deep dive – h1 2024. Accessed: 2024-11-22.

Saxo Bank. 2024. Rates and conditions: Stock commissions. https://www.home.saxo/rates-and-conditions/stocks/commissions. Accessed: 2024-12-16.

Zvi Bodie, Alex Kane, and Alan J. Marcus. 2014. *Investments*, 10th edition. McGraw-Hill Education, New York, NY. Accessed: 2024-11-22.

Pietro Bongini, Niccolò Pancino, Asma Bendjeddou, Franco Scarselli, Marco Maggini, and Monica Bianchini. 2024. Composite graph neural networks for molecular property prediction. *International Journal of Molecular Sciences*, 25(12):6583.

Jian Cao, Zhi Li, and Jian Li. 2019. Financial time series forecasting model based on ceemdan and lstm. *Physica A: Statistical Mechanics and its Applications*, 519:127–139.

Pat Tong Chio. 2022. A comparative study of the macd-based trading strategies. *arXiv preprint arXiv:2206.12282*.

Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph neural networks for social recommendation. In *Proceedings of the 2019 World Wide Web Conference (WWW '19)*, pages 417–427. ACM.

Forbes. 2024. How does the current rally in gamestop stock compare with the fall in 2008 crash? current-rally-in-gamestop-stock. Accessed: 2024-11-22.

Christian Emil Haldan. 2023. Vix vertex predictions. ITU Research Project Report. LSTMs for predicting the Volatility Index (VIX) with a focus on technical indicators and custom loss functions.

J. B. Heaton, N. G. Polson, and J. H. Witte. 2017. Deep learning in finance. *SSRN Electronic Journal*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Investopedia. 2024a. Aroon indicator. https://www.investopedia.com/terms/a/aroon.asp. Accessed: 2024-12-19.

Investopedia. 2024b. Bollinger bands. https://www.investopedia.com/terms/b/bollingerbands.asp. Accessed: 2024-12-19.

Investopedia. 2024c. Exponential moving average - ema. https://www.investopedia.com/terms/e/ema.asp. Accessed: 2024-12-19.

Investopedia. 2024d. Price rate of change (roc). https://www.investopedia.com/terms/p/pricerateofchange.asp. Accessed: 2023-12-19.

Investopedia. 2024e. Relative strength index (rsi). Accessed: 2024-12-18.

Harry Markowitz. 1952. Portfolio selection. *Journal of Finance, Vol 7*, pages 77–91.

Christopher Olah. 2015. Understanding lstm networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/. Accessed: 2024-12-19.

Bloomberg Terminal pricing. 2023a. Bloomberg terminal pricing: How much does bloomberg terminal cost? Accessed: 2024-12-03.

Refinitiv Eikon pricing. 2023b. Bloomberg vs. capital iq vs. factset vs. thomson reuters eikon. Accessed: 2024-12-03.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80.

Sima Siami Namin and Akbar Siami Namin. 2018. Forecasting economic and financial time series: Arima vs. lstm. *Texas Tech University*.

Farzan Soleymani and Eric Paquet. 2021. Deep graph convolutional reinforcement learning for financial portfolio management - deeppocket. *arXiv preprint arXiv:2105.08664*. https://arxiv.org/abs/2105.08664.

S&P Dow Jones Indices. 2024. S&p u.s. indices methodology. https://www.spglobal.com/spdji/en/documents/methodologies/methodology-sp-us-indices.pdf. Accessed: 2024-11-22.

Qiguo Sun, Xueying Wei, and Xibei Yang. 2024. Graphsage with deep reinforcement learning for financial portfolio optimization. *Expert Systems With Applications*, 238:122027.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*. Accessed: 2024-11-22.

Jiawei Zhu, Yujiao Song, Lin Zhao, and Haifeng Li. 2020. A3t-gcn: Attention temporal graph convolutional network for traffic forecasting. *arXiv preprint arXiv:2006.11583*. Accessed: 2024-11-22.