IMPERIAL

Innovative Approaches to Asset Prediction: Combining Deep Learning with Financial Modelling

Final Report

Author

C. Ioannidis

CID: 02533490

Supervised by

Prof. Pasquale Della Corte

Prof. Walter Distaso

Dr Lluis Guasch

A Thesis submitted in fulfillment of requirements for the degree of
Master of Science in Applied Computational Science and Engineering

Department of Earth Science and Engineering
Imperial College London
2024

Abstract

This project addresses the increasing complexity and volatility in financial markets through the development of advanced analytical tools. Leveraging the theoretical foundations established by Bryan Kelly and Kusuma, we propose refining Convolutional Neural Networks (CNNs) to enhance the prediction of financial asset behaviors. Traditional methods like Auto Regressive Integrated Moving Average (ARIMA) often fail to capture the nonlinear dynamics and intricate patterns present in financial data. By contrast, CNNs are capable of exploiting image representations of time-series data to learn spatial and temporal features, potentially offering superior predictive accuracy.

In this study, we develop a robust CNN model trained on candlestick chart images generated from a dataset of randomly selected U.S. stocks spanning from the 1990s to 2017. The model's performance was rigorously evaluated through backtesting on the S&P 500 index over a period from June 2019 to June 2024. The results indicate that the well-trained model achieved a Sharpe ratio of 1.20 and maintained a drawdown of 24.46%, significantly outperforming a poorly trained model and traditional benchmarks. The model also demonstrated robustness in identifying market anomalies, such as the COVID-19 pandemic and subsequent volatility in 2022, underscoring its adaptability to diverse market conditions. While these findings highlight the potential of CNNs in financial forecasting, challenges such as overfitting and data balancing remain. Future work will explore the integration of more advanced architectures, such as Transformers, and the inclusion of additional data features to further enhance model performance and generalizability across different asset classes.

Declaration of Originality

I hereby declare that the work presented in this thesis is my own unless otherwise stated. To the best of my knowledge the work is original and ideas developed in collaboration with others have been appropriately referenced.

Copyright Declaration

The copyright of this thesis rests with the author and is made available under a Creative Commons Attribution Non-Commercial No Derivatives licence. Researchers are free to copy, distribute or transmit the thesis on the condition that they attribute it, that they do not use it for commercial purposes and that they do not alter, transform or build upon it. For any reuse or redistribution, researchers must make clear to others the licence terms of this work.

Contents

1 Introduction

Contents

1.1 Introduction

The prediction of financial markets has undergone significant evolution in recent years, driven largely by advancements in machine learning techniques. This study explores the use of Convolutional Neural Networks (CNNs) as advanced predictive models for analyzing time-series data across various asset classes. The primary objective of this research is to enhance the accuracy of financial market predictions by employing CNN architectures capable of capturing the complex patterns inherent in financial data. This project involves a comprehensive approach, beginning with the collection and preprocessing of extensive time-series datasets, followed by the development and iterative refinement of CNN-based models that address the unique challenges posed by financial market forecasting.

1.2 Background and Motivation

Traditional methods for predicting financial markets, such as the Auto Regressive Integrated Moving Average (ARIMA) model, have been widely applied but often struggle to capture the nonlinear and intricate dynamics of financial data. The advent of machine learning, and specifically CNNs, has introduced more sophisticated techniques capable of addressing these limitations. Recent research has demonstrated the efficacy of CNNs in financial market prediction by transforming time-series data into visual formats that can better capture underlying patterns.

For instance, Kusuma et al. (2019) utilized CNNs to analyze historical stock data converted into candlestick chart images, achieving high prediction accuracy for stock markets in Taiwan and Indonesia [1]. Sezer and Ozbayoglu (2019) further advanced this approach by transforming time-series stock data into 2-D bar chart images and applying CNNs to identify trading signals, demonstrating superior performance to traditional methods, especially during bearish market conditions [2]. Zeng et al. (2021) expanded on these concepts by introducing a video prediction model for economic time series that leveraged CNNs' ability to detect spatial patterns in image sequences, outperforming traditional techniques like ARIMA and Prophet [3]. Jiang (2023) also employed CNNs with OHLC (Open, High, Low, Close) charts to forecast stock returns, highlighting the model's capacity to recognize intricate patterns and adapt across different geographical and temporal scales [4]. Collectively, these studies underscore the potential of CNNs to significantly improve asset price prediction accuracy, offering a marked advantage over traditional forecasting methods.

1.3 Research Objectives

The primary goal of this research is to enhance the predictive capabilities of financial market models through the development and refinement of CNN architectures. By focusing on the analysis of time-series data, this study aims to uncover complex patterns that drive market behavior, thus facilitating more accurate and reliable predictions. Achieving higher predictive accuracy is expected to support more effective risk management strategies, particularly in volatile market environments. Additionally, this research seeks to provide insights that can assist investors and financial analysts in making more informed decisions regarding investment strategies, thereby contributing valuable knowledge to the field of financial analytics. The overarching aim is to advance the adoption of more sophisticated machine learning techniques in financial forecasting, promoting a deeper understanding of global market dynamics.

1.4 Methodology Overview

The methodological approach of this study encompasses several stages. Initially, extensive time-series data will be collected and preprocessed to ensure it is suitable for deep learning applications. The data will then be transformed into image formats, such as candlestick charts, to facilitate the training of CNN models. Following data preparation, various CNN architectures will be developed and iteratively refined to optimize their predictive performance. The models will be trained using historical market data to learn patterns and trends that may inform future market movements.

In the final stages of the project, the developed models will be tested within simulated environments to assess their accuracy and practical applicability in real-world scenarios. This evaluation will be conducted through rigorous backtesting and benchmarking against standard market indices, such as the S&P 500, to measure their performance relative to traditional investment strategies.

1.5 Expected Contributions

The anticipated outcomes of this research are expected to provide significant insights into market dynamics, thereby enhancing the decision-making processes in financial investments. By demonstrating the practical utility of CNNs in financial forecasting, this study aims to bridge the gap between theoretical advancements and their real-world applications in financial markets. The findings are expected to contribute substantially to the field of financial analytics, promoting the integration of advanced machine learning techniques into market prediction and portfolio management strategies. This research represents a meaningful contribution to the ongoing discourse on the application of deep learning in financial markets, with implications for both academic research and practical investment decision-making.

2 Methodology

Contents

This section outlines the technical approach used in developing and testing the Convolutional Neural Network (CNN) model for financial market prediction. The methodology is divided into several components: data acquisition and preprocessing, model development, evaluation and backtesting, and visualization and reporting. Each component is described in detail, highlighting the tools, libraries, and techniques used throughout the development process.

2.1 Data Acquisition and Preprocessing

2.1.1 Data Acquisition and Scope

The initial phase of the project involved the acquisition of high-quality financial time-series data from multiple sources, including the Center for Research in Security Prices (CRSP), Kaggle, and Yahoo! Finance. These datasets provided comprehensive OHLC

(Open, High, Low, Close) data across a wide range of securities listed on major U.S. stock exchanges, covering a period from the 1990s to 2017. The choice of this timeframe was deliberate to ensure that the training data was unrelated to the backtesting period (June 2019 to June 2024), thereby minimizing the risk of overfitting.

2.1.2 Data Preprocessing and Integration

Once acquired, the data underwent a rigorous preprocessing phase to ensure quality and suitability for deep learning applications. This phase involved several steps:

- **Data Cleaning**: Handling missing values using forward and backward filling techniques, and removing outliers using statistical methods such as Z-score analysis and interquartile range (IQR) filtering.
- **Normalization**: Standardizing the scale of OHLC data to ensure consistency across the dataset, typically scaling values between 0 and 1.
- **Transformation to Image Format**: The normalized OHLC data was converted into 64x64 pixel candlestick chart images using libraries such as Pandas, PIL, and Plotly. These images were then stored as .npy files for efficient loading and processing during the model training phase.

Simplified Pseudocode for Data Preparation:

```
Read CSV data files
For each data point:
    Clean and normalize data
    Convert OHLC data to candlestick chart image
    Save image as .npy file
```

2.2 Model Development

2.2.1 CNN Model Development

The core of the methodology focused on developing a robust CNN model using PyTorch, designed to predict financial market conditions based on visual representations of OHLC data.

- **Model Architecture**: The CNN architecture was constructed with multiple layers, including convolutional layers, pooling layers, dropout layers, and fully connected layers. The architecture was optimized to capture spatial patterns in the candlestick chart images.
- **Training Process**: The model was trained using GPU acceleration (CUDA) to handle large datasets efficiently. Hyperparameters such as learning rates, batch sizes, and epochs were carefully tuned to optimize model performance.

Simplified Pseudocode for CNN Model Development:

```
Define CNN architecture with multiple layers

Initialize model parameters
For each epoch:
    Load image data
    Forward pass through the network
    Compute loss and gradients
    Update model parameters
Save trained model
```

2.3 Evaluation and Backtesting

2.3.1 Model Evaluation

The CNN model, developed as a classifier, was evaluated using standard classification metrics to determine its performance in distinguishing between various market conditions:

- **Accuracy**: Proportion of correct predictions out of the total number of predictions.
- **Precision**: Proportion of true positive predictions out of all positive predictions made by the model.
- **Recall**: Model's sensitivity to correctly identifying all actual instances of a specific market condition.
- **F1 Score**: Harmonic mean of precision and recall, balancing the trade-off between these metrics.

2.3.2 Backtesting Strategy

Following model evaluation, a comprehensive backtesting strategy was employed to assess the CNN model's performance in a simulated trading environment. The strategy was implemented over a period from June 2019 to June 2024, aligning with the model's lookback periods and following a weekly rebalancing approach.

- **Trading Signals**: The CNN model generated signals to go long, short, or hold based on its predictions.
- **Backtesting Framework**: QSTrader was used to simulate trades and evaluate the strategy's performance against a benchmark buy-and-hold strategy of the S&P 500 index.

- **Performance Metrics**: Key metrics such as cumulative return, Sharpe ratio, maximum drawdown, and volatility were calculated to assess the effectiveness of the CNN-driven strategy.

Simplified Pseudocode for Backtesting Strategy:

```
Load trained model
For each backtesting period:
    Generate trading signals (long, short, hold)
    Execute trades in simulated environment
    Calculate performance metrics (return, Sharpe ratio, drawdown)
Compare performance to buy-and-hold benchmark
```

2.4 Visualization and Reporting

- **Rationale**: To provide a clear and comprehensive visualization of the model's performance and the results of the backtesting strategy.
- **Implementation**: Utilized Matplotlib to generate plots and charts illustrating key performance metrics and outcomes.

Simplified Pseudocode for Visualization:

```
Collect performance metrics
Generate visual plots for cumulative return, Sharpe ratio, drawdown
Display comparison graphs
Export visual reports
```

2.5 Technical Platform and Implementation Details

The implementation of the solution was conducted entirely using Python, leveraging a range of open-source libraries and frameworks to facilitate various aspects of data processing, model development, and evaluation. The development environment primarily utilized Visual Studio Code (VSCode), which provided a robust platform for writing, testing, and organizing the Python scripts into modular files. This modular approach allowed for the separation of different functional components, such as data preparation, model training, and backtesting, thereby enhancing maintainability and facilitating iterative development.

For data processing and visualization, libraries such as Pandas, PIL, Plotly, and Matplotlib were employed. Pandas was utilized for data manipulation tasks, including reading CSV files and handling missing values, while PIL and Plotly were used to generate candlestick chart images from the processed OHLC data. Matplotlib was also employed to create additional visualizations, both during the exploratory data analysis phase and for presenting the final results.

The core of the model development was implemented using PyTorch, a popular deep learning framework known for its flexibility and support for dynamic computation graphs. PyTorch, in combination with CUDA, enabled efficient training of the CNN model on GPUs, significantly accelerating the computation and allowing for more extensive hyperparameter tuning. NumPy was also utilized for numerical computations, providing efficient array operations and integration with other libraries.

For the evaluation and backtesting of the CNN model, QSTrader, an open-source framework for systematic trading strategies, was used. QSTrader facilitated the simulation of trading strategies based on the model's outputs and enabled a comprehensive analysis of the model's performance against a benchmark buy-and-hold strategy of the S&P 500 index. This integration allowed for a rigorous assessment of the model's predictive capabilities in a controlled environment, replicating real-world trading scenarios.

The development process was initially carried out within Jupyter Notebooks to allow for interactive experimentation and visualization of results. Upon achieving satisfactory performance, the code was then refactored into standalone Python scripts for a more production-ready deployment. This transition ensured that the final implementation was optimized for operational use while retaining the flexibility and modularity required for further enhancements and testing.

Overall, the choice of tools and frameworks was guided by their suitability for handling large-scale financial data and their ability to support the iterative development of deep learning models. The combination of Python's ecosystem of libraries with a modular development strategy provided a robust and scalable solution, capable of addressing the complex challenges associated with financial market prediction using CNN architectures.

3 Results

Contents

This section presents the results of the CNN models developed for financial market prediction and their subsequent evaluation through backtesting. Two models are highlighted: one with poor training performance and another with better training outcomes. The results include both the model training phases and their respective backtesting performances using QSTrader.

3.1 Model Training Results

3.1.1 Poorly Performing CNN Model

The first CNN model displayed suboptimal performance during the training phase. As shown in Figure 3.1, this model struggled to accurately identify market trends and predominantly generated short signals, failing to capture upward market movements. This resulted in an inability to effectively predict bullish conditions.

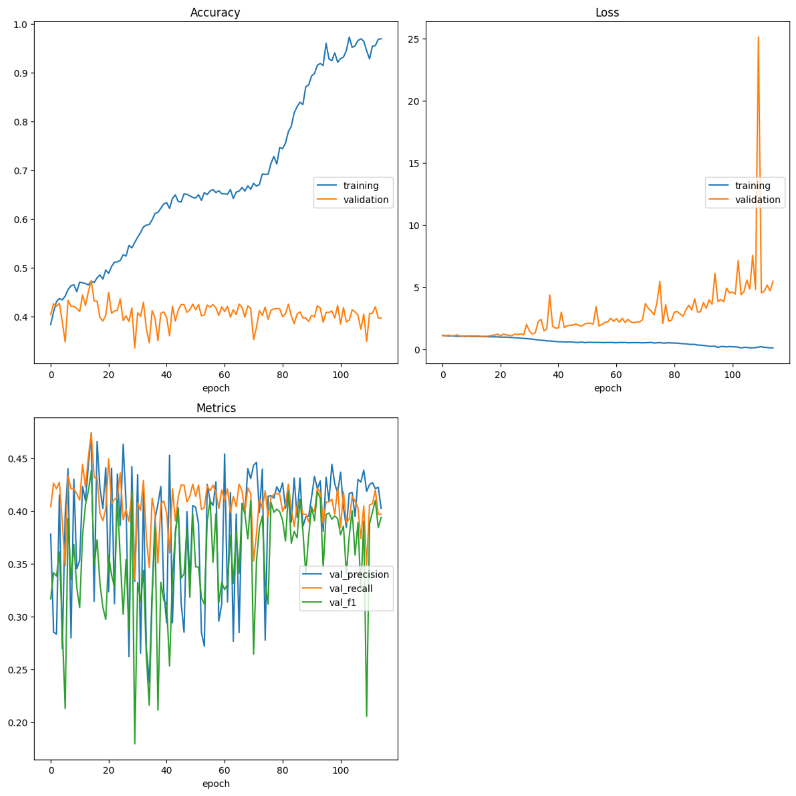


Figure 3.1: Training performance of the poorly performing CNN model, predominantly generating short signals.

3.1.2 Well-Performing CNN Model

In contrast, the second CNN model demonstrated significantly better training performance. As depicted in Figure 3.2, this model was able to effectively learn from the training data, identifying both long and short signals with greater accuracy. The balanced signal generation indicates a more refined model capable of capturing a range of market conditions.

3.2 Backtesting Results

3.2.1 Backtest of Poorly Performing CNN Model

The backtest results for the poorly performing model, as shown in Figure 3.3, reflect its training deficiencies. The model's inability to accurately generate long signals resulted in a predominantly short-biased strategy, which underperformed during bullish market conditions. The backtest metrics indicate a low Sharpe ratio, high maximum drawdown, and negative CAGR, highlighting the model's poor risk-adjusted performance.

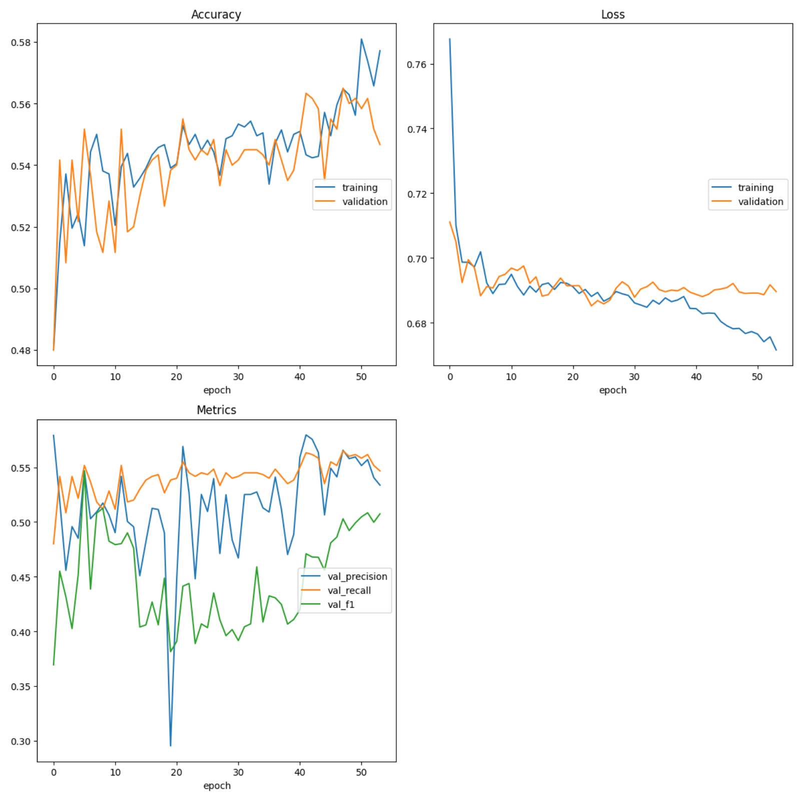
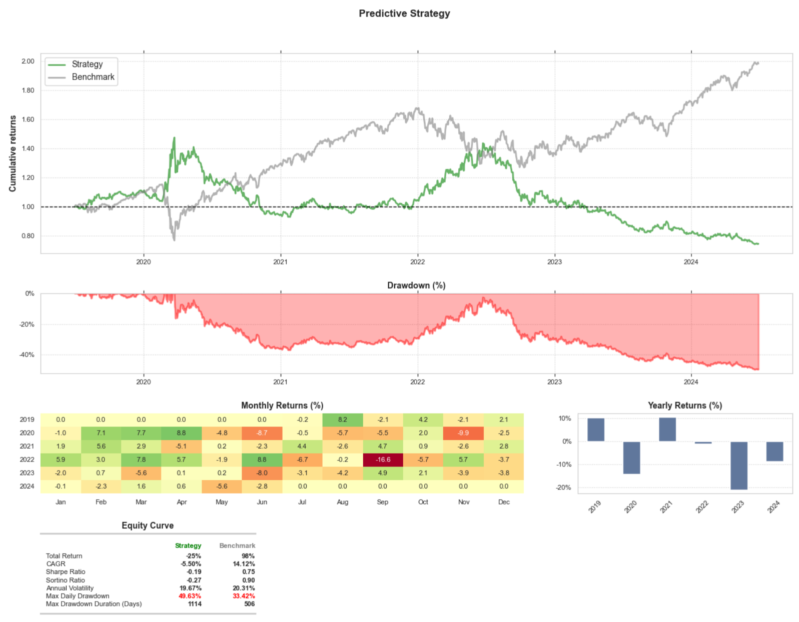


Figure 3.2: Training performance of the well-performing CNN model, showing a balanced understanding of both long and short signals.



Monthly Returns (%)

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2019	0.0	0.0	0.0	0.0	0.0	0.0	-0.2	8.2	-2.1	4.2	-2.1	2.1
2020	-1.0	7.1	7.7	8.8	-4.8	-8.7	-0.5	-5.7	-5.5	2.0	-9.9	-2.5
2021	1.9	5.6	2.9	-5.1	0.2	-2.3	4.4	-2.6	4.7	0.9	-2.6	2.8
2022	5.9	3.0	7.8	5.7	-1.9	8.8	-6.7	-0.2	-16.6	-5.7	5.7	-3.7
2023	-2.0	0.7	-5.6	0.1	0.2	-8.0	-3.1	-4.2	4.9	2.1	-3.9	-3.8
2024	-0.1	-2.3	1.6	0.6	-5.6	-2.8	0.0	0.0	0.0	0.0	0.0	0.0

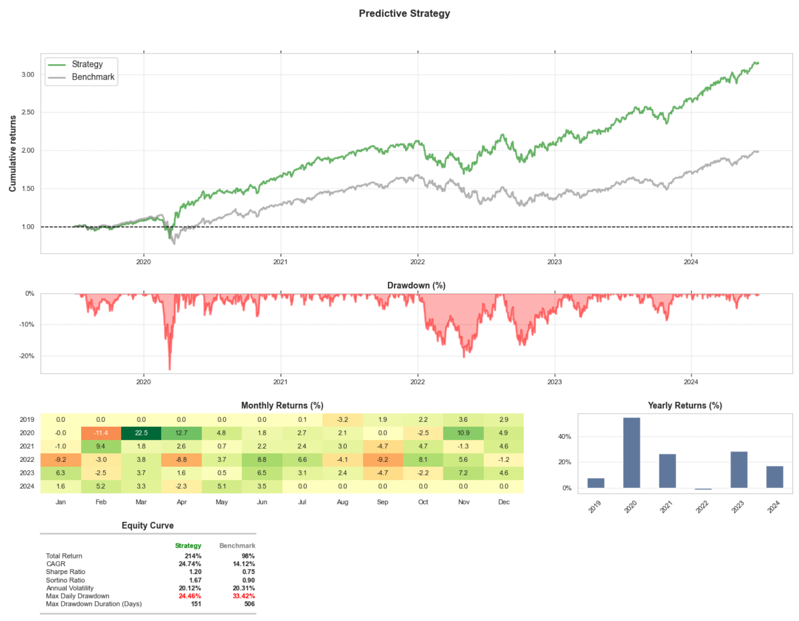
Equity Curve

	Strategy	Benchmark
Total Return	-25%	98%
CAGR	-5.50%	14.12%
Sharpe Ratio	-0.19	0.75
Sortino Ratio	-0.27	0.90
Annual Volatility	19.67%	20.31%
Max Daily Drawdown	49.63%	33.42%
Max Drawdown Duration (Days)	1114	506

Figure 3.3: Backtest results for the poorly performing CNN model, showing a failure to generate effective long signals.

3.2.2 Backtest of Well-Performing CNN Model

Figure 3.4 presents the backtest results for the well-performing CNN model. This model effectively generated both long and short signals, resulting in a more balanced and adaptive trading strategy. The backtest metrics demonstrate a higher Sharpe ratio, lower drawdown, positive CAGR, and reduced volatility, indicating superior risk management and return optimization compared to the poorly performing model.



Monthly Returns (%)

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2019	0.0	0.0	0.0	0.0	0.0	0.0	0.1	-3.2	1.9	2.2	3.6	2.9
2020	-0.0	-11.4	22.5	12.7	4.8	1.8	2.7	2.1	0.0	-2.5	10.9	4.9
2021	-1.0	9.4	1.8	2.6	0.7	2.2	2.4	3.0	-4.7	4.7	-1.3	4.6
2022	-9.2	-3.0	3.8	-8.8	3.7	8.8	6.6	-4.1	-9.2	8.1	5.6	-1.2
2023	6.3	-2.5	3.7	1.6	0.5	6.5	3.1	2.4	-4.7	-2.2	7.2	4.6
2024	1.6	5.2	3.3	-2.3	5.1	3.5	0.0	0.0	0.0	0.0	0.0	0.0

Equity Curve

	Strategy	Benchmark
Total Return	214%	98%
CAGR	24.74%	14.12%
Sharpe Ratio	1.20	0.75
Sortino Ratio	1.67	0.90
Annual Volatility	20.12%	20.31%
Max Daily Drawdown	24.46%	33.42%
Max Drawdown Duration (Days)	151	506

Figure 3.4: Backtest results for the well-performing CNN model, demonstrating effective signal generation and improved portfolio performance.

3.3 Performance Analysis

Table 3.1 summarizes the key performance metrics for both models based on their backtesting results. These metrics, including drawdown, Sharpe ratio, CAGR, and volatility, provide a comparative analysis of the two models' effectiveness in predicting market movements and managing risk.

Table 3.1: Performance Metrics for Trained CNN Models

Metric	Poorly Performing Model	Well-Performing Model
Drawdown (%)	49.63	24.46
Sharpe Ratio	0.19	1.20
CAGR (%)	-5.50	24.74
Volatility (%)	19.67	20.12

4 Discussion

Contents

This section provides an in-depth analysis of the results obtained from developing and evaluating the Convolutional Neural Network (CNN) model for financial market prediction. The discussion is structured to address the challenges encountered during implementation, the strengths and limitations of the proposed solution, and a quantitative analysis of the model's performance based on the experimental findings.

4.1 Challenges in Implementation

The implementation of the CNN model for financial forecasting posed several significant challenges. One of the most difficult tasks was ensuring that the data used for training was both balanced and representative of various market conditions. Initially, the model struggled with an imbalance in the dataset, leading to a bias in signal prediction, predominantly favoring short signals over long ones. This imbalance necessitated a comprehensive data preprocessing phase where the dataset had to be carefully curated to ensure equal representation of all classifier classes (long, short, and hold). The challenge was further compounded by the limited availability of high-quality financial data, which constrained the volume of data available for training. This limitation required careful selection and preparation of the dataset to ensure it was sufficient to train a robust model without introducing bias.

Another substantial challenge was managing overfitting during the model training phase. The initial model configurations, particularly those with more than 500 million parameters, tended to overfit the training data, especially when the dataset contained outliers or was not sufficiently diverse. Overfitting was mitigated by incorporating techniques such as leaky ReLU activation functions and dropout layers, which helped prevent the model from becoming too tightly fitted to the training data. However, some degree of overfitting remained, suggesting that further refinement of the model architecture or additional data preprocessing techniques could be beneficial.

The backtesting phase presented its own set of challenges. Limited documentation and examples for QSTrader required a deep dive into the framework's source code to adapt it for the specific needs of this project. Setting up a reliable backtesting environment that accurately reflected real-world trading conditions was crucial for validating the model's performance. Despite these challenges, the backtesting framework ultimately provided valuable insights into the model's practical applicability.

4.2 Strengths of the Solution

Despite the challenges, the solution demonstrated several key strengths that contributed to its overall success. A pivotal strength was the strategic decision to expand the dataset beyond the S&P 500 index to include a more extensive range of randomly selected U.S. stocks. This expansion significantly enhanced the model's ability to learn from a diverse set of market conditions, allowing it to generalize more effectively. The decision to use data from 30 random stocks, after experimenting with smaller datasets, proved to be optimal. The model's performance plateaued with more than 30 stocks, indicating that this dataset size was sufficient to capture the necessary market dynamics without introducing redundancy.

Another strength was the flexibility provided by the use of a JSON configuration file for model design. This approach allowed for rapid experimentation with different CNN architectures and hyperparameters, facilitating an efficient iterative development process. The ability to easily modify the model architecture—by adding or removing layers such as convolutional or LSTM layers—enabled the exploration of various configurations, ultimately leading to a more robust model. This flexibility was particularly beneficial in identifying a model structure that exceeded a 50% accuracy threshold, a critical benchmark for moving forward with backtesting.

The model's ability to identify economic outliers, such as the COVID-19 pandemic period and the market instability in 2022, highlights another strength. The model's recognition of these outlier events, despite being trained on data up to 2017, indicates a strong capacity to generalize from historical data to unprecedented market conditions. This capability is crucial for financial forecasting, where the ability to anticipate rare but impactful events can significantly affect portfolio performance.

4.3 Limitations of the Solution

While the solution demonstrated significant strengths, it also had notable limitations that need to be addressed. A primary limitation was the model's initial bias towards generating only long or short signals, a problem rooted in the dataset's imbalance. Although balancing the data resolved this issue to some extent, it also reduced the amount of data available for training, potentially limiting the model's ability to learn from a broader range of scenarios. This trade-off highlights a fundamental challenge in machine learning: balancing the need for diverse, representative training data with the need to maintain sufficient dataset size for robust model training.

The tendency of the model to overfit, particularly when using a highly complex architecture, also presented a significant limitation. While the introduction of dropout layers and leaky ReLU activations helped mitigate this issue, the model still exhibited signs of overfitting, particularly when exposed to outlier data points. This suggests that further refinement is needed, either through more advanced regularization techniques or by improving the quality and diversity of the training data.

Another limitation was the model's initial approach to handling multiple asset classes, including equities, currencies, and bonds. This approach proved to be overly ambitious, given the model's architecture and the data limitations. The model was unable to effectively capture the diverse information needed to make accurate predictions across multiple asset classes, resulting in suboptimal performance. Simplifying the model to focus solely on randomly selected U.S. stocks led to better results, but this approach also reduced the model's applicability to broader market contexts.

4.4 Quantitative Findings

The quantitative findings of this study provide a clear comparison between the well-performing and poorly performing models. The well-trained model, optimized using data from 30 randomly selected U.S. stocks, achieved a Sharpe ratio of 1.20 and maintained a drawdown of 24.46%. These metrics indicate a significantly better risk-adjusted return compared to the poorly performing model, which recorded a Sharpe ratio of 0.19 and a drawdown of 49.63%. The well-performing model also exhibited a positive Compound Annual Growth Rate (CAGR) of 24.74%, in contrast to the negative CAGR of -5.50% for the poorly performing model, underscoring its effectiveness in leveraging market opportunities while minimizing potential losses.

Despite the similar levels of volatility between the two models, the marked difference in their CAGRs and drawdowns underscores the importance of effective model training and careful data selection in building robust financial forecasting tools. The well-trained model's capacity to perform effectively during periods of market instability, such as the COVID-19 pandemic, further highlights its robustness and adaptability to varying market conditions.

Additionally, the analysis of trade distribution and mean returns provides further insight into the model's trading strategy. As shown in Figure 4.1, the distribution of trades with their corresponding mean returns and standard deviations indicates a positive mean return across most months, suggesting that the model was generally effective in identifying profitable trading opportunities. This consistency in generating positive returns, even with a degree of variability, demonstrates the model's potential reliability in real-world trading scenarios.

Overall, the quantitative results highlight both the strengths and limitations of the proposed CNN model for financial forecasting. The well-trained model's ability to maintain a positive risk-adjusted performance and its capacity to adapt to unstable market periods provide a strong foundation for further refinement and development.

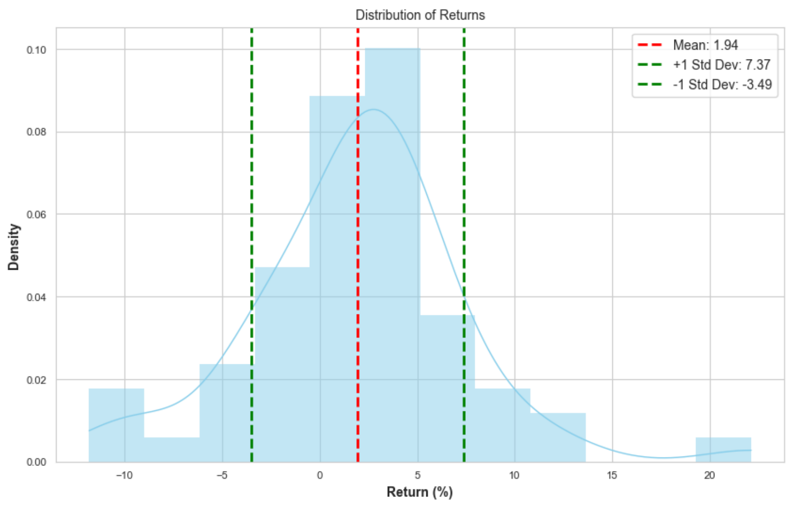


Figure 4.1: Distribution of trades with mean returns and standard deviations, showing a generally positive mean return across most months.

5 Conclussion

Contents

5.1 Future Work

Building on the findings of this study, several directions for future work can be explored to enhance the performance and applicability of the CNN model for financial market prediction. One promising avenue is the integration of more advanced deep learning architectures, such as Transformers, which have recently gained prominence in natural language processing and sequence modeling. Leveraging the advanced computational capabilities of Transformers could potentially improve the model's ability to capture temporal dependencies and complex patterns in financial time series data. This approach may enhance the model's accuracy and robustness, particularly in identifying market anomalies and outlier events.

Another area for improvement involves the use of ensemble methods that combine different model architectures, such as combining time series forecasting models with the existing CNN classifier. This ensemble approach could leverage the strengths of both models, potentially leading to improved predictive performance in both backtesting and forward testing scenarios. Additionally, incorporating more diverse data sources, such as trading volume, macroeconomic indicators, and sentiment analysis from news and social media, could further enrich the feature set available to the model, enabling it to capture a broader range of market signals and conditions. This idea aligns with findings in prior research, such as those by Bryan Kelly, which demonstrated the positive impact of incorporating additional data features on model performance.

Expanding the application of the model to other asset classes, such as currencies, commodities, and bonds, represents another promising direction. Given the model's demonstrated ability to identify significant market moves in U.S. equities, applying it to other markets could provide valuable insights and further validate its robustness and adaptability. Additionally, exploring methods to effectively filter outliers from the dataset prior to training could help reduce overfitting and improve the model's generalization capabilities.

Finally, enhancing the backtesting framework to include more sophisticated strategies, such as dynamic rebalancing and risk management techniques, could provide a more comprehensive evaluation of the model's real-world applicability. Incorporating forward testing with live market data would also help validate the model's performance under actual trading conditions, further bridging the gap between theoretical research and practical application.

5.2 Conclusion

This study developed and evaluated a Convolutional Neural Network (CNN) model for predicting financial market movements, leveraging time-series data from U.S. stocks. The research demonstrated that CNN models could provide valuable insights into market dynamics and outperform traditional financial forecasting methods under certain conditions. By expanding the dataset beyond the S&P 500 to include randomly selected U.S. stocks, the model achieved a more comprehensive understanding of diverse market conditions, ultimately leading to a well-performing model with a Sharpe ratio of 1.20 and a drawdown of 24.46%.

The implementation challenges, particularly around data balancing and overfitting, were addressed through careful data preprocessing and the introduction of regularization techniques such as dropout layers and leaky ReLU activations. The use of a JSON configuration file for model setup allowed for flexible experimentation with different architectures, leading to an optimized model configuration that was effective in backtesting scenarios. The model's ability to identify market outliers, such as the COVID-19 pandemic, further demonstrated its robustness and adaptability.

However, the study also highlighted several limitations, including the tendency to overfit with complex models and challenges in generalizing across multiple asset classes. These findings suggest that while CNNs hold significant promise for financial forecasting, further research is needed to refine these models and explore new architectures and data sources.

In conclusion, this research contributes to the growing body of literature on the application of deep learning techniques to financial markets. The findings underscore the potential of CNNs to enhance predictive accuracy and provide a foundation for more informed investment strategies. Future work focusing on model refinement, the integration of additional data features, and expansion to other asset classes will further enhance the utility and effectiveness of these models in real-world trading environments.

Bibliography

[1] R. M. I. Kusuma, T.-T. Ho, W.-C. Kao, Y.-Y. Ou, and K.-L. Hua, *Using deep learning neural networks and candlestick chart representation to predict stock market*, 2019. arXiv: 1903.12258 [q-fin.GN].

[2] O. B. Sezer and A. M. Ozbayoglu, *Financial trading model with stock bar chart image time series with deep convolutional neural networks*, 2019. arXiv: 1903.04610 [cs.LG].

[3] Z. Zeng, T. Balch, and M. Veloso, *Deep video prediction for time series forecasting*, 2021. arXiv: 2102.12061 [cs.CV].

[4] J. Jiang, B. Kelly, and D. Xiu, "(re-)imag(in)ing price trends," *The Journal of Finance*, vol. 78, no. 6, pp. 3193–3249, 2023. DOI: 10.1111/jofi.13268. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/jofi.13268. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/jofi.13268.