Prediction of Stock Prices Using Time-Frequency Analysis and Convolutional Neural Networks

COMP 4971C SPRING 2024

Hong Kong University of Science and Technology

Supervised by: Dr.David Rossiter

Muhammad Umer Farooq

Abstract

This study explores a novel approach for predicting stock prices using time-frequency analysis and Convolutional Neural Networks (CNNs). By converting stock price data into spectrograms via Short-Time Fourier Transform (STFT), the study aims to uncover patterns that improve prediction accuracy. Two datasets were used: a diverse set of short-term stock data and long-term data from the S&P 500 index. Results show that short-term, multi-stock data provide better predictive power than long-term single-index data, highlighting the importance of high-resolution spectrograms and granular data for enhancing predictive accuracy. The paper suggests future research should focus on these aspects to further refine stock price prediction models.

Contents

1 Introduction

This research delves into a novel technique for predicting stock prices by utilizing time-frequency analysis of stock data via spectrograms and Convolutional Neural Networks (CNNs) [1]. The study investigates the viability of this approach by testing it on two distinct datasets, each differing in sources and timeframes. By transforming stock price data into spectrograms through the Short-Time Fourier Transform (STFT), the aim is to identify patterns that can potentially enhance prediction accuracy [2].

Two datasets were used in this study: one consisting of diverse, short-term data from multiple stocks and another comprising long-term data from a single index [3]. The results indicated that the dataset with diverse, short-term data provided better predictive power than the long-term data from the single index. This finding suggests that leveraging data from multiple stocks over shorter, more recent periods is more effective for stock price prediction [4].

The study also underscores the importance of using higher-resolution spectrograms and more granular data to improve predictive accuracy in stock price forecasting. Future research should focus on these aspects to enhance the effectiveness of this technique [2].

The paper is structured as follows: Section 2 provides the technical background on CNNs and STFT. Section 3 describes the datasets used. Section 4 details the methodology, including the architecture of the CNN models and the performance metrics. Section 5 presents the results and discusses their implications. Finally, Section 6 concludes the report and suggests future research directions.

2 Technical Background

2.1 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a powerful class of deep learning algorithms that have revolutionized image recognition and classification tasks. They are particularly well-suited for processing data with a grid-like topology, such as images, by automatically learning spatial hierarchies of features from the input data [5, **?**].

CNNs consist of several types of layers, each serving a specific function in the network:

- **Convolutional Layers**: These layers are the cornerstone of CNNs. They apply a set of learnable filters (also known as kernels) to the input data to produce a set of feature maps. Each filter convolves across the width and height of the input, performing a dot product between the filter entries and the input at each spatial position. This operation can be expressed as:

  $$(I * K)(i, j) = \sum_m \sum_n I(m, n) \cdot K(i - m, j - n)$$

  where I is the input image, K is the filter, and (i, j) denotes the spatial position in the output feature map. Convolutional layers are responsible for detecting local patterns such as edges, textures, and shapes [5].

- **Activation Functions**: After the convolution operation, an activation function is applied to introduce non-linearity into the model. The most commonly used activation function is the Rectified Linear Unit (ReLU), defined as:

  $$\text{ReLU}(x) = \max(0, x)$$

  ReLU helps the network to learn complex patterns by breaking linearity [**?**].

- **Pooling Layers**: Pooling layers, also known as downsampling layers, follow the convolutional layers. Their primary function is to reduce the dimensionality of the feature maps while preserving important information. This process helps in reducing the computational load and makes the detection of features invariant to small translations. Max pooling, which selects the maximum value in each sub-region of the feature map, is commonly used and can be expressed as:

  $$\text{MaxPooling}(x, y) = \max_{i,j} \text{Region}_{i,j}(x, y)$$

  Pooling layers help in achieving spatial invariance [5].

- **Fully Connected Layers**: These layers are typically located at the end of the CNN and are responsible for making the final predictions. Each neuron in a fully connected layer is connected to every neuron in the previous layer, allowing the model to combine the features extracted by the convolutional and pooling layers to make predictions.

CNNs have been successfully applied to a wide range of applications beyond image recognition, including natural language processing, video analysis, and more recently, financial time series analysis, such as stock price prediction [6].

2.2 Short-Time Fourier Transform (STFT)

The Short-Time Fourier Transform (STFT) is a versatile tool used for analyzing the frequency characteristics of non-stationary signals, which are signals whose frequency content changes over time. The STFT operates by dividing a longer time signal into shorter segments of equal length and then computing the Fourier Transform separately on each segment. This results in a sequence of time-localized frequency spectra known as spectrograms.

First of all, the original time series is mapped into the time-frequency domain. STFT provides the time-localized frequency information for situations in which frequency components of a signal vary over time, whereas the standard Fourier transform provides the frequency information averaged over the entire signal time interval. Financial time series is non-stationary signals, thus STFT is used in time-frequency representation, which can be expressed as

$$X_t(m,k) = \sum_{n=0}^{T-1} x_{n+t-T+1} w(n-mR) e^{j2\pi \frac{k}{M} n}$$

where x_n is a data point of the time series $[x_{t-r+1}, \cdots, x_t]$ and $X_t(m,k)$ is the time-localized frequency information, $0 \le m \le T/R$, $0 \le k \le M/2$. $w(n)$ is a Hann window with length of M. R is the amount of shift for $w(n)$ called hop size. To improve the computational efficiency, STFT is implemented by calculating as the fast Fourier transform (FFT) of a series of windowed signal in which the window slides over times.

The spectrogram of time series which is the time-frequency power spectrum can be expressed as

$$S_t(m,k) = \log(|X_t(m,k)|^2)$$

In the context of stock price prediction, the STFT is used to transform time-series stock price data into a time-frequency representation [1]. This transformation enables the identification of patterns and features in both the time and frequency domains that may not be apparent in the raw time-series data. By converting stock prices into spectrograms, this approach leverages the strengths of CNNs in image recognition to analyze financial data.

The combination of STFT and CNNs provides a powerful framework for capturing the complex, non-linear relationships inherent in financial time series data, offering a new avenue for improving the accuracy of stock price predictions.

3 Materials and Methods

3.1 Data Description

The data utilized for training the models encompasses two distinct datasets, each offering unique insights into stock price dynamics:

3.1.1 Rand-Data:

The Rand-Data dataset comprises 2-minute interval data spanning across a period of 60 days for a diverse selection of 6928 randomly chosen stocks. This dataset was meticulously curated utilizing the Yfinance API for Python, ensuring comprehensive coverage and accuracy. Following rigorous filtering procedures aimed at addressing data gaps and inconsistencies, approximately 20,000 datapoints were retained for analysis.

Each datapoint within the Rand-Data dataset encapsulates vital information, including the 2-minute interval closing prices for the preceding 15 working days (X Value) and the corresponding percentage change in stock price after 5 working days (Y Value). This granular level of temporal resolution enables the models to capture transient fluctuations and subtle trends inherent in the stock price movements, thereby facilitating robust predictive analytics.

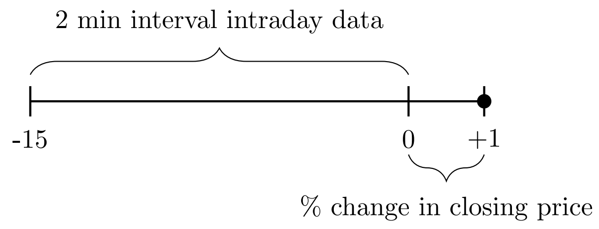

Figure 1: Rand Data set Datapoint description

3.1.2 S&P-Data:

In contrast, the S&P-Data dataset offers a comprehensive perspective by focusing on the renowned S&P 500 index. Spanning from 2008 to 2024, this dataset encompasses 1-minute interval closing prices. Given the vastness of the dataset, 50,000 datapoints were randomly sampled to ensure computational tractability while retaining representative insights.

Each datapoint within the S&P-Data dataset comprises closing prices recorded over the preceding 381*5 minutes (X Value), encapsulating a broader time window compared to the Rand-Data. Additionally, the dataset includes the percentage change in stock price after 381 days (Y Value), offering insights into longer-term market dynamics and investment trends.

In summary, the Rand-Data and S&P-Data datasets offer complementary perspectives on stock

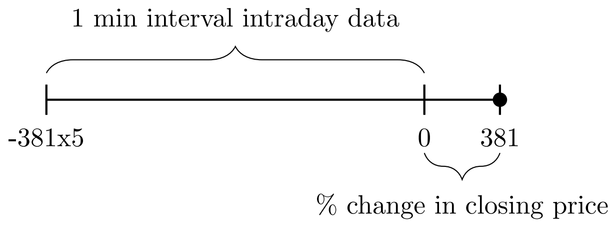

Figure 2: S&P Data set datapoint description

price dynamics, encompassing varying temporal resolutions and market scopes. By leveraging these rich and diverse datasets, the models are poised to uncover latent patterns and insights essential for accurate stock price prediction and strategic decision-making in financial markets.

3.2 Data Processing

The time-frequency analysis was conducted using the Short-Time Fourier Transform (STFT). The STFT is particularly effective in providing time-localized frequency information for signals whose frequency components vary over time. This approach was adopted as it aligns with the methodology employed in the original research paper upon which this work is based.

Originally, the spectrogram resolution was 640x480 pixels. However, due to resource constraints, the resolution was reduced to 320x240 pixels. The following Python function illustrates the process of converting data into spectrograms using STFT:

```python
def data_to_Stft(Name):
    detrended = signal.detrend(X_Val[Name])
    f, t, Sxx = signal.stft(detrended, fs=1, nperseg=512, nfft=1024)
    plt.pcolormesh(t, f, abs(Sxx), shading='gouraud')
    plt.ylim(0,0.01)
    plt.axis('off')
    plt.subplots_adjust(bottom=0)
    plt.subplots_adjust(top=1)
    plt.subplots_adjust(right=1)
    plt.subplots_adjust(left=0)
    plt.savefig(str(Name + 1) + '.png', bbox_inches='tight', pad_inches=0)
    if Name % 100 == 0:
        print(f"{Name}/{len(X_Val)}")
```

Listing 1: Code used to create spectrograms

The function `data_to_Stft` takes an index as an input and processes the corresponding data entry. The steps include:

1. **Detrending**: The signal is detrended to remove linear trends using `signal.detrend`.
2. **STFT Calculation**: The STFT of the detrended signal is computed using `signal.stft`, with a sampling frequency (`fs`) of 1, a segment length (`nperseg`) of 512, and a Fourier transform length (`nfft`) of 1024.
3. **Spectrogram Creation**: A spectrogram is generated with `plt.pcolormesh`, setting frequency limits and adjusting plot aesthetics to ensure the resulting image is clear and devoid of axes or margins.
4. **Saving the Image**: The spectrogram is saved as a PNG file, named incrementally based on the input index.
5. **Progress Indication**: Every 100 entries, the function prints the progress to the console.

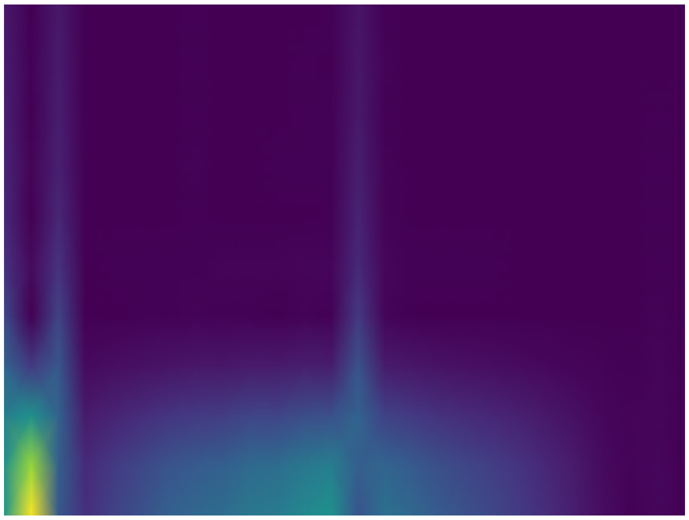

Figure 3: Example of a spectogram

4 Methodology

4.1 Architecture

Various Convolutional Neural Network (CNN) architectures were meticulously evaluated throughout the course of this research endeavor. Among these, the most promising and effective model, utilized across both datasets, comprised a sophisticated arrangement of convolutional and pooling layers, followed by densely connected layers. This subsection elucidates the architectural configuration of the CNN model employed for stock price prediction.

4.1.1 Model Architecture Overview

The CNN architecture employed in this study follows a sequential arrangement of layers, each contributing to the extraction of relevant features from the spectrogram representations of stock price data. The model architecture can be summarized as follows:

1. **Input Layer**: The initial entry point for the data, representing the spectrograms generated from the stock price data.
2. **Convolutional Layers**: A series of convolutional layers, denoted as `conv2d`, responsible for extracting spatial features from the input spectrograms.
3. **Max Pooling Layers**: Following each convolutional layer, max pooling layers, indicated as `maxpooling`, are employed to downsample the feature maps, enhancing computational efficiency and reducing overfitting.
4. **Flatten Layer**: Subsequent to the final max pooling layer, a flatten layer is introduced to transform the multidimensional feature maps into a vector format suitable for input to the fully connected layers.

5. **Fully Connected Layers**: These densely connected layers, denoted as `dense`, leverage the extracted features to make predictions regarding the future percentage change in stock prices.
6. **Output Layer**: The concluding layer of the model, responsible for producing the final prediction outputs.

4.1.2 Model Architecture Diagram

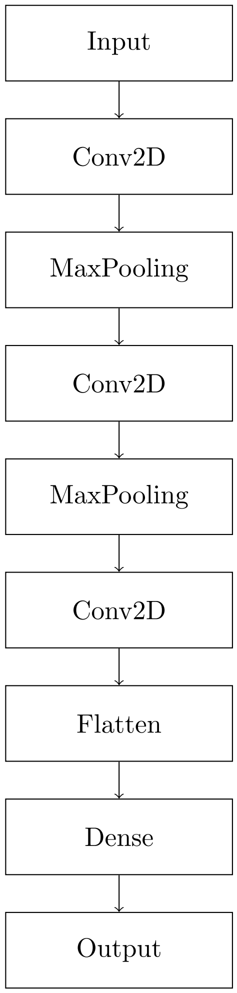

Figure 4: Architecture of the CNN Model

4.2 Model Performance Metrics

Performance was assessed using several metrics:

- **Mean Squared Error (MSE)**: This metric measures the average of the squares of the errors, providing a sense of how close the predictions are to the actual values. It is defined as:

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

  where Y_i is the actual value and $\hat{Y}_i$ is the predicted value.

- **Mean Absolute Error (MAE)**: This metric measures the average absolute differences between the predicted and actual values, providing a more interpretable measure of prediction accuracy. It is defined as:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i|$$

- **Percentage Sign Accuracy**: This metric evaluates the model's ability to correctly predict the direction (positive or negative) of stock price changes. It is calculated as the percentage of predictions where the predicted sign matches the actual sign:

$$\text{Percentage Sign Accuracy} = \frac{\text{Number of Correct Sign Predictions}}{\text{Total Number of Predictions}} \times 100\%$$

5 Results and Discussion

5.1 Performance

The models were trained for 10 epochs with a batch size of 32 across both datasets. The results were as follows:

- **Rand-Data**:
  - MAE (Validation set): 5.1644
  - Correct Sign Percentage (Validation set): 52.81%
- **S&P-Data**:
  - MAE (Validation set): 0.79
  - Correct Sign Percentage (Validation set): 17.19%

The results indicate that while the S&P-Data model achieved a lower MAE, it was less effective at predicting the direction of price changes compared to the Rand-Data model. This suggests that diverse, short-term data from multiple stocks may have higher predictive power.

6 Conclusion and Recommendations

6.1 Discussion of Results

Although the models produced under this research yielded disappointing results in terms of prediction accuracy, they bring to light significant insights regarding the nature of the data used for training. Notably, while the S&P dataset resulted in a model with a lower mean absolute error (MAE), it performed significantly worse in predicting the direction (sign) of stock price changes compared to the Rand dataset. This suggests that the Rand dataset possesses a higher predictive power for determining stock price trends than the S&P dataset.

These findings highlight an important consideration when working with stock data: it appears more effective to utilize data collected from multiple stocks over shorter, recent periods rather than data from a single stock over an extended period. This approach leverages the diversity and more current relevance of the data, potentially capturing more robust and generalized patterns within the market.

Moreover, the results hint at an intriguing phenomenon: different stocks might share common features in their time-frequency analysis when observed over the same period. This similarity could be harnessed to improve predictive models by focusing on a broader spectrum of stocks within a consistent timeframe.

The observed disparity in the MAE between the two datasets can also be attributed to the difference in data granularity. The Rand dataset uses data at 2-minute intervals, whereas the S&P dataset uses 1-minute interval data. The coarser granularity of the Rand data may help to smooth out noise and highlight more significant trends, thus enhancing its predictive utility.

In summary, these results underscore the value of diversified, recent data from multiple stocks and suggest that such an approach may yield better predictive models in stock price forecasting. Future research could further explore the implications of these findings by examining a wider array of stocks and refining the granularity of data collection. “‘

This version elaborates on the initial points and provides a more structured discussion of the implications of the findings.

6.2 Limitations and Future Work

During the course of this research, several oversights and limitations were encountered that impacted the results. One of the primary limitations was the granularity of the data. While the Rand dataset used data at 2-minute intervals, future research should aim to collect data at 1-minute intervals from a diverse set of randomly selected stocks. This finer granularity may lead to improved model performance by capturing more detailed market dynamics.

Additionally, the spectrograms used for training the models were scaled down to half their original resolution (from 640x480 pixels to 320x240 pixels) due to resource constraints. Future studies should strive to utilize spectrograms at their full resolution to potentially enhance the feature extraction process and overall model accuracy.

Despite these limitations, this research provides valuable and somewhat counter-intuitive insights into stock data prediction. It suggests that leveraging data from multiple stocks over shorter, recent periods can be more effective than using long-term data from a single stock. This insight contributes to the broader understanding of how to approach training predictive models for stock prices and may guide future research efforts in the field.

In summary, while this study faced several challenges, it highlights important considerations for future work. Enhancing data granularity and using higher-resolution spectrograms are crucial steps for improving model performance. The findings also underscore the potential benefits of using diversified, short-term data from multiple stocks, offering a new perspective on stock price prediction methodologies.

References

[1] D. Jia, Q. Gao, and H. Deng, "Stock market prediction based on time-frequency analysis and convolutional neural network," *Journal of Physics: Conference Series*, vol. 2224, p. 012017, apr 2022.

[2] Y. L. A. S. Wenjie Lu, Jiazheng Li and J. Wang, "A cnn-lstm-based model to forecast stock prices," *Complexity*, vol. 2020, p. 6622927, 2020.

[3] P. Khodaee, A. Esfahanipour, and H. Mehtari Taheri, "Forecasting turning points in stock price by applying a novel hybrid cnn-lstm-resnet model fed by 2d segmented images," *Engineering Applications of Artificial Intelligence*, vol. 116, p. 105464, 11 2022.

[4] J. Cao and J. Wang, "Stock price forecasting model based on modified convolution neural network and financial time series analysis," *International Journal of Communication Systems*, vol. 32, p. e3987, 08 2019.

[5] G. H. Yann LeCun, Yoshua Bengio, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[6] T. Kim and H. Y. Kim, "Forecasting stock prices with a feature fusion lstm-cnn model using different representations of the same data," *PLOS ONE*, vol. 14, pp. 1–23, 02 2019.