Rotman

GROUP Assignment Cover Sheet

Course Code:	RSM 8521	**Student Numbers**: *Please list all student numbers included in this group assignment.*	1005936490 1007787968 1005828749 1010746356 1009969205
Course Title:	Leveraging AI and Deep Learning Tools in Marketing		
Instructor Name:	Kanchana Padmanabhan		
Assignment Title:	Final Project		
Date:	2024/04/21		

Using CNN Models to Capture and Predict Stock Trends from Candlestick Charts

1. Introduction

Candlestick charts, with their rich visual representation of price action, have long been employed by traders to gauge market sentiment and forecast potential price trends. The patterns formed by candlesticks contain valuable information about the dynamics between buyers and sellers, offering insights into market psychology and momentum.

However, we often find different market participants have totally different interpretations of the same candlestick chart. Moreover, manual interpretation of these charts is not only subject to human biases and limitations but also time-consuming given the fast-moving nature of the capital markets.

Fortunately, the surge in computational power and progression in machine learning / artificial intelligence techniques has enabled us to explore automated methods to extract meaningful signals from candlestick charts. By leveraging Convolutional Neural Networks (CNNs) to conduct candlestick chart analysis, market participants would possibly gain a competitive edge in navigating volatile markets. Moreover, the potential impacts extend beyond individuals to the whole market, because the market mispricing will be eliminated more swiftly and accurately, fostering a more efficient market.

In light of these considerations, this report aims to explore the feasibility and effectiveness of employing CNNs for candlestick chart recognition in predicting stock price trends. We found that despite some inherent limitations of computational resources, the CNN models are useful for predicting stock price trends by extracting features from candlestick charts.

2. Background

2.1 Stock Price / Trend Prediction

Market participants (investors, traders, etc) conduct stock price prediction in different ways. Some conduct fundamental analysis, looking at a company's financial information and make their predictions, while others employ technical analysis, looking at price charts like candlestick charts, trying to predict price movements. Stock trend prediction is important on two levels: for individual market participants, investors and traders always want their buying, selling, or holding decisions to be as correct as possible, thereby maximizing profits or minimizing losses; for the whole market as a whole, the stock price predictions will drive price movements by triggering investors' buy or sell activities and "help" the stock to move towards fair value.

2.2 Technical Analysis

In capital markets with frequent and public trading activities, like stock markets, analysts and traders often apply technical analysis charts to discover potential trading opportunities. Unlike fundamental analysis, which focuses on assessing a company's financial information and financial metrics, technical analysis primarily relies on the analysis of price patterns and market psychology. Technical analysts believe that human psychology remains largely consistent when faced with similar market situations. Therefore, technical patterns are developed from historical price charts, thus enabling investors to make predictions of future price movements and in turn make trading profits. Those patterns have been given different names, such as the following "Head and Shoulder Top" pattern, which indicates price decrease.

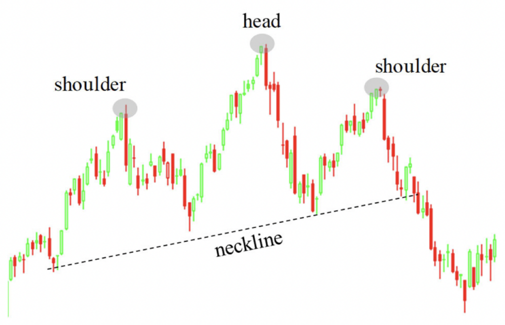


Chart: Head and Shoulder Top

2.3 Candlesticks

Candlestick charts, originating from Japanese rice merchants in the 18th century, have become a cornerstone of technical analysis due to their ability to visually represent price action in a concise and informative manner. Each candlestick has four key price points: the opening price, the closing price, the highest price (high), and the lowest price (low) within a specific time frame. The body of the candlestick was colored differently (in western context, green is bullish and red is bearish, while in some Asian markets it's the opposite) to indicate whether the closing price was higher or lower than the opening price. Meanwhile, the lines above and below the body represent the range between the highest and lowest prices during the time frame. Candlestick charts are widely used in technical analysis due to their intuitive visual representation.

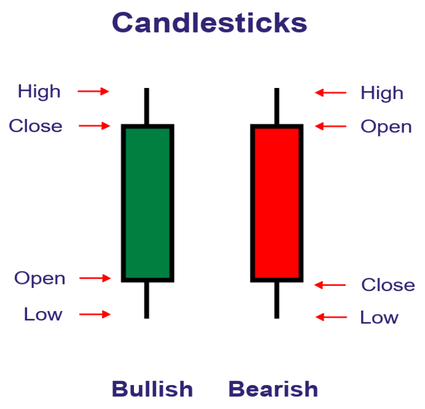


Chart: Candlestick Chart Interpretation

2.4 CNN and Stock Trend Analysis:

In this report, we aim to apply CNN to candlestick graphs to predict stock prices. While CNNs are traditionally associated with image processing tasks, their application to time series data has proven to be effective. In time series analysis, CNNs can extract meaningful features from sequential data like candlestick patterns. Firstly, CNNs are capable of learning hierarchical representation, which means CNN can learn to detect patterns at different levels. For example, for candlestick charts, CNN can learn low-level features such as individual shapes, mid-level features such as combinations of candlesticks, and high-level features such as patterns across multiple periods or with other market indicators. Secondly, CNNs are highly parameter-efficient because of weight-sharing. It can learn from large volumes of data without an excessive number of parameters. In time series prediction, this efficiency is crucial for handling the vast amount of historical data in financial markets.

3. Data Preparation

Step 1 - Collect raw data:

Data: Historical **stock price** and **trading volume** data of companies listed on S&P 500 index from 2010-01-01 to 2019-12-31

Attributes:

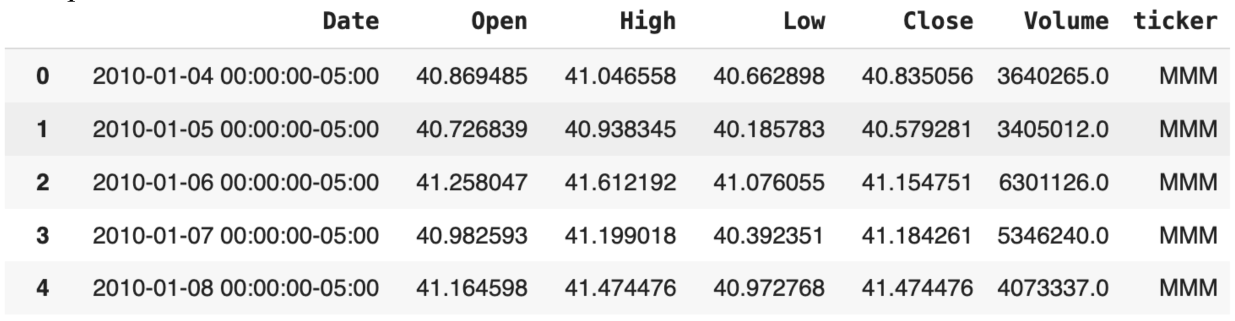
- Transaction Date
- Ticker (Identifier of a stock)
- Open Price (Stock price when the market opens in the morning)
- Close Price (Stock price when the market closes in the evening)
- Low (Lowest price during the day)
- High (Highest price during the day)
- Volume (Trading volume of the stock during the day)

Supplementary Info:

- Sector in which the company operates.
- Date when the company went public and added to the S&P 500 index.

Data Source: ***yfinance*** (Yahoo Finance) library in Python

Example of data and attributes:



	Date	Open	High	Low	Close	Volume	ticker
0	2010-01-04 00:00:00-05:00	40.869485	41.046558	40.662898	40.835056	3640265.0	MMM
1	2010-01-05 00:00:00-05:00	40.726839	40.938345	40.185783	40.579281	3405012.0	MMM
2	2010-01-06 00:00:00-05:00	41.258047	41.612192	41.076055	41.154751	6301126.0	MMM
3	2010-01-07 00:00:00-05:00	40.982593	41.199018	40.392351	41.184261	5346240.0	MMM
4	2010-01-08 00:00:00-05:00	41.164598	41.474476	40.972768	41.474476	4073337.0	MMM

Dimension of the dataframe: 1,178,469 rows × 7 columns

- 500 companies (tickers) * 10 years * 250 trading days each year ~ 1,250,000 rows
- Reasons for missing data:
  - Some companies were not public until later in the 2010s.
  - Some companies quit the index in the middle of the observation period.
  - Other data recording issues.

Step 2 - Convert raw numerical data into standardized candlestick charts:

- Target picture resolution: 448 * 336 (4:3) with RGB.
- Target picture size: < 50KB. This size will ensure the photo contains sufficient information while not occupying too large memory space.

Library used: ***Matplotlib.pyplot***.

We use a 20-day window to filter out a continuous trading period of a stock, and plot one candlestick for each day. The color of the stick is given as **red** when close_price > open price, and **green** when open_price > close price. The shape of the stick is determined by the values of open, close, daily high and low. Stock price relevant information is encoded in both the color and the shape of the candlestick. We then scale the 20 candlesticks into one chart and output the file.

Alternative method: Beside the candlestick chart, we also considered the trading volume chart which is conventionally placed below the candlesticks. As demonstrated by the experiment, this does not help improve the model performance. Therefore, we chose not to include the volume chart.

Step 3 - Label each chart and store them in designated directory:

The immediate 5 days after the 20-day observation period (X) is considered as the prediction period (y). We give each chart a target label based on the stock performance in the 5 days:

- If 5-day close price > 5-day open * 1.01: **Buy (0)**
- If 5-day close price < 5-day open * 0.99: **Sell (1)**
- Else: **Hold (2)**

As each chart is given a label, our problem is defined as a multi (3)-class classification problem with supervised learning. The threshold **1.01** is carefully chosen so that:

- The resulting proportion of 3 target labels is approximately balanced (⅓ each)
- The annualized return of 1.01 is 65%, which is a reasonable return that affects the investor's buy-sell-hold decision.

The labeled charts are stored in three different directories. Later in the data loader stage, we take and shuffle charts from the storage and feed them into the model for training and testing.

Step 4 - DataLoader

We build a data loader to iterate through the file storage and prepare batches (batch size = **32**) of training/testing images for the model. Several modifications are made to the data loader so that it fits the requirement of the model input:

- Resize the shape of the photo to **224 * 224** and scale the color value by **255** as required by ResNet and other pretrained models.
- We did not add image processing such as vertical/horizontal flip and rotation, because candlestick charts will always be standardized in real-world application.
- The data loader is designed to catch problematic images and skip it whenever the model cannot read the image during training, as a few images (<0.1%) are unidentifiable by the model for unclear reasons.

4. Experiments

Upon initial preprocessing and generation of candlestick charts, we obtained approximately 60,000 PNG files. This substantial amount posed a significant challenge due to our computational limitations and the need for time efficiency. Hence, in line with our methodological index, we utilized a subset of 12,000 PNG files for the experiments to maintain a balance between computational feasibility and data diversity.

Consistent with the preprocessing section, the selected data samples were categorized into three classes: 0 for "buy", 1 for "sell", and 2 for "hold". We partitioned these labeled images into training and testing datasets, with an 80-20 split, to establish a robust validation framework for our models. During the initial phase of our experiments, we encountered several technical challenges. The most prominent was the slow upload speed experienced when using Google Colab, which significantly delayed syncing our image data to Google Drive folders. Given the large data volume of approximately 12,000 PNG files, this posed a bottleneck in our workflow.

To handle this issue, we execute the train-test split locally using Jupyter notebooks, thereby avoiding the upload bottleneck and allowing for continuous progress in our project. While we awaited the image data synchronization between Google Colab and Google Drive, we proceeded with our model training and testing locally to ensure that our project timelines were adhered to.

Then we kept going in our modeling part. We have tested 3 models, each model underwent training and validation phases, demonstrating varying capabilities in handling the classification task.

Model 1: ResNet50-Based Architecture

Architecture:

- **Base Model**: We used ResNet50, a pre-trained model known for its depth and residual learning feature, to handle complex image recognition tasks. The model was configured without the top layer (**include_top=False**) to allow for custom layers suited to our specific classification task.
- **Custom Layers**: A series of Dense layers were added on top of the base ResNet50 output. This includes a Global Average Pooling 2D layer to reduce spatial dimensions, followed by three Dense layers with ReLU activation to capture nonlinear dependencies, and finally a softmax layer for classifying into three categories: Buy, Sell, Hold.

Training and Validation:

- **Freezing Layers**: The layers of the ResNet50 base model were frozen to retain the learned features and only the custom layers were trained.
- **Compilation**: The model was compiled with Adam optimizer and categorical cross-entropy as the loss function, focusing on accuracy as the metric.
- **Training**: We trained the model for 10 epochs using the training data generator and validated using the test data generator.

Model 2: VGG16-Based Architecture

Architecture:

- **Base Model**: VGG16 was chosen for its simplicity and effectiveness in image classification tasks. Similar to the first model, the top layers were removed to customize the output layers.
- **Custom Layers**: The custom layers are identical to those in the ResNet50 model, aiming to provide a robust feature extraction and classification capability.

Training and Validation:

- **Freezing Layers**: Like with ResNet50, the pre-trained layers of VGG16 were frozen to leverage the pre-learned patterns.
- **Compilation and Training**: The model was compiled and trained similarly to the first model, using Adam optimizer and the same loss and metric.

Model 3: Custom CNN Model

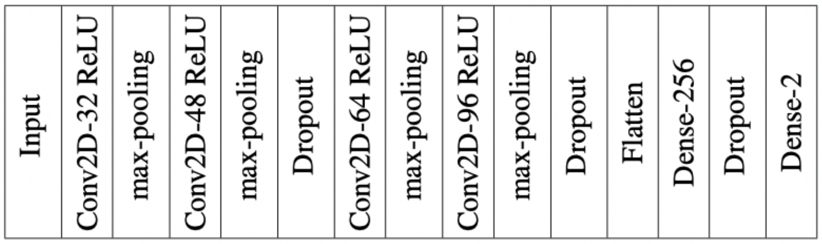


Chart: CNN layer architecture

To validate the ordinary CNN architecture versus the pre-trained models, we built a simple CNN model from scratch. The layers are configured in a way (see above) that has been proven to be effective by a paper[1] (Kusuma et al. 2019) which tests the CNN on Taiwan50 stock dataset. It contains several convolution layers, max pooling layers, dropout layers along with fully-connected layers at the output.

[1] https://arxiv.org/pdf/1903.12258.pdf

The model trains faster than the pre-trained models as there are less parameters in this model. The classifications results from this model are later compared to the pre-trained ones.

Performance Metrics: For all models, the primary performance metric was accuracy, which measures the proportion of correctly predicted images against the total number of predictions made. This metric is crucial for evaluating the effectiveness of each model in correctly classifying the candlestick chart images into Buy, Sell, or Hold categories.

- Results:

Model	Class	Precision	Recall	F1-Score
Customized CNN	Buy (0)	0.73	0.65	0.69
	Sell (1)	0.41	0.44	0.42
	Hold (2)	0.47	0.45	0.46
VGG 16	Buy (0)	0.68	0.73	0.70
	Sell (1)	0.5	0.43	0.46
	Hold (2)	0.56	0.61	0.58
ResNet 50	Buy (0)	0.79	0.75	0.77
	Sell (1)	0.63	0.7	0.66
	Hold (2)	0.58	0.55	0.56

While all 3 models provide a similar outcome of around 60% accuracy, the model built on ResNet50 demonstrates the optimal result. Notably, all models tend to perform better on identifying the "Buy" class, which conforms to our expectation that bullish patterns are more easily recognized. Hyperparameter tuning such as learning rates (1e-4) and batch sizes (32) are proven to have little effect on model performance.

During the training process, we also noticed that the pre-trained models had a small loss (<10) since the first epoch, while the CNN from scratch had a convergence process (where loss decreased from over 100 to <10) that takes at least 5 epochs. That means the pre-trained models have an advantage due to the image patterns it had learnt before. Nevertheless, it is possible that the customized CNN model can minimize the loss with more epochs (which we were unable to verify due to the computational power constraint).

5. Limitation

In our study, we recognized that there are several limitations, such as accessibility of data, computational and time resources, and assumptions behind our model. Those limitations help us clarify the application boundaries of the CNN model and serve as areas of improvement to explore further, if given additional resources.

5.1 Challenges of accessing enough volume of data

Despite the substantial size of our dataset, comprising over 1.1 million rows across 500 companies listed on the S&P 500 index from 2010 to 2019, certain factors contribute to data gaps. The model might underperform in circumstances such as the Financial Crisis or the COVID pandemic. The complexity of the stock market, including fluctuating market dynamics and corporate events, underscores the inherent difficulty for us when obtaining a complete dataset.

5.2 Limitations on computation power

The computational requirements for training CNNs can be substantial. Limited computational resources impose constraints on the size of the model we can use, compromising our CNN's predictive performance. Additionally, longer training times hinder our exploration of different model architectures and hyperparameters.

5.3 Nuances of applying the CNN model to different sectors / stock markets

The dynamics of stock markets vary significantly across different sectors. Currently, we are using the S&P 500 Index as our data for building the CNN. Focusing solely on the S&P 500 Index may overlook important patterns specific to other sectors or stock markets. Therefore, when applying this model, we should be aware of the potential inaccurate prediction result of generalizing our findings to another sector without considering the market's differences.

5.4 Not considering the Random Walk Model of stock price / macroeconomic factors

The assumption of a random walk model for stock prices, where future prices are independent of past prices, challenges the effectiveness of our CNN model, which predicts future stock prices based solely on historical data. Moreover, macroeconomic factors such as interest rates and inflation can influence stock prices, which are not incorporated into our model, potentially limiting predictive accuracy.

5.5 Potential conflicts with the Efficient Market Hypothesis

The efficient market hypothesis says that stock prices reflect all available information, making it difficult to outperform the market consistently. While technical analysis methods, like candlestick patterns as we mentioned, aim to identify inefficiencies in the market, the efficient market hypothesis suggests that any predictable patterns may be quickly arbitraged away, limiting the effectiveness of CNN predictions.

6. Conclusion

This report shows that using Convolutional Neural Networks (CNNs), especially pre-trained image classification models like ResNet50, to analyze candlestick charts for predicting stock trends works well. These models are good at picking up patterns from the data and predict stock prices effectively. Even though we faced some challenges with handling a large amount of data and the need for high computing power, these models proved to be promising as they cater to the need for speed in the stock market. In the future, we may explore other ways such as summing the occurrence of bearish and bullish patterns before reaching a final decision. Overall, using this technology in the stock market helps make trading fairer and more efficient by reducing human mistakes and biases in predicting stock prices.

Rotman

References

[1]https://towardsdatascience.com/identifying-candlestick-patterns-using-deep-learning-b7d706726874

[2]https://www.kaggle.com/code/farshidhossain68/candle-stick-full-project-accuracy-0-9542#Image-Load

[3]https://www.researchgate.net/publication/326764676_Stock_Chart_Pattern_recognition_with_Deep_Learning

[4]https://arxiv.org/pdf/1903.12258.pdf

[5]https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9092995

[6]https://cs231n.stanford.edu/reports/2022/pdfs/157.pdf

[7]https://www.researchgate.net/publication/376685209_Enhancing_Financial_Chart_Analysis_Advanced_Detection_of_Candlestick_Patterns_Using_Deep_Learning_Models_for_Mastering_Trend_Recognition

[8]https://www.strike.money/technical-analysis/types-of-candlesticks-patterns