STOCK PRICE MOVEMENT DETECTION USING CANDLESTICK PATTERNS

A Major Project Report Submitted

In partial fulfillment of the requirement for the award of the degree of

Bachelor of Technology
in
Computer Science and Engineering (Internet Of Things)

by

KATTA SHIVANI 20N31A6927
L.SREENIVAS REDDY 20N31A6933
Y.SUDHARSHAN REDDY 20N31A6959

Under the Guidance of

Dr. M. GAYATRI

PROFESSOR

Department of Emerging Technologies

MRCET (Autonomous Institution, UGC Govt. of India)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING-INTERNET OF THINGS

(EMERGING TECHNOLOGIES)

MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY

(Autonomous Institution – UGC, Govt. of India)

(Affiliated to JNTU, Hyderabad, Approved by AICTE, Accredited by NBA & NAAC – 'A' Grade, ISO 9001:2015 Certified)

Maisammaguda (v), Near Dullapally, Via: Kompally, Hyderabad – 500 100, Telangana State, India

2023-2024

DECLARATION

We here by declare that the project entitled **"STOCK PRICE MOVEMENT DETECTION USING CANDLESTICK PATTERNS"** submitted to **Malla Reddy College of Engineering and Technology UGC Autonomous Insitution,** affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH) as part of IV Year B.Tech – II Semester and for the partial fulfillment of the requirement for the award of **Bachelor of Technology** in **Computer Science and Engineering (Internet of Things)** is a result of original research work done by us.

It is further declared that the project report or any part thereof has not been previously submitted to any University or Institute for the award of degree or diploma.

Katta Shivani (20N31A6927)
L. Sreenivas Reddy (20N31A6933)
Y. Sudharshan Reddy (20N31A6959)

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY
(Autonomous Institution – UGC, Govt. of India)
(Sponsored by CMR Educational Society)
Recognized under 2(f) and 12 (B) of UGC ACT 1956
Estd : 2004
(Affiliated to JNTUH,Hyderabad, Approved by AICTE- Accredited by NBA & NAAC– 'A' Grade - ISO 9001:2015 Certified)

CERTIFICATE

This is to certify that this is the bonafide record of the project titled **" STOCK PRICE MOVEMENT DETECTION USING CANDLESTICK PATTERNS"** submitted by **KATTA SHIVANI, L. SREENIVAS REDDY, Y. SUDHARSHAN REDDY** bearing roll no **20N31A6927, 20N31A6933, 20N31A6959** of **B.Tech IV Year – II Semester** in the partial fulfillment of the requirements for the degree of **Bachelor of Technology** in **Computer Science and Engineering (Internet of Things)**, Dept. of CSE (Emerging Technologies) during the year 2023-2024. The results embodied in this project report have not been submitted to any other university or institute for the award of any degree or diploma.

Project Guide
Department of CSE (ET)

Project Coordinator
Department of CSE (ET)

EXTERNAL EXAMINER

HEAD
OF THE DEPARTMENT

Date of Viva-Voce Examination held on:_____

ACKNOWLEDGEMENTS

We feel our self-honored and privileged to place our warm salutation to our college Malla Reddy College of Engineering and Technology (Autonomous Institution – UGC Govt. of India) and our Principal **Dr. S Srinivasa Rao,** Professor who gave us the opportunity to do the Major Project during our IV Year B.Tech II Semester and profound the technical skills.

We express our heartiest thanks to our Director **Dr. V S K Reddy,** Professor for encouraging us in every aspect of our project and helping us realize our full potential.

We also thankful to our Head of the Department **Dr. M V Kamal,** Professor for providing training and guidance, excellent infrastructure and a nice atmosphere for completing this project successfully.

We would like to express our sincere gratitude and indebtedness to our project supervisor **Dr. M. GAYATRI,** Professor for her valuable suggestions and interest throughout the course of this project.

We convey our heartfelt thanks to our Project Coordinator **Dr. P Dileep,** Professor for allowing for their regular guidance and constant encouragement during our dissertation work.

We would like to thank all our supporting **staff** of the Department of CSE (Emerging Technologies) and even all other department who have been helpful directly and in-directly in making our project a success.

Finally, We would like to take this opportunity to thank our **families** for their support and blessings for completion of our project that gave us the strength to do our project.

KATTA SHIVANI (20N31A6927)

L. SREENIVAS REDDY (20N31A6933)

Y. SUDHARSHAN REDDY (20N31A6959)

ABSTRACT

Stock Market forecasting is a knotty challenging task due to the highly noisy, trigonometric, complex and chaotic nature of the stock price time series. Since stock prices vary dramatically, it is important to determine when to buy and sell stocks in order to get high returns from stock investment.

Predicting the direction of speak price movements is a challenging task, as the stock market is influenced by a wide range of complex and interconnected factors. The aim of stock prediction is to effectively predict future stock market trends, which can lead to increased profit. One major stock representation of the price action over a set period of time. In this research paper, we propose using a convolution neural network (CNN) machine learning model, implemented with TensorFlow and Keras, to predict the direction of stock price movements using candlestick chart data.

The use of a CNN model allows for the analysis of the visual patterns present in the candlestick charts, which may not be easily discernible to the human eye. However, implementing a machine learning model for this task is not without its challenges. Some potential difficulties that may arise include: Limited data availability, Complexity of stock market data, Overfitting. Short-term fluctuations.

Overall, using a CNN machine learning model with TensorFlow and Keras to predict the directions of stock price movements using candlestick chart data.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Introduction

Investment has become an integral part of our daily lives, and many people have become accustomed to using their personal assets to generate more profit. In general, conservative investors place their excess cash in banks, funds, or the stock market, and expect to obtain returns on interest and dividends. In contrast, active investors prefer to use their funds to buy/sell stocks, funds, futures, options, and so on, expecting to get higher returns. Although stock investment can bring high returns for investors, the stock market is affected by macroeconomic and international factors, stock market volatility, as well as governmental policies that combine to make it very difficult to achieve an assured level of accurate stock prediction. The main idea behind stock prediction is to analyze the available information from the stock market in order to effectively predict future trends (stock prices), which can increase profit. Stock trend prediction techniques play a crucial role to bring more people into market and encourage markets as a whole.

There are two primary methods used to predict stock price movements and support investment decisions, i.e., fundamental analysis and technical analysis.

Fundamental analysis involves analyzing a company's financial data to determine the fair value of the company, and to forecast future stock value. Because of this analyzing process, most investors believe that fundamental analysis is mainly suitable for long-term prediction. Technical analysis is based on the assumption that history repeats itself. It attempts to predict future stock price movements based on an examination of past stock price movements. One of the important types of technical analysis is candlestick chart patterns, known as "candlestick charting." The candlestick chart patterns usually consisting of only a few consecutive candlesticks, providing short-term predictions for traders. Dozens of candlestick chart patterns are identified to be signals of bullish/bearish reversals and continuations.

CANDLESTICK PATTERNS

A technical methodology known as a candlestick chart condenses data from many time periods into a single price bar. They are therefore more beneficial than conventional open, high, low, close (OHLC) bars or straightforward lines that link closing price dots. Candlesticks create patterns that, once finished, may be used to forecast price movement.

To spot probable trend reversals or continuations in the price movement of a financial asset, such as stocks, currencies, or commodities, technical analysts frequently employ candlestick patterns.

Candlestick charts (also known as candlestick graphs) are a type of financial chart used to display the price movements of a security, currency, or commodity over a period of time. It looks like a candlestick with a vertical rectangle and a wick at the top and bottom. The top and bottom of the candlestick show open and closed prices.

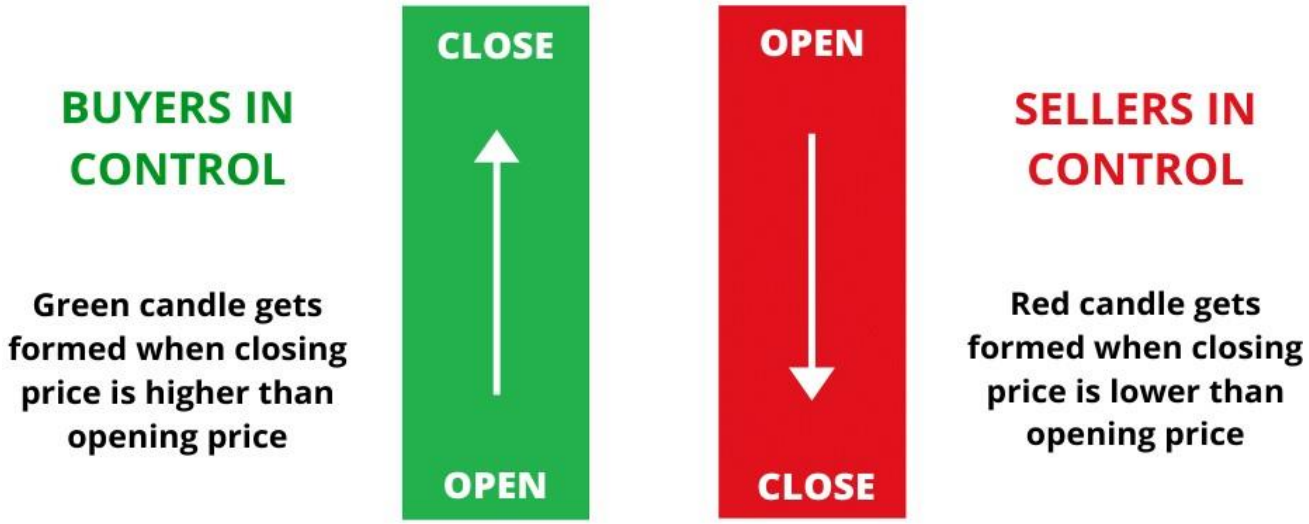


Fig 1.1 Candlesticks

Many Candlestick patterns exist. Following are a few typical candlestick patterns:

BULLISH ENGULFING PATTERN

A little red candlestick is followed by a larger green candlestick that totally engulfs the body of the previous candle, forming a bullish engulfing pattern. It implies that the sellers have been outnumbered by the buyers and that the price may rise further

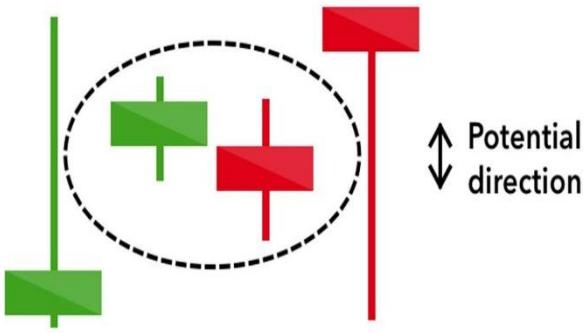
Fig 1.2 Bearish Engulfing Pattern

Fig 1.3 Bearish Engulfing Pattern

BEARISH ENGULFING PATTERN

It happens when a tiny green candlestick is followed by a bigger red candlestick that totally engulfs the body of the first candle. It implies that the sellers have outnumbered the purchasers and that the price could drop further.

DOJI



When the opening and closing prices are external near to one other or the same, a doji candlestick pattern results. It implies that the market is unsteady and that a trend reversal might happen.

Fig 1.4 The Doji Pattern

HAMMER

This pattern develops when the price starts off around the day's high, falls off dramatically over the day, and then rises back up to close to the high. It implies that buyers have beaten out sellers, and the price maygo up much further.

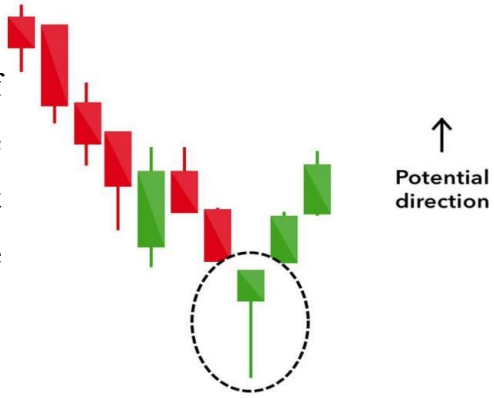


Fig 1.5 Hammer

Fig 1.6 Shooting Star

SHOOTING STAR

This pattern develops when the price starts off close to the day's high before falling sharply throughout the day and closing close to the day's low. It implies that sellers have gained the upper hand over buyers and that the price may drop further.

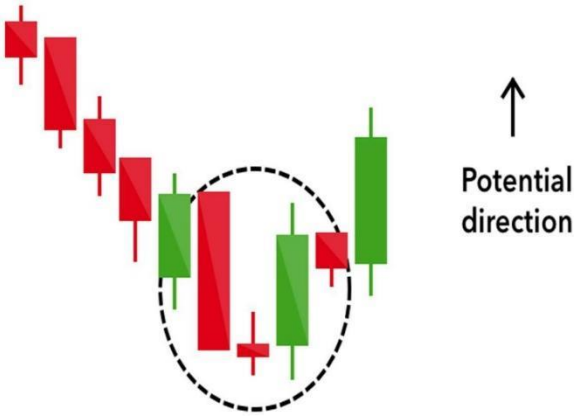


Fig 1.7 Morning Star

MORNING STAR

This pattern appears when a long red candlestick is followed by a small candlestick that gaps down (either red green), followed by a long green candlestick. It implies the possibility of a bullish to bearish trend reversal.

Fig 1.8 Evening Star

EVENING STAR

The Morning Star pattern's antithesis is the Evening Star pattern. It developsfrom a long green candlestick, a short gaping red orgreen candlestick, and a long red candlestick. It implies the possibility of a positive trend turning bearish.

1.2 Motivation

Detecting stock price movement using candlestick patterns is motivated by the desire to gain insights into market sentiment and potential price direction. Candlestick patterns offer visual representations of price action, reflecting the dynamics between buyers and sellers in the market. By analyzing these patterns, traders seek to identify potential trends, reversals, and trading opportunities.

The motivation lies in the ability to make informed trading decisions based on historical price patterns and market psychology. Candlestick patterns encapsulate market sentiment and provide valuable signals about future price movements, helping traders anticipate market trends and mitigate risks. Additionally, automated detection of candlestick patterns through software enables efficient analysis of vast amounts of data in real-time, allowing traders to react swiftly to changing market conditions.

Ultimately, the motivation for stock price movement detection using candlestick patterns is to enhance trading strategies, optimize risk management, and potentially achieve better investment returns. Traders and investors aim to capitalize on the predictive power of candlestick patterns to navigate the complexities of financial markets and achieve their financial objectives.

1.3 Literature Review

1. Technical Analysis of Stock Trends by Robert D. Edwards and John Magee (1977)

This classic text provides foundational knowledge on technical analysis, including candlestick patterns and their significance in predicting stock price movements.

Advantages:

Comprehensive Coverage: "Technical Analysis of Stock Trends" offers a comprehensive overview of technical analysis techniques, covering various aspects such as chart patterns, trend analysis, and candlestick patterns. Readers can gain a deep understanding of these concepts from the ground up.

Timeless Principles: Despite being published in 1977, many of the principles discussed in the book remain relevant today. The book's focus on price action, trends, and market psychology transcends time, providing enduring insights for traders and investors.

Disadvantages:

Limited Emphasis on Fundamental Analysis: While technical analysis is the primary focus of the book, it may not adequately address the importance of fundamental analysis in conjunction with technical analysis. Ignoring fundamental factors such as company earnings, industry trends, and macroeconomic indicators could lead to incomplete market analysis.

Complexity for Novice Traders: Some readers, particularly novice traders, may find the content of the book overwhelming due to its technical nature and complex terminology. Beginners might struggle to grasp certain concepts without prior knowledge or additional guidance.

2. Japanese Candlestick Charting Techniques by Steve Nison (1991)

Steve Nison introduced Japanese candlestick charting techniques to Western traders. This book explores various candlestick patterns and their interpretations in predicting price movements.

Advantages of Japanese Candlestick Charting Techniques:

Visual Clarity: Candlestick charts offer a visually intuitive representation of price movements, making it easier for traders to identify patterns and trends compared to traditional bar charts or line graphs.

Comprehensive Analysis: Candlestick patterns convey information about market sentiment, including bullishness, bearishness, and indecision, enabling traders to make more informed decisions about entry and exit points.

Disadvantages of Japanese Candlestick Charting Techniques:

Subjectivity: Interpreting candlestick patterns involves a degree of subjectivity, as different traders may interpret the same pattern differently, leading to inconsistencies in trading strategies and outcomes.

Limited Predictive Power: While candlestick patterns can provide valuable insights into market sentiment, they are not foolproof predictors of future price movements and may sometimes lead to false signals or misinterpretations.

3. Candlestick Charting Explained: Timeless Techniques for Trading Stocks and Futures by Gregory L. Morris (1995)

Morris provides detailed insights into candlestick patterns and their applications in stock trading. The book covers both basic and advanced candlestick patterns along with real-world examples.

Advantages:

Visual Clarity: Candlestick charts offer a visually intuitive representation of price movements, making it easier for traders to identify patterns and trends compared to traditional bar charts.

Pattern Recognition: Morris delves into various candlestick patterns, empowering traders to recognize market sentiment shifts and potential trading opportunities with greater accuracy.

Disadvantages:

Subjectivity: Interpretation of candlestick patterns can be subjective, leading to potential discrepancies in analysis among traders.

Limited Scope: While candlestick patterns are valuable tools, relying solely on them may overlook other important aspects of technical analysis, such as volume or momentum indicators.

4. Candlestick and Pivot Point Trading Triggers: Setups for Stock, Forex, and Futures Markets by John L. Person (2006)

Person discusses the integration of candlestick patterns with pivot point analysis for identifying potential trading opportunities in various markets, including stocks.

Advantages:

Clear Entry and Exit Signals : Integrating candlestick patterns with pivot point analysis provides traders with clear signals for entry and exit points, enhancing decision-making in the markets.

Versatility Across Markets : The approach can be applied to various markets such as stocks, forex, and futures, allowing traders to utilize the same methodology across different asset classes.

Disadvantages:

Subjectivity in Interpretation: Interpretation of candlestick patterns and pivot points can be subjective, leading to different traders identifying different signals from the same data.

False Signals: Like any trading strategy, there is a risk of false signals where the identified setups do not result in profitable trades, potentially leading to losses if not managed properly.

5. Predicting Stock Price Movement with Candlestick Patterns by Shih-Chuan Tsai and Jui-Kuei Chen (2010)

This research paper explores the effectiveness of candlestick patterns in predicting stock price movements. It analyzes a large dataset of historical stock prices and evaluates the predictive power of different candlestick patterns.

Advantages:

Visual Interpretation: Candlestick patterns provide a visual representation of price movements over a given period, making it easier for traders to interpret market sentiment and potential price trends at a glance.

Historical Significance: Many candlestick patterns have been observed and studied over centuries, allowing traders to leverage historical data and patterns to make informed decisions about future price movements.

Disadvantages:

Subjectivity: Interpretation of candlestick patterns can vary among traders, leading to subjective analysis and potentially conflicting conclusions about the direction of price movements.

False Signals: Candlestick patterns may sometimes produce false signals, where a predicted price movement does not materialize, leading to losses for traders who rely solely on these patterns for decision-making.

6. Novel Approach for Stock Price Movement Prediction Using Candlestick Patterns by Eman Saad, Walid Abdelmoez, and Nashwa El-Bendary (2015)

The authors propose a novel method for stock price movement prediction based on candlestick patterns. They introduce a feature selection technique to enhance prediction accuracy and validate their approach using real-world stock data.

Advantages:

Incorporation of Candlestick Patterns: Utilizing candlestick patterns provides a rich source of historical price information, allowing for nuanced analysis of market sentiment and potential price movements.

Feature Selection Technique: The introduction of a feature selection technique enhances prediction accuracy by focusing on relevant variables, potentially reducing noise and improving model performance.

Validation with Real-world Data: Validating the approach with real-world stock data lends credibility to the proposed method, demonstrating its applicability and effectiveness in practical trading scenarios.

Disadvantages:

Limited Predictive Power: While candlestick patterns offer valuable insights, their predictive power might be limited, especially in highly volatile or unpredictable markets, leading to inaccurate forecasts.

Dependency on Historical Data: Relying solely on historical data, including candlestick patterns, may overlook emerging trends or external factors not captured in the dataset, potentially reducing the model's adaptability to changing market conditions.

7. Predicting Stock Price Movement Using Candlestick Patterns and Support Vector Machines by Ming-Hui Wen and Fu-Rong Chen (2017)

This paper investigates the application of support vector machines (SVMs) in conjunction with candlestick patterns for predicting stock price movements. It compares the performance of SVMs with traditional machine learning techniques.

Advantages:

Non-linear Decision Boundaries: SVMs are capable of capturing complex non-linear relationships between input features and target variables, making them suitable for analyzing stock price movements which often exhibit non-linear patterns.

Robustness to Overfitting: SVMs are less prone to overfitting compared to some other machine learning algorithms, as they aim to maximize the margin between different classes, leading to better generalization to unseen data.

Disadvantages:

Sensitivity to Hyperparameters: SVMs require careful selection of hyperparameters such as the choice of kernel and regularization parameter. Poor selection can lead to suboptimal performance or computational inefficiency.

Limited Interpretability: SVMs provide a black-box model, making it challenging to interpret the underlying decision-making process. Understanding the relationship between input features and predictions may be difficult, especially in the context of financial markets where interpretability is crucial.

8. Stock Price Movement Prediction Using Technical Indicators and Machine Learning Algorithms" by Savita Gupta and Sangeeta Mittal (2019)

Gupta and Mittal examine the efficacy of combining various technical indicators, including candlestick patterns, with machine learning algorithms for predicting stock price movements. They conduct experiments on historical stock data to evaluate prediction accuracy.

Advantages:

Improved Prediction Accuracy: By combining multiple technical indicators and machine learning algorithms, the model can potentially capture more nuanced patterns in stock price movements, leading to higher prediction accuracy.

Robustness: Integrating diverse indicators and algorithms can enhance the robustness of the prediction model, making it less susceptible to noise or sudden market changes compared to simpler models.

Disadvantages:

Complexity and Overfitting: Incorporating numerous indicators and algorithms may lead to a complex model prone to overfitting, especially if not carefully managed or validated with proper techniques.

Computational Resources: Utilizing multiple indicators and algorithms may require significant computational resources, both in terms of processing power and time, which could limit scalability or real-time applicability.

9. A Hybrid Deep Learning Model for Stock Price Movement Prediction by Xin Zhang, Chao Zhang, and Jian Zhang (2020)

This study proposes a hybrid deep learning model that integrates convolutional neural networks (CNNs) and long short-term memory (LSTM) networks for stock price movement prediction. Candlestick patterns are among the features used in the model.

Advantages:

Enhanced Prediction Accuracy: By combining CNNs and LSTM networks, the model can capture both spatial and temporal dependencies in the data, leading to improved accuracy in predicting stock price movements.

Feature-rich Representation: Incorporating candlestick patterns as features enriches the model's representation of market dynamics, potentially leading to better performance compared to models relying solely on numerical data.

Disadvantages:

Complexity and Computational Overhead: Integrating CNNs and LSTM networks increases the model's complexity, requiring significant computational resources for training and inference, which may limit its scalability.

Interpretability Challenges: Deep learning models, especially hybrids like this one, often lack interpretability, making it difficult to understand the rationale behind specific predictions and potentially hindering their adoption in certain contexts.

10. Enhanced Stock Price Prediction Using Technical Analysis Indicators and Machine Learning by Manish Kumar and Sanjay Jasola (2021)

Kumar and Jasola present an enhanced stock price prediction framework that incorporates candlestick patterns, along with other technical analysis indicators and sentiment analysis of news articles. They assess the effectiveness of their approach using real market data.

Advantages:

Holistic Approach: Incorporating candlestick patterns, technical analysis indicators, and sentiment analysis of news articles provides a comprehensive view of market dynamics, capturing both quantitative and qualitative factors influencing stock prices.

Improved Predictive Accuracy: By integrating multiple data sources and utilizing machine learning techniques, the framework is likely to enhance predictive accuracy compared to models relying solely on historical price data or individual indicators.

Disadvantages:

Data Complexity: Integrating diverse data sources like candlestick patterns and sentiment analysis adds complexity to the model, requiring robust data preprocessing and feature engineering techniques. This complexity can increase computational costs and model training time.

Model Over fitting: The incorporation of multiple indicators and data sources may increase the risk of over fitting, particularly if the model is not properly regularized or validated. Over fitting could lead to poor generalization performance when applied to unseen data.

1.4 Problem Definition

The problem at hand is the development of a software solution capable of detecting stock price movements utilizing candlestick patterns. In financial markets, candlestick patterns offer valuable insights into market sentiment and potential price changes. However, manually identifying and interpreting these patterns can be time-consuming and prone to human error. Therefore, the aim is to create an automated system that can accurately recognize various candlestick patterns across different time frames, ranging from minutes to days. This system must efficiently process historical and real-time stock price data, apply sophisticated pattern recognition algorithms, and provide actionable insights to traders and investors. By automating the detection process, the software will empower users to make informed decisions more quickly, capitalize on trading

opportunities, and mitigate risks associated with market fluctuations. The ultimate goal is to enhance trading efficiency, improve decision-making processes, and potentially increase returns for users operating in financial markets.

1.5 Objective of the Project

The objective of the project is to develop a sophisticated software system capable of accurately detecting stock price movements by analyzing candlestick patterns. By leveraging historical and real-time stock price data, the system aims to identify key patterns such as doji, hammer, engulfing, and more across various time frames. Through advanced algorithms and pattern recognition techniques, the software will not only pinpoint the occurrence of these patterns but also assess their significance and reliability. The primary goal is to empower traders and analysts with timely insights into potential market trends and signals, enabling informed decision-making in trading strategies. By providing customization alerts, visualization tools, and integration with trading platforms, the system seeks to enhance efficiency, improve trading outcomes, and ultimately contribute to achieving greater profitability for users in the dynamic landscape of financial markets.

CHAPTER 2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM AND PROPOSED SYSTEM

EXISTING SYSTEM

- The existing systems for stock price movement detection are diverse, with various approaches and technologies used to analyze and predict stock price movements. The system collects historical and real-time stock price data from various sources such as stock exchanges, financial news websites, APIs (e.g., Yahoo Finance, Alpha Vantage), and data providers.
- Raw data is preprocessed to clean and transform it into a suitable format for analysis. Preprocessing steps may include handling missing data, removing outliers, and adjusting for stock splits and dividends.
- In this existing system there is stock price movement detection only by scatter plot, line plot,
- Histogram, heatmap, and confusion matrix.

PROPOSED SYSTEM

- A theoretically proposed system for stock price movement detection using candlestick patterns would be designed to identify and predict price movements in financial markets based on the analysis of candlestick patterns.
- In our project we developed stock price movements detection using candlestick patterns where those candlestick patterns are generated using scatter plot, line plot, histogram, heatmap and confusion matrix.
- So, the previous year stocks are all detected by candlestick patterns where all these scatter plot, line plot, histogram, heatmap and confusion matrix will be analyzed by convolution neural network and at last the pattern is generated which is very easy to the stake holders.

2.2 FUNCTIONAL REQUIREMENTS

HARDWARE:

Processor : Intel i3 and above.

RAM : 4GB and Higher.

Hard Disk : 500GB Minimum

SOFTWARE:

Operating System: Recent Versions of Windows or Mac

Python Libraries: Pandas, Keras, Matplotlib, TensorFlow.

IDE: Visual Studio

Python3

CHAPTER 3

SOFTWARE ENVIRONMENT

3.1 PYTHON

Python is a high-level, interpreted programming language renowned for its simplicity, readability, and versatility. Developed by Guido van Rossum and first released in 1991, Python has since grown into one of the most popular languages in the world, embraced by developers across various domains, from web development and data science to artificial intelligence and automation. Its design philosophy emphasizes code readability and simplicity, making it an ideal choice for beginners and experienced programmers alike.

At the core of Python's appeal is its clean and intuitive syntax, characterized by the use of indentation to denote code blocks, eliminating the need for cumbersome braces or semicolons. This readability not only enhances the ease of understanding for programmers but also facilitates collaboration and maintenance of codebases. Additionally, Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming, providing developers with the flexibility to choose the most appropriate approach for their projects.

One of Python's greatest strengths lies in its extensive standard library, which offers a rich set of modules and packages for performing a wide range of tasks without the need for external dependencies. From handling file I/O operations and working with data structures to networking, threading, and graphical user interface (GUI) development, the standard library provides developers with robust tools to streamline development workflows and accelerate project delivery. Furthermore, Python's package management system, pip, simplifies the installation and management of third-party libraries, fostering a vibrant ecosystem of community-contributed packages catering to diverse application requirements.

Python's versatility extends beyond traditional software development to encompass domains such as web development, powered by frameworks like

Django and Flask, which facilitate the rapid creation of dynamic and scalable web applications. These frameworks leverage Python's simplicity and expressiveness to enable developers to focus on building functionality rather than dealing with boilerplate code. Additionally, Python's support for asynchronous programming paradigms through libraries like asyncio enhances the performance and responsiveness of web applications by efficiently handling concurrent tasks.

In recent years, Python has emerged as a dominant force in the field of data science and machine learning, fueled by libraries such as NumPy, pandas, and scikit-learn, which provide powerful tools for data manipulation, analysis, and modeling. Python's ease of use and rich ecosystem have made it the preferred choice for data scientists and researchers, enabling them to leverage advanced algorithms and techniques to extract insights from complex datasets and build predictive models. Furthermore, the emergence of deep learning frameworks like TensorFlow and PyTorch has propelled Python to the forefront of artificial intelligence research, empowering developers to create sophisticated neural network architectures for tasks such as image recognition, natural language processing, and reinforcement learning.

Beyond traditional software development and data science, Python finds applications in automation and scripting, thanks to its ability to interface seamlessly with operating system APIs and external libraries. Whether automating repetitive tasks, scripting system administration tasks, or building command-line utilities, Python's simplicity and versatility make it an ideal choice for writing concise and expressive scripts that enhance productivity and efficiency.

Python's popularity and widespread adoption have been further accelerated by its vibrant community, which actively contributes to its development, maintains an extensive repository of documentation and tutorials, and fosters collaboration through forums, mailing lists, and online communities. This collaborative ethos not only ensures the continuous improvement and evolution of the language but also provides developers with access to a wealth of knowledge and resources to support their learning journey and solve challenges encountered during development.

Advantages of Python:

Python offers a plethora of advantages that have contributed to its widespread adoption and popularity among developers across various domains. Some of the key advantages of Python include:

1. **Simplicity and Readability:**

   Python's clean and intuitive syntax, characterized by the use of indentation to denote code blocks, enhances code readability and reduces the cognitive load on developers. This simplicity makes Python an ideal language for beginners learning to code and facilitates collaboration among team members working on shared projects.

2. **Versatility:**

   Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming, providing developers with the flexibility to choose the most appropriate approach for their projects. Its versatility enables it to be used for a wide range of applications, from web development and data science to automation and artificial intelligence.

3. **Extensive Standard Library:**

   Python comes with a comprehensive standard library that offers a rich set of modules and packages for performing various tasks, such as file I/O operations, data manipulation, networking, and GUI development. This extensive library reduces the need for external dependencies and accelerates development workflows by providing developers with robust tools out of the box.

4. **Large Ecosystem of Third-Party Packages:**

   In addition to its standard library, Python boasts a vibrant ecosystem of third-party packages contributed by the community. These packages cater to diverse application requirements, ranging from web frameworks like Django and Flask to data science libraries like NumPy, pandas, and scikit-learn. Python's package management system, pip, simplifies the installation and management of these

packages, fostering innovation and collaboration within the community.

5. **Platform Independence:**

   Python is a platform-independent language, meaning that code written in Python can run on various operating systems, including Windows, macOS, and Linux, without modification. This platform independence facilitates cross-platform development and deployment, enabling developers to build applications that can be deployed seamlessly across different environments.

6. **Ease of Learning and Accessibility:**

   Python's simplicity and readability make it an ideal language for beginners learning to code. Its gentle learning curve and extensive documentation resources enable newcomers to quickly grasp the fundamentals of programming and start building meaningful projects. Furthermore, Python's supportive community provides ample tutorials, forums, and online resources to aid learners at every stage of their journey.

7. **Community Support and Collaboration:**

   Python boasts a vibrant and supportive community of developers who actively contribute to its development, maintain extensive documentation, and provide assistance through forums, mailing lists, and online communities. This collaborative ethos fosters knowledge sharing, innovation, and the continuous improvement of the language and its ecosystem.

8. **High-Level Language Features:**

   Python offers high-level language features such as dynamic typing, automatic memory management (garbage collection), and built-in data structures (lists, dictionaries, sets) that simplify development and accelerate prototyping. These features enable developers to focus on solving problems rather than dealing with low-level implementation details.

9. **Interpreted and Interactive:**

   Python is an interpreted language, which means that code written in Python is executed line by line by the Python interpreter. This interpreted nature facilitates

rapid development and debugging, as developers can quickly test and iterate on their code without the need for compilation. Furthermore, Python's interactive mode allows developers to experiment with code snippets and explore language features in real-time, enhancing the learning experience and productivity.

10.Scalability and Performance:

While Python is often criticized for its perceived lack of performance compared to lower-level languages like C or C++, advancements in runtime implementations (e.g., CPython, PyPy) and optimization techniques have significantly improved Python's performance. Additionally, Python's ability to seamlessly integrate with performance-critical libraries written in languages like C or Fortran (e.g., NumPy, TensorFlow) allows developers to achieve near-native performance for computationally intensive tasks.

Disadvantages of Python:

While Python offers numerous advantages, it also has some disadvantages and limitations that developers should consider when choosing it as their primary programming language. Some of the notable disadvantages of Python include:

1. **Performance:**

   Python is an interpreted language, which typically results in slower execution speeds compared to compiled languages like C or C++. While Python's performance has improved over the years, especially with the introduction of optimized runtime implementations like PyPy, it may still struggle with performance-critical tasks or high-throughput applications. Additionally, Python's dynamic typing and automatic memory management (garbage collection) can introduce overhead and impact performance in certain scenarios.

2. **Global Interpreter Lock (GIL):**

   Python's Global Interpreter Lock (GIL) is a mechanism that prevents multiple native threads from executing Python bytecodes simultaneously in a single

process. This can limit Python's ability to fully utilize multicore processors for parallel execution, particularly in CPU-bound applications where multithreading would otherwise offer performance benefits. While there are workarounds such as multiprocessing or using external libraries like NumPy for parallelism, the GIL remains a notable limitation for certain types of applications.

3. **Mobile Development:**

   Python is not widely used for mobile app development compared to languages like Java or Swift, which are commonly used for Android and iOS development, respectively. While there are frameworks like Kivy or BeeWare that enable mobile development with Python, they may not offer the same level of ecosystem support, performance, or native user experience as platform-specific alternatives.

4. **Library and Tooling Fragmentation:**

   While Python boasts a vast ecosystem of third-party packages and libraries, the quality, compatibility, and documentation of these packages can vary widely. This fragmentation can lead to challenges in finding the right tools for a given task, ensuring compatibility between dependencies, and troubleshooting issues that arise from using multiple libraries together. Additionally, the fast-paced evolution of the Python ecosystem may result in deprecation or abandonment of older libraries, leading to maintenance challenges for existing projects.

5. **Runtime Overhead:**

   Python's dynamic typing and runtime interpretation incur overhead compared to statically typed, compiled languages. While this flexibility enhances developer productivity and code readability, it can result in larger memory footprint and slower startup times for Python applications. Additionally, the need to distribute Python's runtime environment (e.g., the Python interpreter) alongside the application adds to deployment complexity, especially in constrained environments such as embedded systems or containers.

6. **Community and Corporate Governance:**

Python's development is overseen by the Python Software Foundation (PSF), a non-profit organization dedicated to advancing and promoting the Python programming language. While the PSF and Python's community of developers contribute to its open-source development and maintenance, the lack of a centralized corporate sponsor or commercial entity may impact the pace of development, long-term support, and strategic direction of the language compared to languages backed by major technology companies.

7. **Compatibility and Versioning:**

Python's evolution and adoption of new language features may introduce compatibility issues between different versions of Python. The transition from Python 2 to Python 3, in particular, highlighted the challenges associated with migrating existing codebases and libraries to newer versions. While efforts have been made to encourage adoption of Python 3 and provide migration tools, legacy codebases and dependencies may still require ongoing maintenance and support for compatibility with older Python versions.

8. **Security Concerns:**

Like any popular programming language, Python is not immune to security vulnerabilities and risks. While Python's simplicity and high-level abstractions can mitigate certain types of vulnerabilities (e.g., buffer overflows), developers must remain vigilant and follow security best practices to protect against common threats such as injection attacks, cross-site scripting (XSS), and code injection vulnerabilities in third-party packages.

MACHINE LEARNING:

Machine learning (ML) is an area of artificial intelligence (AI) that enables computers to "learn" for themselves over time from training data and develop without explicit programming. Data patterns can be found by machine learning algorithms, which can then use this information to learn and develop their own predictions. Algorithms and models used for machine learning, in essence, gain experience.

In contrast, machine learning is a process that is automated and gives computers the ability to solve issues with little to no human involvement and make decisions based on prior experiences.

While machine learning and artificial intelligence are frequently used synonymously, they are actually two distinct ideas. Machine learning, a subset of AI that enables intelligent systems to autonomously learn new things from data, is what allows intelligent systems to make decisions, gain new abilities, and solve problems in a way that is similar to people. AI is the more general notion..

Machine learning methods are used in image classification to examine the existence of objects in a picture and classify it.

The foundation of computer vision and image recognition is this specific problem. Machines don't examine an image in its entirety. They just examine pixel patterns or vectors while analysing a picture. The elements will subsequently be classified and given labels in accordance with the various rules that were established during algorithm configuration.

There are two types of Image Classification techniques:

If you are facing various pictures, and you only want to know whether the object in them is a cat or not, the problem you will have to handle is a binary classification. You only need to label one class of items for all images, or not to label it. The binary classification model is in charge of computing the presence or the absence of the object.

If there is more than one category, you handle a multiclass classification problem which implies the creation of multiple labels that will correspond to various objects. The machine will then predict which single object class is contained in the photographs or pictures, among a set of predefined labels.

These techniques both mainly rely on the way the reference images are labeled.

ARCHITECTURE

This concept is similar to discussions regarding machine learning and its algorithms. For generalization and to examine its specifics, neural networks are

primarily used.

One of the most widely used machine learning techniques today is neural networks. Over time, it has been unequivocally demonstrated that neural networks perform better in accuracy and speed than alternative algorithms. With numerous variations, including CNN (Convolutional Neural Networks), RNN (Recurrent Neural Networks), Autoencoders, Deep Learning, etc., neural networks are gradually becoming what linear regression was for statisticians in the field of machine learning or data science. So, it is crucial to have a basic understanding of what a neural network is, how it functions, and its capabilities and constraints. This article aims to describe neural networks, starting with their most fundamental component, the neuron, and moving on to their most well-known versions, such as CNN, RNN, etc.

Convolutional Neural Network is being used for the development of this system.

CNN:

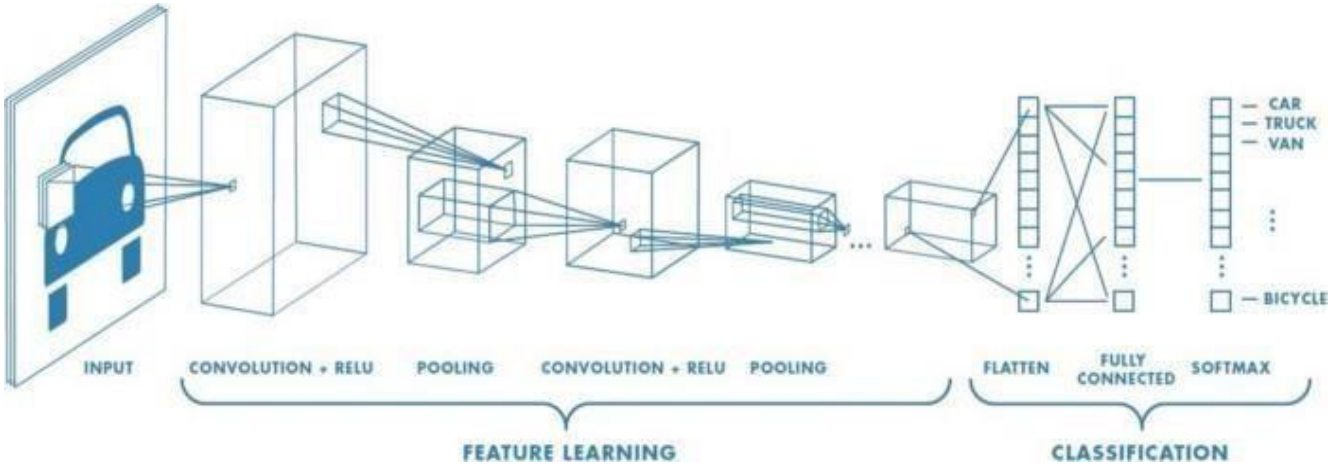


Fig. 3.1 CNN architecture image

A ConvNet's architecture was influenced by how the Visual Cortex is organized and is similar to the connectivity network of neurons in the human brain. Only in this constrained area of the visual field, known as the Receptive Field, do individual neurons react to stimuli. The entire visual

field is covered by a series of such fields that overlap.

A ConvNet may effectively capture the spatial and temporal dependencies in an image by using the appropriate filters. Due to the reduction in the number of parameters needed and the reuse of weights, the architecture achieves a better fitting to the picture dataset. In other words, the network may be trained to comprehend the complexity of the image more effectively.

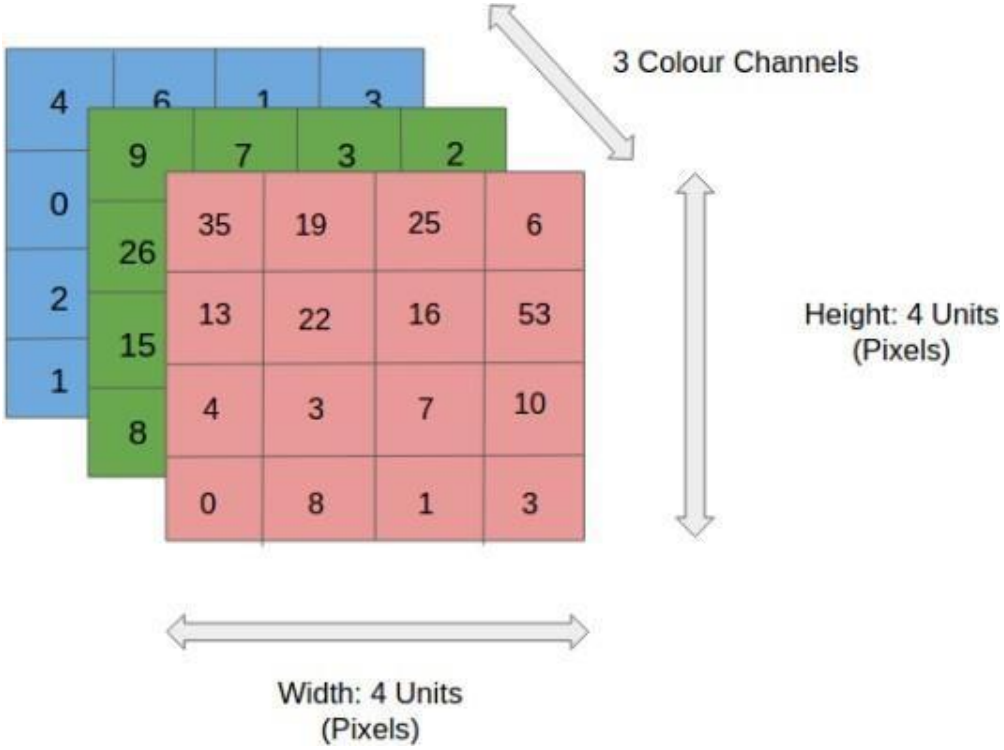


Fig 3.2 Image Complexity Demonstration

To understand the ConvNet as a whole in a simpler way, ConvNet has 3 layers in it which are listed as

1. A Convolutional Layer

2. A Pooling Layer

3. A Fully Connected Layer

1. A Convolutional Layer

CNN's fundamental building block is the convolution layer. The majority of the network's computational load is carried by it.

The dot product of two matrices is performed by this layer, where one matrix

represents the kernel—a set of learnable, parameters— the other matrix represents the constrained area of the receptive field. Compared to a picture, the kernel is smaller in space but deeper. This indicates that the kernel height and width will be spatially small if the image consists of three (RGB) channels, but the depth will go up to all three channels.

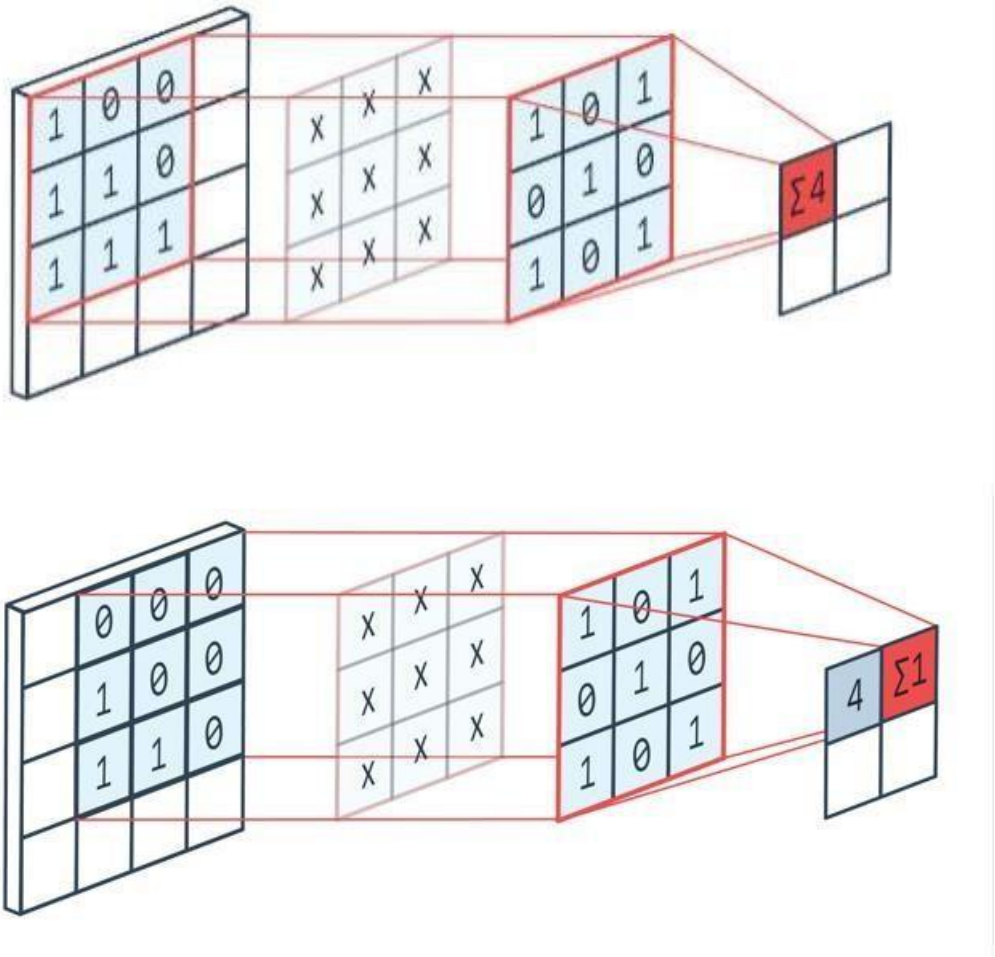


Fig 3.3 Convolutional Layer Demonstration

2. A Pooling Layer

The pooling layer replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs. This helps in reducing the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually.

A weighted average based on the distance from the central pixel is one of the pooling functions, along with the average of the rectangular neighborhood and the L2 norm of the rectangle neighborhood. However, max pooling, which reports the highest output from the neighborhood, is the most widely used technique.

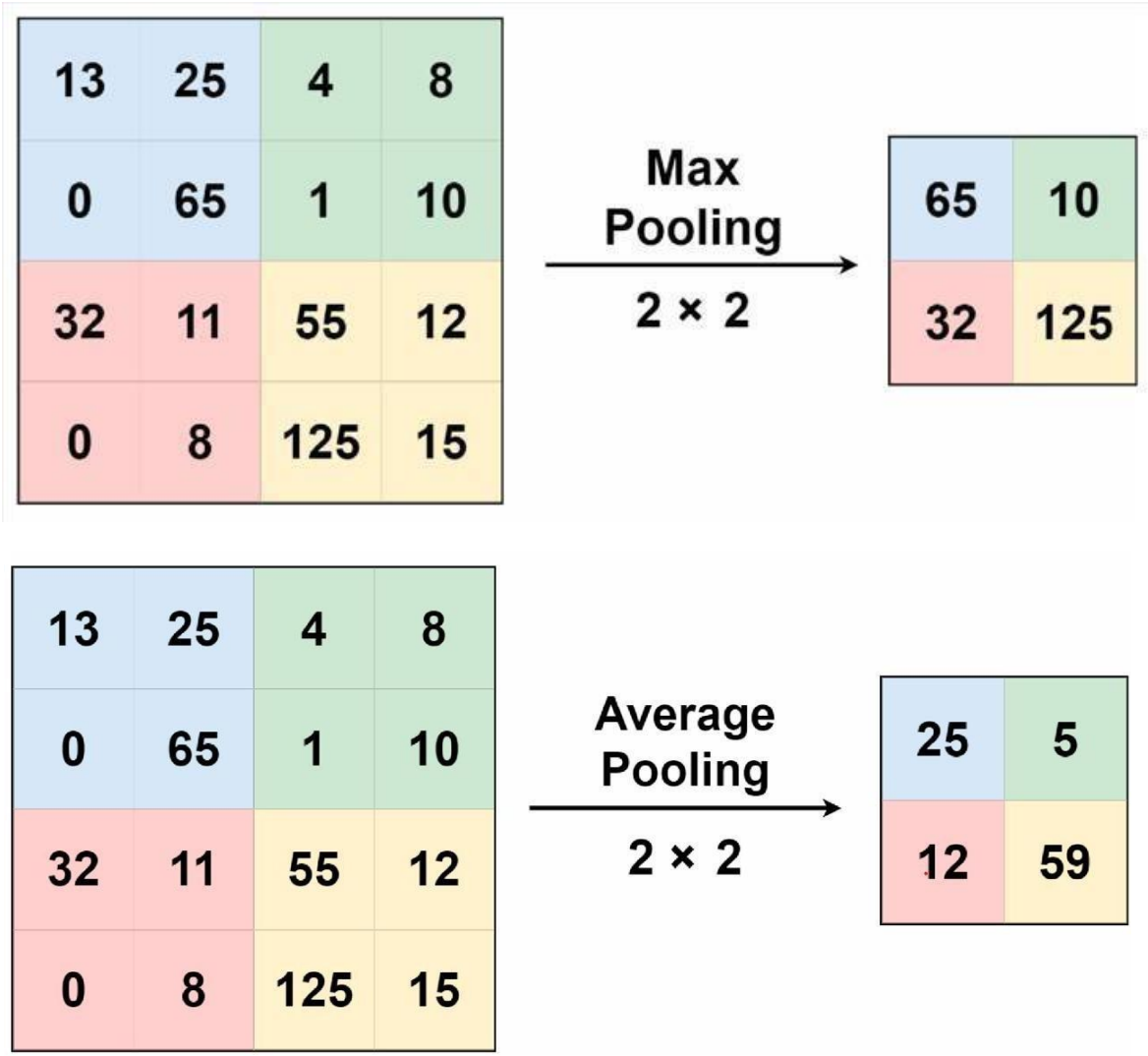


Fig 3.4 Pooling

3. A Fully Connected Layer

As with a conventional FCNN, all of the neurons in this layer are fully connected to every other neuron in the layer above and below. Because of this, it can be calculated using the standard method of matrix multiplication followed by a bias effect.

The representation between the input and the output is mapped with the aid of the FC layer.

Non-Linearity Layers

Non-linearity layers are frequently included right after the convolutional layer to add non-linearity to the activation map as convolution is a linear process and pictures are anything but linear.

There are several types of non-linear operations, the popular ones being:

- Sigmoid
- Tanh
- ReLU

3.2 MODULES:

For efficient stock price movement detection using candlestick patterns, the software can be organized into several modules, each responsible for specific functionalities. Here's a breakdown of these modules:

1. Data Retrieval Module:

- This module is responsible for fetching historical and real-time stock price data from external sources such as financial APIs or data providers.
- It handles the retrieval of data in various formats like JSON, CSV, or XML and ensures data integrity during transmission.

2. Data Preprocessing Module:

- The data preprocessing module cleans and normalizes the retrieved data.
- It handles missing or erroneous data points and prepares the data for candlestick pattern analysis.

3. Candlestick Pattern Recognition Module:

- This module implements algorithms for identifying candlestick patterns such as doji, hammer, engulfing, etc.

- It allows customization of parameters defining each pattern and supports pattern recognition across different timeframes.

4. Pattern Analysis and Confirmation Module:

- The pattern analysis module analyzes consecutive candlesticks to confirm the occurrence of a pattern.
- It incorporates volume analysis to assess the strength of identified patterns and provides statistical analysis of pattern reliability.

5. Alerting and Notification Module:

- This module generates real-time alerts for users when specified candlestick patterns occur.
- It supports various notification channels such as email, SMS, mobile notifications, or integration with trading platforms.

6. Visualization and Reporting Module:

- The visualization module provides graphical representations of detected candlestick patterns overlaid on price charts.
- It generates reports summarizing identified patterns, their frequency, and historical performance, with customization options for chart styles and indicators.

7. Integration with Trading Systems Module:

- This module integrates with trading platforms or APIs to execute trades automatically based on identified patterns.
- It offers APIs for seamless integration with external trading algorithms or systems.

8. User Management and Authentication Module:

- The user management module handles user authentication and authorization, ensuring data security and privacy.

- It provides role-based access control for different user roles such as administrators, analysts, and traders.

9. Performance Optimization Module:

- This module optimizes algorithms for performance to minimize latency in pattern detection.
- It implements scalability measures to handle large volumes of data efficiently, especially during peak trading hours.

10. Backtesting and Simulation Module:

- The backtesting module allows users to test identified patterns against historical data to evaluate their effectiveness.
- It provides simulation environments for testing trading strategies based on detected patterns without risking real capital.

CHAPTER 4

SYSTEM DESIGN AND UML DIAGRAMS

The stock price movement detection system utilizing candlestick patterns is designed to provide traders and investors with valuable insights into market trends and potential price movements. At its core, the system integrates historical and real-time stock price data, employing algorithms to identify specific candlestick patterns indicative of market sentiment and potential future price movements. The system architecture typically comprises several key components. Firstly, a data ingestion module retrieves historical price data from various sources such as financial databases or APIs and stores it in a centralized database for analysis. Secondly, a candlestick pattern detection engine processes the stored data, scanning for predefined candlestick patterns that signal potential market reversals or continuations. This engine often employs machine learning techniques to improve pattern recognition accuracy over time. Additionally, a real-time data feed continuously updates the system with the latest price information, allowing for timely analysis and decision-making. The detected patterns and corresponding analysis are then presented to users through a user interface, which may include visualizations such as charts and graphs to aid in interpretation. Furthermore, the system may incorporate features such as customizable alerts and notifications to keep users informed of significant market developments. Overall, by leveraging candlestick pattern analysis and real-time data, the system empowers traders and investors with actionable insights to make informed trading decisions and capitalize on market opportunities.

4.1 DATAFLOW DIAGRAM:

The data flow diagram for stock price movement detection using candlestick patterns involves several key components. Initially, historical stock price data is retrieved from a data source, which could be an external API or a database server. This data is then fed into the candlestick pattern detection module, where algorithms analyze the price data to identify specific patterns indicative of potential price movements. Once these patterns are detected, the information is

passed to the application server, which processes the data and generates relevant insights. Finally, the results are transmitted to the client/user interface, where users can visualize the detected patterns and make informed decisions regarding their investments. Throughout this process, data flows smoothly from one component to another, enabling efficient analysis and presentation of stock price movement information based on candlestick patterns. Proper handling of data integrity and security is crucial to ensure the accuracy and reliability of the detected patterns for users.

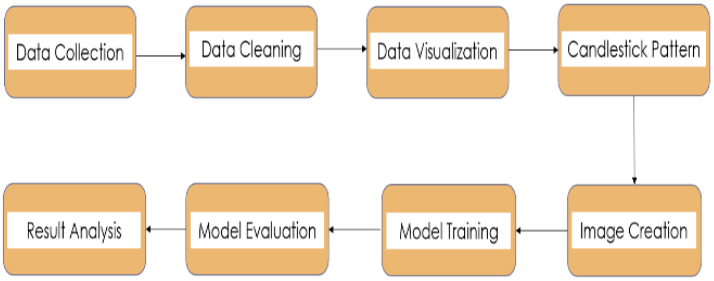


Fig 4.1 Data Flow Diagram

4.2 SYSTEM ARCHITECTURE:

The system architecture is a sophisticated framework designed to harness the predictive power of candlestick patterns in stock price movements. By integrating real-time or historical data with advanced pattern recognition algorithms, the system can discern subtle trends and shifts in market sentiment. The architecture's modular design allows for seamless updates and enhancements, such as incorporating new machine learning models or adjusting the decision-making criteria to align with evolving trading strategies. The end goal is to provide traders and investors with a robust tool for navigating the complexities of financial markets, offering a blend of automated analysis and human expertise to support strategic trading decisions. This approach exemplifies the innovative fusion of technology and finance, aiming to elevate the accuracy and efficiency of market predictions.

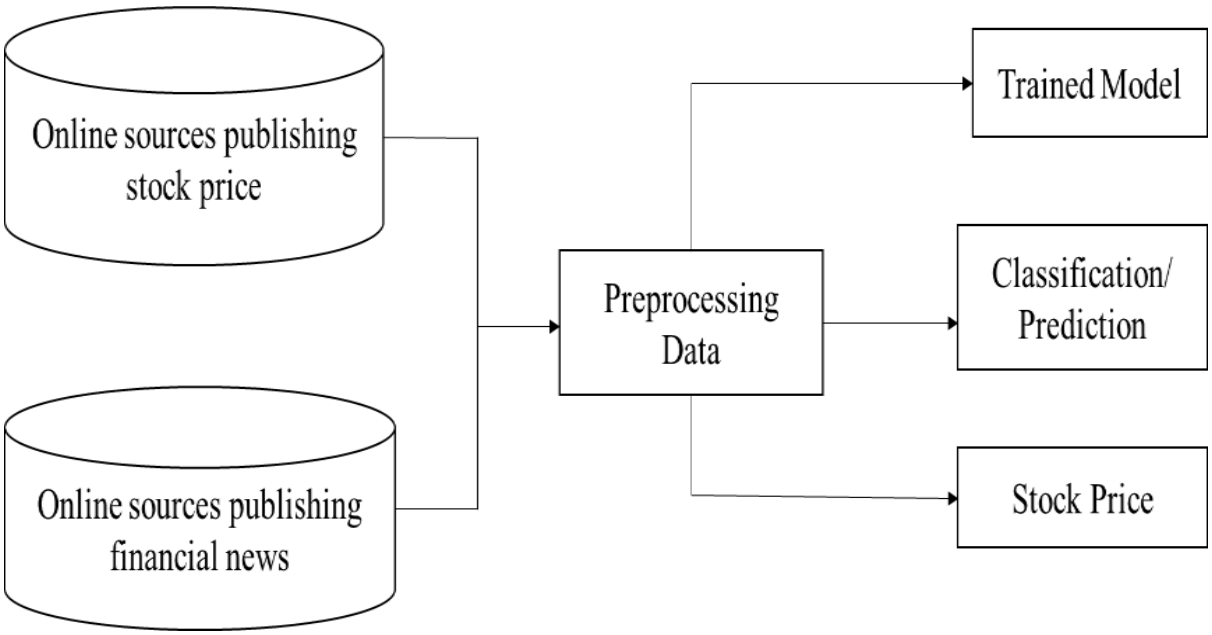


Fig 4.2 System Architecture

4.3 UML DIAGRAMS:

The use of UML diagrams to represent the system for detecting stock price movements through candlestick patterns is a structured approach to understanding and designing complex software. The class diagram serves as the blueprint, detailing the classes like 'Candlestick Pattern', 'Stock Data', and 'Pattern Recognition Algorithm', which are crucial for the system's data structure and functionality. The activity diagram breaks down the process into a series of actions, showing the flow from data analysis to pattern identification, which is essential for algorithm developers and financial analysts. The sequence diagram provides insight into the real-time interactions, ensuring that the system components such as the data source and user interface work in harmony for efficient pattern recognition. Finally, the deployment diagram maps out the software's distribution across various hardware nodes, which is vital for system administrators to understand the deployment architecture. Together, these UML diagrams create a comprehensive framework that aids stakeholders in visualizing, developing, and maintaining the system effectively.

4.3.1 USE CASE DIAGRAM:

The use case diagram for a stock price movement detection system utilizing candlestick patterns is a strategic tool that delineates the roles and actions of various stakeholders within the system. It serves as a blueprint for the system's architecture, ensuring that traders, analysts, and administrators can interact with the system effectively. Traders can view and interpret candlestick patterns to gauge market sentiment, analysts can delve into stock price trends for a comprehensive understanding, and administrators are responsible for maintaining the system's integrity and accessibility. The diagram not only facilitates a clear understanding of the system's capabilities but also aids in the meticulous planning and execution of tasks, thereby optimizing the stock market analysis and enhancing decision-making processes. It is an indispensable component in the design and development of financial analysis tools, providing a structured approach to managing complex data and user requirements.

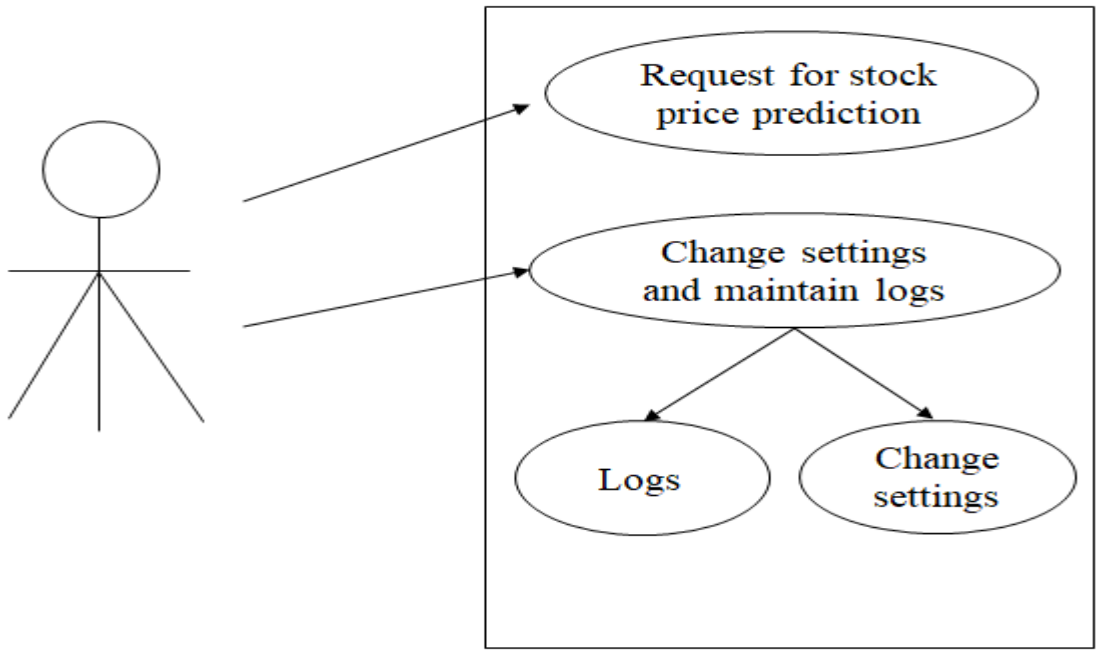


Fig 4.3.1 Use Case Diagram

4.3.2 CLASS DIAGRAM:

The class diagram for a stock price movement detection system is indeed a critical component, as it outlines the structure and relationships of the various classes that make up the system. The financial instrument classes hold key price data, which is fundamental for analyzing candlestick patterns—a popular method used by traders to predict future market movements based on past price actions. The inheritance of candlestick pattern classes from a base class allows for a modular and scalable approach to adding new patterns as the system evolves. Technical indicators and statistical analysis classes further enhance the system's predictive capabilities by providing additional context to the candlestick patterns. Classes dedicated to data retrieval ensure that the system has access to real-time market data, which is crucial for accurate analysis. The processing and visualization classes allow for the efficient handling of data and the presentation of insights in a user-friendly manner. Overall, the class diagram serves as a roadmap for developers, guiding them through the complex landscape of financial software development and ensuring that each component works harmoniously tosupport the goal of detecting stock price movements.

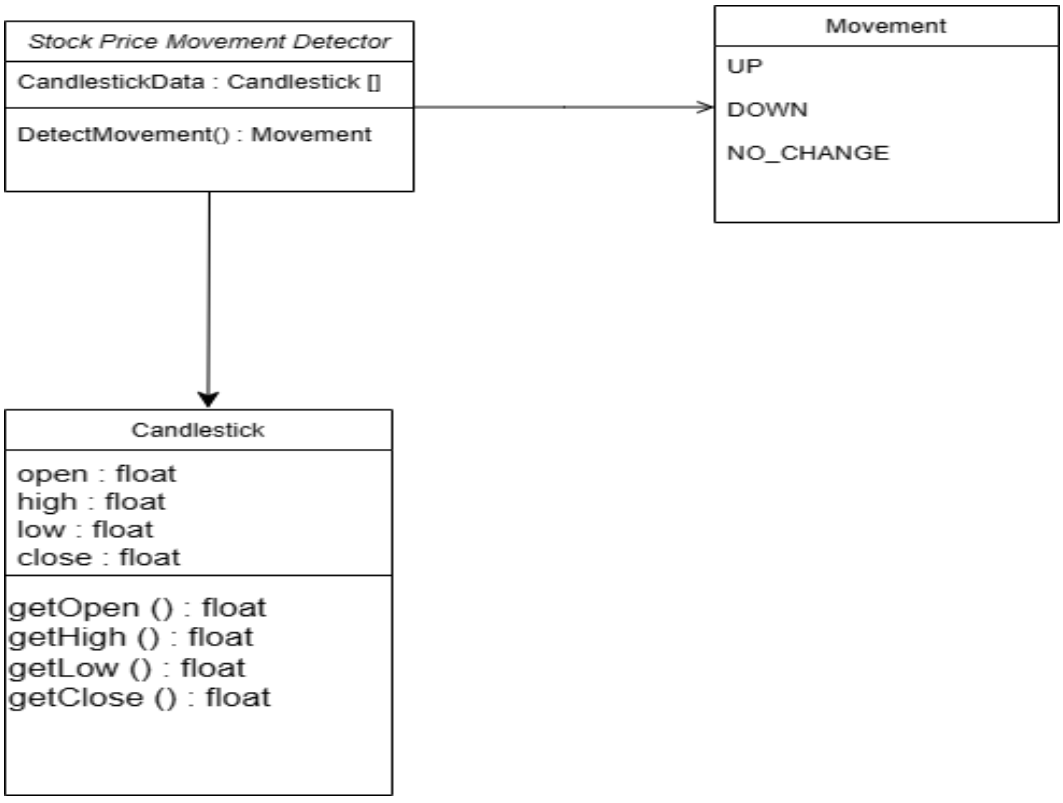


Fig 4.3.2 Class Diagram

4.3.3 SEQUENCE DIAGRAM:

The sequence diagram is a clear representation of the systematic process involved in stock price movement detection using candlestick patterns. It begins with the client interface initiating a request for stock price data, which sets into motion a series of interactions across different components of the system. The application server acts as the central hub, coordinating the retrieval of historical stock data from the database server and then engaging an external service to apply candlestick pattern analysis. This analysis is crucial as it can reveal patterns that signal potential future price movements, providing valuable insights for traders and investors. Once the patterns are identified, the information flows back through the application server to the client interface, where it can be visualized and further examined. This efficient flow of data is essential for timely and informed decision-making in the fast-paced world of stock trading.

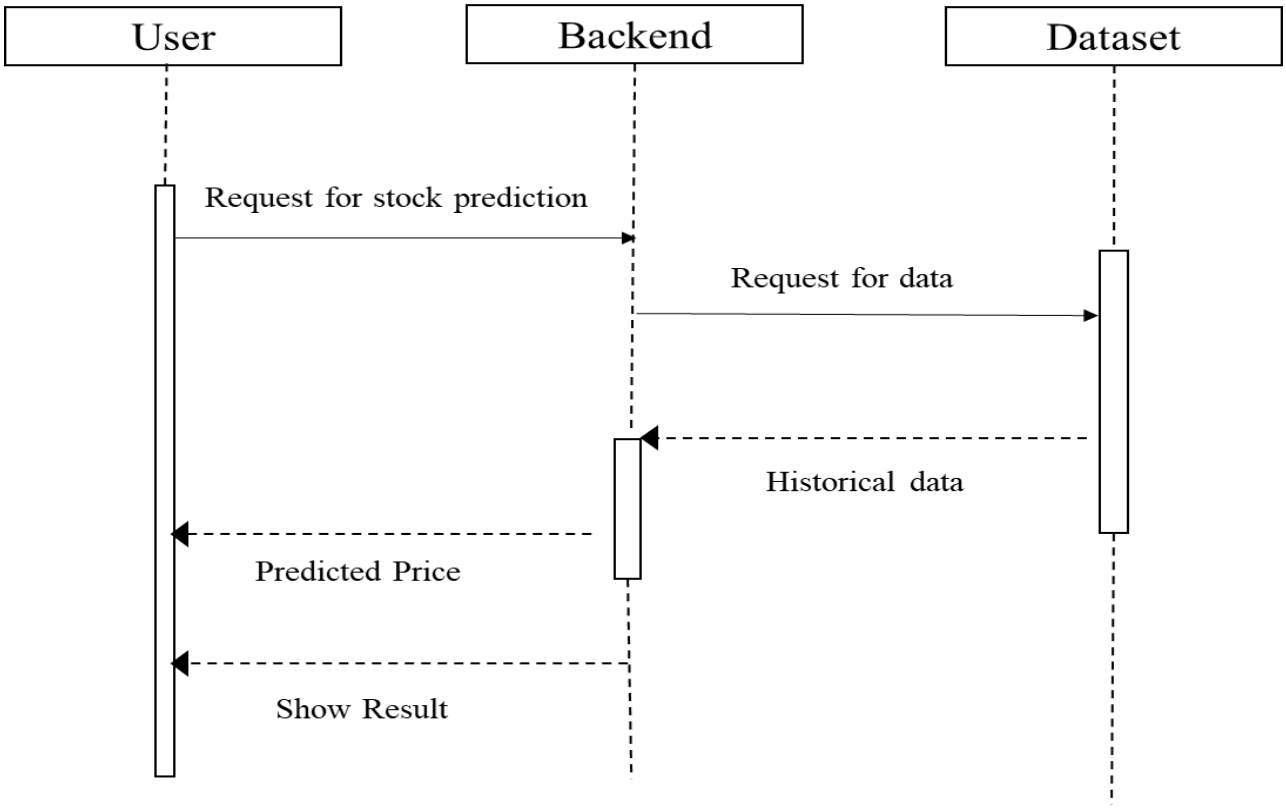


Fig 4.3.3 Sequence Diagram

CHAPTER 5

SOFTWARE DEVELOPMENT LIFE CYCLE

The software development life cycle (SDLC) for developing a system for stock price movement detection using candlestick patterns typically follows a series of steps to ensure the successful development, deployment, and maintenance of the software. Here's an overview of the SDLC stages along with how they apply to this specific project:

5.1 PHASES OF SDLC

1. Requirement Analysis:

- Understand the requirements for the system, including the specific candlestick patterns to be detected, the frequency of data updates, and the desired output format.
- Gather input from stakeholders such as traders, financial analysts, or domain experts to ensure the system meets their needs.

2. System Design:

- Design the architecture of the system, including components for data retrieval, candlestick pattern detection, user interface, and any external integrations.
- Define the data flow within the system, including how historical and real-time stock price data will be obtained and processed.
- Choose appropriate technologies and tools for implementation, such as programming languages, databases, and third-party APIs for financial data.

3. Implementation:

- Develop the software components according to the design specifications.
- Implement algorithms for candlestick pattern detection based on established financial analysis techniques.

- Integrate with external APIs or services for retrieving stock price data if necessary.
- Ensure code quality through testing and code reviews, adhering to best practices and coding standards.

4. Testing:

- Conduct unit tests to verify the functionality of individual components.
- Perform integration tests to ensure that different modules work together correctly.
- Test the system with sample stock price data to validate the accuracy of candlestick pattern detection.
- Perform regression testing to ensure that new changes do not introduce defects in existing functionality.

5. Deployment:

- Prepare the system for deployment to production environments.
- Set up servers or cloud infrastructure to host the software components.
- Configure monitoring and logging to track system performance and detect issues.
- Deploy the system to production and perform smoke tests to verify its functionality in the live environment.

6. Maintenance and Support:

- Monitor the system in production to identify and address any issues or performance bottlenecks.
- Provide ongoing support to users and address their feedback or feature requests.
- Perform regular updates and patches to keep the system secure and up-to-date with changes in the financial markets or external APIs.

CHAPTER 6

IMPLEMENTATION

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
import os

# read in data
data = pd.read_csv('C:\\Users\\Windows\\Desktop\\Major Project\\Stock project\\spy.csv')

# convert 'Date' column to datetime data type
data['Date'] = pd.to_datetime(data['Date'])

# Line plot of Open, Close, High, Low over time
fig, ax = plt.subplots(figsize=(10, 6))
ax.plot(data['Date'], data['Open'], label='Open')
ax.plot(data['Date'], data['Close'], label='Close')
ax.plot(data['Date'], data['High'], label='High')
ax.plot(data['Date'], data['Low'], label='Low')
ax.set_xlabel('Date')
ax.set_ylabel('Price')
ax.set_title('Stock Prices over Time')
ax.legend()
plt.show()

# Scatter plot of Open vs Close prices
fig, ax = plt.subplots(figsize=(8, 8))
```

```
ax.scatter(data['Open'], data['Close'], alpha=0.5)
ax.set_xlabel('Open Price')
ax.set_ylabel('Close Price')
ax.set_title('Open vs Close Prices')
plt.show()

# Histogram of daily price changes
price_changes = data['Close'] - data['Open']
fig, ax = plt.subplots(figsize=(8, 6))
ax.hist(price_changes, bins=30)
ax.set_xlabel('Price Change')
ax.set_ylabel('Frequency')
ax.set_title('Histogram of Daily Price Changes')
plt.show()

# Heatmap of correlation matrix
corr_matrix = data[['Open', 'Close', 'High', 'Low']].corr()
fig, ax = plt.subplots(figsize=(8, 6))
im = ax.imshow(corr_matrix, cmap='coolwarm')
ax.set_xticks(np.arange(4))
ax.set_yticks(np.arange(4))
ax.set_xticklabels(['Open', 'Close', 'High', 'Low'])
ax.set_yticklabels(['Open', 'Close', 'High', 'Low'])
ax.set_title('Correlation Matrix')
plt.colorbar(im)
plt.show()

data.isna().sum()
#Engulfing pattern signals
import random
def Revsignal1(data1):
    length = len(data1)
    High = list(data1['High'])
    Low = list(data1['Low'])
```

```
    Close = list(data1['Close'])
    Open = list(data1['Open'])
    signal = [0] * length
    bodydiff = [0] * length

    for row in range(1, length):
        bodydiff[row] = abs(Open[row]-Close[row])
        bodydiffmin = 0.003
        if (bodydiff[row]>bodydiffmin and bodydiff[row-1]>bodydiffmin and
            Open[row-1]<Close[row-1] and
            Open[row]>Close[row] and
            #Open[row]>=Close[row-1] and Close[row]<Open[row-1]):
            (Open[row]-Close[row-1])>=+0e-5 and Close[row]<Open[row-1]):
            signal[row] = 1
        elif (bodydiff[row]>bodydiffmin and bodydiff[row-1]>bodydiffmin and
            Open[row-1]>Close[row-1] and
            Open[row]<Close[row] and
            #Open[row]<=Close[row-1] and Close[row]>Open[row-1]):
            (Open[row]-Close[row-1])<=-0e-5 and Close[row]>Open[row-1]):
            signal[row] = 2
        else:
            signal[row] = 0
        #signal[row]=random.choice([0, 1, 2])
        #signal[row]=1
    return signal
data['signal1'] = Revsignal1(data)
data[data['signal1']==1].count()


#Target
def mytarget(data1, barsfront):
    length = len(data1)
    High = list(data1['High'])
    Low = list(data1['Low'])
```

```
    Close = list(data1['Close'])
    Open = list(data1['Open'])
    trendcat = [None] * length

    piplim = 300e-5
    for line in range (0, length-1-barsfront):
        for i in range(1,barsfront+1):
                            if    ((High[line+i]-max(Close[line],Open[line]))>piplim)    and
((min(Close[line],Open[line])-Low[line+i])>piplim):
                trendcat[line] = 3 # no trend
            elif (min(Close[line],Open[line])-Low[line+i])>piplim:
                trendcat[line] = 1 #-1 downtrend
                break
            elif (High[line+i]-max(Close[line],Open[line]))>piplim:
                trendcat[line] = 2 # uptrend
                break
            else:
                trendcat[line] = 0 # no clear trend
    return trendcat

data['Trend'] = mytarget(data,3)
#data.head(30)

import numpy as np
conditions = [(data['Trend'] == 1) & (data['signal1'] == 1),(data['Trend'] == 2) &
(data['signal1'] == 2)]
values = [1, 2]
data['result'] = np.select(conditions, values)

trendId=2
print(data[data['result']==trendId].result.count()/data[data['signal1']==trendId].sign
al1.count())
data[ (data['Trend']!=trendId) & (data['signal1']==trendId) ] # false positives
```

```
dfpl = data[150:200]
import plotly.graph_objects as go
from datetime import datetime

fig = go.Figure(data=[go.Candlestick(x=dfpl.index,
                open=dfpl['Open'],
                high=dfpl['High'],
                low=dfpl['Low'],
                close=dfpl['Close'])])
# fig.add_scatter(x=dfpl.index, y=dfpl['pointpos'], mode="markers",
#                 marker=dict(size=5, color="MediumPurple"),
#                 name="signal1")
fig.show()

 # Set the directory path where the images are located
dir_path = '/content'

# Loop through the range of i values and delete the corresponding files
for i in range(127):
    file_path = os.path.join(dir_path, f'image_{i}.png')
    if os.path.exists(file_path):
        os.remove(file_path)

# Load the dataset
data = pd.read_csv('C:/Users/DELL/OneDrive/Desktop/spy.csv')  # Assuming
stock prices are stored in a CSV file
num_entries = data.shape[0]
num_parts = num_entries // 60  # Number of parts to divide the dataset into

# Convert the dataset into images
images = []
for i in range(num_parts):
    start_idx = i * 60
    end_idx = (i + 1) * 60
```

```
    part_data = data.iloc[start_idx:end_idx + 1]
    part_data = part_data[['Open', 'Close', 'High', 'Low']].values
    fig, ax = plt.subplots()
    ax.plot(part_data[:, 0], 'r', label='Open')
    ax.plot(part_data[:, 1], 'g', label='Close')
    ax.plot(part_data[:, 2], 'b', label='High')
    ax.plot(part_data[:, 3], 'k', label='Low')
    ax.legend()
    plt.savefig(f'image_{i}.png')
    plt.close(fig)
    images.append(f'image_{i}.png')
print(len(images))

# Prepare the data for CNN training
X = []
y = []
for i in range(num_parts):
    img_path = images[i]
    img = plt.imread(img_path)
    X.append(img)
    if data.iloc[(i+1)*60-1]['Close'] > data.iloc[(i+1)*60]['Close']:
        y.append(1)
    else:
        y.append(0)
import os

# Set the directory path where the images are located
dir_path = '/content'

# Loop through the range of i values and delete the corresponding files
for i in range(127):
    file_path = os.path.join(dir_path, f'image_{i}.png')
    if os.path.exists(file_path):
        os.remove(file_path)
```

```
X = np.array(X)
y = np.array(y)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, random_state=47)
# Build the CNN model
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(X.shape[1], X.shape[2], X.shape[3])))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Train the model
model.fit(X_train, y_train, epochs=8, batch_size=32)
model.save_weights('model_weights.h5')

# Load the saved weights
model.load_weights('model_weights.h5')
# Evaluate the model
loss, accuracy = model.evaluate(X_test, y_test)
print(f'Test Loss: {loss}, Test Accuracy: {accuracy}')

# Make predictions using the trained model
input_data = data.iloc[-60:].values  # Assuming input data consists of the last 60 days of stock prices
fig, ax = plt.subplots()
ax.plot(input_data[:, 0], 'r', label='Open')
ax.plot(input_data[:, 1], 'g', label='Close')
```

```
ax.plot(input_data[:, 2], 'b', label='High')
ax.plot(input_data[:, 3], 'k', label='Low')
ax.legend()
plt.savefig('input_image.png')
plt.close(fig)
input_image = plt.imread('input_image.png').reshape(1, X.shape[1], X.shape[2], X.shape[3])
prediction = model.predict(input_image)
history = model.fit(X_train, y_train, epochs=8, batch_size=32, validation_data=(X_test, y_test))
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model Loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

# Plot the stock price prediction for the last 60 days
input_data = data.iloc[-60:].values
fig, ax = plt.subplots()
ax.plot(input_data[:, 0], 'r', label='Open')
ax.plot(input_data[:, 1], 'g', label='Close')
ax.plot(input_data[:, 2], 'b', label='High')
ax.plot(input_data[:, 3], 'k', label='Low')
ax.legend()
```

```
import matplotlib.pyplot as plt
import numpy as np

def convert_to_image(data):
    fig, ax = plt.subplots(figsize=(6, 4))
    ax.plot(data['Date'], data['Open'], label='Open')
    ax.plot(data['Date'], data['Close'], label='Close')
    ax.plot(data['Date'], data['High'], label='High')
    ax.plot(data['Date'], data['Low'], label='Low')
    ax.set_xlabel('Date')
    ax.set_ylabel('Price')
    ax.legend()
    plt.title('Stock Price Data')
    plt.tight_layout()

# Save the plot as an image
    plt.savefig('temp_image.png')
    plt.close(fig)

    # Load and return the saved image as a NumPy array
    image = plt.imread('temp_image.png')
    return image

 def predict_movement(input_data):
    input_images = []
    for i in range(7):
        start_idx = -65 + i
        end_idx = -6 + i
        part_data = input_data[start_idx:end_idx + 1]
        input_image = convert_to_image(part_data)
        input_images.append(input_image)
    input_images = np.array(input_images)
```

```
    # Convert the predictions to binary values (1 for up, 0 for down)
    binary_predictions = [1 if pred < 0.5 else 0 for pred in predictions]

    # Return the list of binary predictions
    return binary_predictions

input_data = data.iloc[-65:-7].values
binary_predictions = predict_movement(input_data)
print(binary_predictions)
actual_values=y[-7:]
days = [1, 2, 3, 4, 5,6,7]
fig, ax = plt.subplots()
ax.plot(days, binary_predictions, 'bo', markersize=10, fillstyle='none',
label='Predicted')
ax.plot(days, actual_values, 'rs', markersize=15, fillstyle='none', label='Actual')

# Set x-axis and y-axis labels
ax.set_xlabel('Days')
ax.set_ylabel('Movement')
plt.xticks([1, 2, 3, 4, 5,6,7,8,9,10])
plt.yticks([0, 1])
# Set title
ax.set_title('Predicted vs Actual Movement')

# Add legend to the top right corner
ax.legend(loc='upper right')
 plt.show()

   print('Price goes down!')
```

CHAPTER 7

TESTING

7.1 INTRODUCTION

Field testing will be performed manually and functional tests will be written in detail.

TEST OBJECTIVES:

• All field entries must work properly.
• Pages must be activated from the identified link.
• The entry screen, messages and responses must not be delayed.
• To ensure that during operation the system will perform as per specification.
• To make sure that system meets the user requirements during operation
• To make sure that during the operation, incorrect input, processing and output will bedetected
• To see that when correct inputs are fed to the system the outputs are correct
• To verify that the controls incorporated in the same system as intended
• Testing is a process of executing a program with the intent of finding an error
• A good test case is one that has a high probability of finding an as yet undiscoverederror

FEATURES TO BE TESTED:

• Verify that the entries are of the correct format
• No duplicate entries should be allowed
• All links should take the user to the correct page.

The actual purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner.

7.2 TYPES OF TESTING

- Unit testing.

- Black box testing.
- White box testing.
- Integration testing.
- Functional Testing.
- System testing.

There are many types of testing methods are available in that mainly used testing methodsare as follows

UNIT TESTING:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

INTEGRATION TESTING:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

FUNCTIONAL TESTING:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted. Invalid Input : identified classes of invalid input must be rejected. Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identifiedand the effective value of current tests is determined.

SYSTEM TESTING:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integrationpoints.

WHITE BOX TESTING:

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

BLACK BOX TESTING:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document.

CHAPTER 8

OUTPUT SCREEN

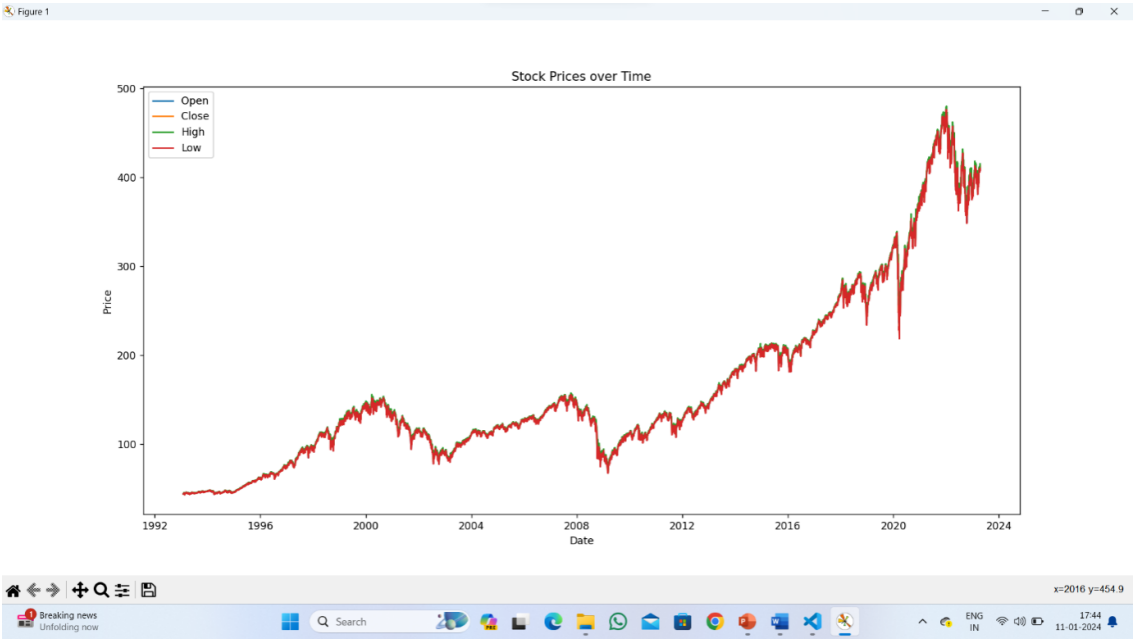


Fig 8.1 Line Plot

Line plot of Open, Close, High, Low over time:

It will give an idea about the price movements of the stock over a period of time. It can help in identifying trends and patterns in the data.

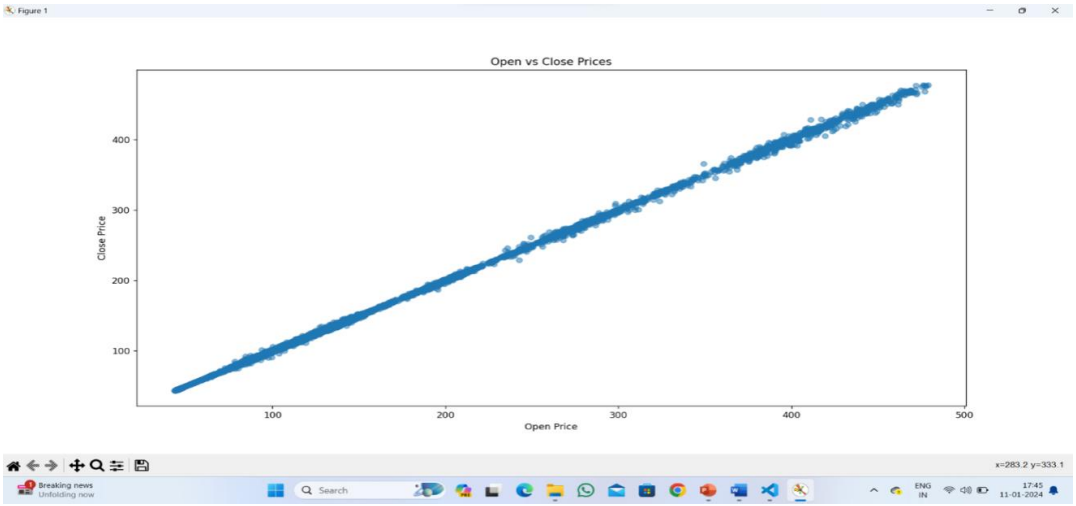


Fig 8.2 Scatter Plot

It will show the relationship between the opening and closing prices of a stock over time. Here the points on the scatter plot are clustered around a straight line with a positive slope, it suggests a strong positive correlation between the opening and closing prices, indicating that the stock tends to close at a higher price than it opened on most days.

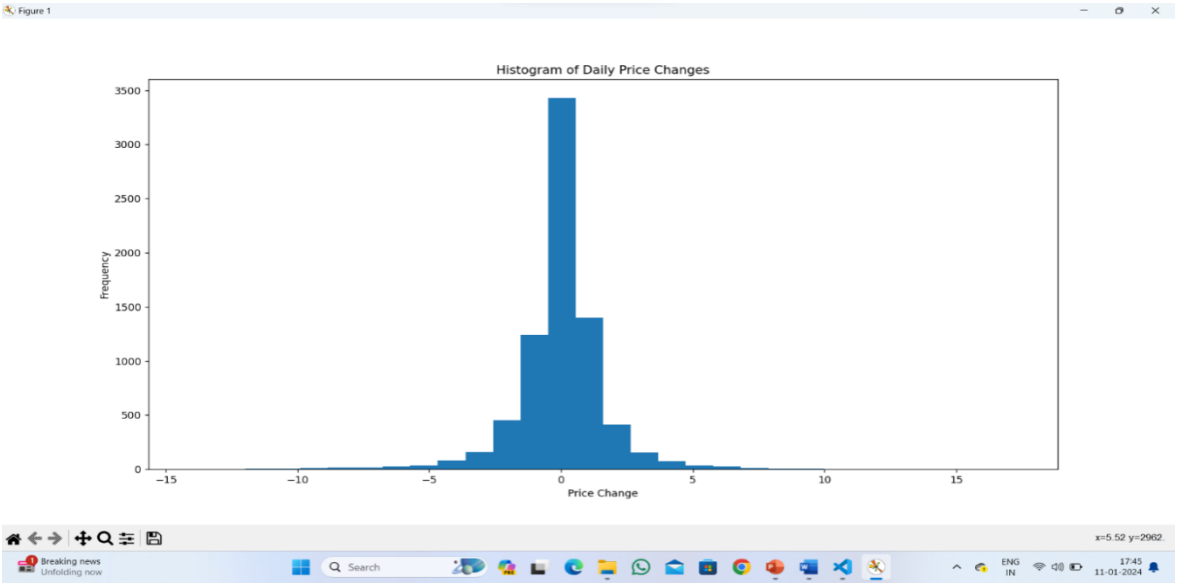


Fig 8.3 Histogram

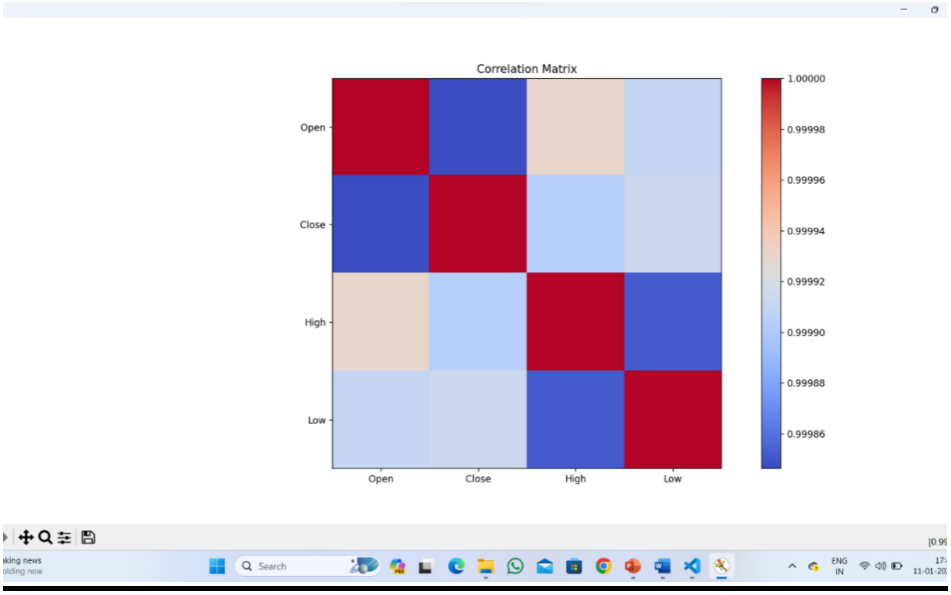


Fig 8.4 Correlation Matrix

The correlation matrix shows how strongly each variable is correlated with every other variable. In this case, the variables are the Open, Close, High, and Low

prices of the stock. The correlation coefficients range from -1 to 1, wirth values close to -1 indicating a strong negative correlation, values close to 1 indicating a strong positive correlation, and values close to 0 indicating little or no correlation.

Fig 8.5 Candlestick

Candlestick pattern recognition is a popular technique used by traders and investors to analyze stock price movements. It involves identifying patterns formed by the open, close, high and low prices of a stock over a given time

period. Here there is a Bullish engulfing pattern in 50 days of candlestick chart.

CHAPTER 9

CONCLUSION AND FUTURE SCOPE

9.1 Conclusion

In conclusion, stock price movement detection using candlestick patterns offers a nuanced understanding of market behavior, empowering traders to make informed decisions amidst market volatility. Through the analysis of historical price data and the identification of candlestick formations, traders can discern market sentiment and anticipate potential price movements. This approach facilitates the timely execution of trades, enabling traders to capitalize on emerging opportunities and mitigate risks effectively.

While candlestick patterns serve as valuable indicators, their effectiveness is enhanced when combined with other technical analysis tools and fundamental insights. Successful implementation relies on continual refinement of algorithms and adaptation to evolving market conditions. Additionally, comprehensive risk management strategies are essential for safeguarding against unexpected market fluctuations.

Moreover, integrating candlestick pattern analysis into trading strategies fosters a disciplined and systematic approach to decision-making.

9.2 Future Scope

The main idea behind stock price prediction is to analyze the available information from the stock market in order to effectively predict future trends which can increase profit. Stock trend prediction techniques play a crucial role to bring more people into market and encourage markets as a whole.

Fundamental analysis involves analyzing a company's financial data to determine the fair value of the company, and to forecast future stock value. Because of this analyzing process, most investors believe that fundamental analysis is mainly suitable for long-term prediction.

CHAPTER 10

REFERENCES

10.1 Websites

Here are some reference websites that provide valuable information and resources on predictive analysis for retail sales using machine learning algorithms:

1. Kaggle (https://www.kaggle.com/): Kaggle is a popular platform for data science competitions and collaborative data analysis. You can find various datasets, kernels (code notebooks), and discussions related to predictive analysis for retail sales, including Big Mart sales datasets and machine learning algorithms.

2. Analytics Vidhya (https://www.analyticsvidhya.com/): Analytics Vidhya is a community-based knowledge portal for analytics and data science professionals. They offer tutorials, articles, courses, and competitions related to predictive analytics and machine learning in retail, including sales forecasting and demand prediction.

3. Medium (https://medium.com/): Medium hosts numerous blogs and publications covering a wide range of topics, including data science and predictive analytics. You can explore articles and case studies related to retail sales prediction and machine learning algorithms by searching relevant keywords.

4. TradingView (www.tradingview.com): TradingView is a comprehensive platform that provides advanced charting tools, including the ability to detect candlestick patterns. Users can customize their charts and receive alerts based on specific candlestick patterns.

5. StockCharts (www.stockcharts.com): StockCharts offers a wide range of technical analysis tools, including the identification of candlestick patterns. Users can access predefined scans or create custom scans to detect specific patterns.

6. Finviz (www.finviz.com): Finviz provides a variety of stock screening tools, including the ability to screen for stocks based on candlestick patterns. Users can filter stocks based on their preferred candlestick patterns and other criteria.

7. ChartMill (www.chartmill.com): ChartMill offers technical analysis tools, including the detection of candlestick patterns. Users can scan for specific patterns across various timeframes and receive alerts when patterns are identified.

8. Investopedia (www.investopedia.com): Investopedia offers educational resources on candlestick patterns and technical analysis. While it doesn't provide real-time scanning capabilities, it offers valuable information on how to identify and interpret candlestick patterns.

RESEARCH PAPERS

electronics

MDPI

Article

Stock Price Prediction Using Candlestick Patterns and Sparrow Search Algorithm

Xiaozhou Chen [1], Wenping Hu [2] and Lei Xue [1,*]

[1] School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China; xzchen429@shu.edu.cn
[2] School of Science, Xinjiang Institute of Technology, Aksu 843000, China; wenpinghu028@126.com
* Correspondence: xuelei@shu.edu.cn

Abstract: Accurately forecasting the trajectory of stock prices holds crucial significance for investors in mitigating investment risks and making informed decisions. Candlestick charts visually depict price information and the trends in stocks, harboring valuable insights for predicting stock price movements. Therefore, the challenge lies in efficiently harnessing candlestick patterns to forecast stock prices. Furthermore, the selection of hyperparameters in network models has a profound impact on the forecasting outcomes. Building upon this foundation, we propose a stock price prediction model SSA-CPBiGRU that integrates candlestick patterns and a sparrow search algorithm (SSA). The incorporation of candlestick patterns endows the input data with structural characteristics and time series relationships. Moreover, the hyperparameters of the CPBiGRU model are optimized using an SSA. Subsequently, the optimized hyperparameters are employed within the network model to conduct predictions. We selected six stocks from different industries in the Chinese stock market for experimentation. The experimental results demonstrate that the model proposed in this paper can effectively enhance the prediction accuracy and has universal applicability. In comparison to the LSTM model, the proposed model produces an average of 31.13%, 24.92%, and 30.42% less test loss in terms of MAPE, RMSE and MAE, respectively. Moreover, it achieves an average improvement of 2.05% in R^2.

Keywords: stock price forecasting; candlestick pattern; sparrow search algorithm; BiGRU

Citation: Chen, X.; Hu, W.; Xue, L. Stock Price Prediction Using Candlestick Patterns and Sparrow Search Algorithm. *Electronics* **2024**, *13*, 771. https://doi.org/10.3390/electronics13040771

Academic Editor: Domenico Rosaci

Received: 27 December 2023
Revised: 1 February 2024
Accepted: 12 February 2024
Published: 16 February 2024

1. Introduction

The stock market embodies an intricate and dynamic system replete with features like non-linearity, non-stationarity, volatility, and significant noise. These attributes render the change process of time series data, such as stock prices, susceptible to the influence of multifaceted factors, thereby imparting a profound sense of uncertainty to the direction of these changes [1,2]. Forecasting the movements of stock prices with precision can provide investors with a greater level of confidence in making informed decisions, leading to substantial gains. Consequently, the prediction of stock prices has emerged as a focal point of research within the realm of the financial market.

Experts and scholars have conducted extensive research on forecasting financial time series such as stock prices. The commonly used methods of technical analysis can be roughly categorized into three types: statistical analysis methods, machine-learning methods, and deep learning methods. Statistical analysis methods such as Autoregressive Moving Average (ARMA) [3], Autoregressive Integrated Moving Average (ARIMA), Generalized Autoregressive Conditional Heteroskedasticity (GARCH), and Autoregressive Conditional Heteroskedasticity (ARCH), have been extensively researched and applied by numerous scholars and investors. Jarrett et al. [4] used the ARIMA model to forecast and analyze the Chinese stock market. Juan et al. [5] applied the GARCH model to determine the predictive ability of Dubai International Airport's activity volume on the UAE stock

market. Given the inherent nonlinearity and excessive noise found in stock price data, the limitations arise when employing statistical analysis methods that assume a linear model structure. Consequently, achieving improved accuracy in predictions becomes challenging.

Machine-learning methods, such as Random Forest and Support Vector Machine (SVM), have been employed in the prediction of stock prices. This choice is motivated by their adeptness in capturing intricate relationships in time series data while simultaneously ensuring efficient computational performance. Lin et al. [6] utilized Principal Component Analysis to analyze and predict stocks, which provides a deeper exploration of machine-learning algorithms in the prediction field of financial time series data. Patel et al. [7] employed the naive Bayesian method for stock index prediction. Nti et al. [8] utilized the Random Forest method, optimized through decision tree optimization, to predict stock market indices and compared its performance against various traditional machine-learning models. Fu et al. [9] employed an SVM algorithm to predict financial time series data, yielding favorable prediction outcomes. Xu et al. [10] used an optimized integrated learning classifier to predict stock prices, thereby reducing prediction errors. The prowess of machine-learning techniques in the acquisition and processing of information from data falls short, despite notable strides in enhancing both speed and accuracy in predictions.

Deep learning, with its capacity for large-scale data processing, computation, and remarkable non-linear fitting ability, enables predictive models to deeply comprehend the profound correlation within the data. Common methods encompass the Back Propagation (BP) neural network, Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), etc. Zhang [11] contrasted the deep learning neural network model with the time series model, demonstrating that the neural network model is more capable of processing nonlinear data. Additionally, it outperformed the time series ARIMA model. Yu et al. [12] used the local linear embedding dimensional reduction algorithm (LLE) to diminish the dimensionality of factors influencing stock prices. Subsequently, they employed the BP neural network to forecast stock prices, thereby enhancing the accuracy of predictions. The employment of deep learning in the prediction framework has proven advantageous in accomplishing more accurate data predictions. Nonetheless, current prediction approaches and models necessitate further refinement. For instance, the information source exhibits a relatively narrow focus, primarily consisting of prevalent technical data, such as opening price, highest price, and lowest price, lacking comprehensive integration of other relevant information. Furthermore, the determination of hyperparameter magnitudes in models is typically reliant on experiential judgment, resulting in a challenging and rather subjective selection process.

Candlesticks are essentially daily samples of high-frequency price data, which can effectively express trend signals and have structural characteristics. In the empirical analysis, astute investors frequently anticipate fluctuations in stock prices by analyzing the candlestick series, confirming the inherent association between candlesticks and stock movements. Candlestick patterns are particular sequences of candlesticks on a candlestick chart that are primarily employed to identify trends. The candlestick pattern captures this information and portrays it in the form of a single candlestick or a combination of candlesticks with varying lengths [13].

However, employing candlestick patterns for the prediction of stock prices continues to be a challenging task. By learning and combing the existing research methods, this paper proposes a Bidirectional Gated Recurrent Unit Model Integrating Candlestick Patterns and a Sparrow Search Algorithm (SSA-CPBiGRU) model for stock price forecasting. The innovations and contributions of our work comprise the following three points:

1. Currently, the primary information utilized for predicting stock prices is basic market data, which lacks structural relationships and exhibits limited expression capacity for the overall system state. This model ingeniously integrates candlestick patterns with stock market data, serving as input for the stock price prediction model, imbuing the input data with structural characteristics and time series relationships. Furthermore,

this paper uses a Bidirectional Gated Recurrent Unit (BiGRU) network to extract deeper feature relationships, thereby enhancing the learning ability of the network;
2. This paper applies a sparrow search algorithm (SSA) [14] to stock price forecasting, addressing the challenge of high randomness and difficulty in hyperparameter selection of the CPBiGRU network. Simultaneously, it enhances the accuracy of stock price forecasting;
3. Current research typically utilizes data from the same time window for forecasting. However, in actual trading decisions, investors often refer to stock price information from multiple trading days. Therefore, this paper explores the impact of extracting stock data from different time windows on prediction results.

The rest of the paper is organized as follows. Related work is covered in Section 2. The methodology is explained in Section 3. Section 4 presents the results, showing the experiments performed, and Section 5 provides the conclusions, with future directions listed in detail.

2. Related Work

We discuss published literature in two different categories here. These categories are related to candlestick patterns analysis and using deep learning models in stock prediction, respectively. The models pertaining to these domains is reviewed in detail and elaborated on.

2.1. Candlestick Patterns Analysis

Candlestick charts are a form of technical analysis visualization that are created by plotting the opening, highest, lowest, and closing prices of each analysis period [15]. Figure 1 represents an example of a candlestick chart. A box used to represent the difference between the open price and the close price is called the body of the candlestick. When the close price is higher than the open price, the body color is red, otherwise the body is green. The research environment of candlestick patterns is multifaceted. It can be categorized into three primary cases: sequential pattern mining and its applications, candlesticks and their applications, and stock time series forecasting.

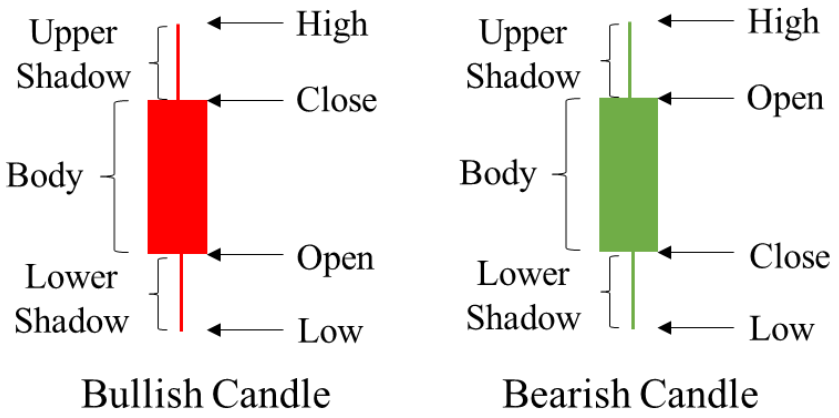


Figure 1. Example of candlestick chart.

The sequential pattern mining algorithms have mainly concentrated on discovering rules, generating sequences, combining resources [16], and predicting sequences. Aiming for accurate destination forecasting, and giving its first partial trip trajectory as input, Iqbal et al. [17] considered it a multi-class prediction problem and proposed an efficient, non-redundant contrast sequence miner algorithm to distinguish mine patterns, which can only be seen in one class. Li et al. [18] proposed a one-pass algorithm, with a commonly used frequency as the output for a given sequence, and two advanced models to further improve the processing efficiency of voluminous sequences and streaming data. To address the shortcomings of traditional sequence pattern mining algorithms, Wang et al. [19] pro-

posed a timeliness variable threshold and an increment Prefixspan algorithm, verifying the effectiveness of the algorithm. Sophisticated investors can analyze candlestick sequences in historical data and speculate on the patterns that will appear in the next period, thus enabling them to prognosticate future trends in stock markets. Candlesticks and their applications mainly focus on the image processing of candlesticks, identifying and interpreting certain candlestick patterns, while also striving to enhance recognition accuracy. Birogul et al. [20] and Guo et al. [21] encoded candlestick data into 2D candlestick charts and learned the morphological features of the candlestick data through deep neural networks. Chen et al. [22] proposed a two-step approach for the automatic recognition of eight candlestick patterns, with an average accuracy surpassing that of the LSTM model. Fengqian et al. [23] used candlestick charts as a generalization of price data across a timeframe, serving as a denoising tool. They subsequently employed cluster analyses and reinforcement learning methods to achieve online adaptive control of parameters within unfamiliar environments, ultimately enabling the implementation of high frequency transaction strategies.

Stock time series forecasting is mainly divided into two categories: stock price forecasting and stock trend forecasting. Wang et al. [24] evaluated the effectiveness of several renowned candlestick patterns, employing recent data from 20 American stocks to estimate stock prices. Udagawa et al. [13] proposed a hybrid algorithm that combines candlesticks sharing a specific price range into a single candlestick, thereby eliminating noisy candlesticks. Madbouly et al. [25] combined a cloud model, fuzzy time series and Heikin-Ashi candlesticks to forecast stock trends, thereby enhancing the accuracy of predictions. Wang et al. [26] proposed a quantification method for stock market candlestick charts based on the Hough variation. They utilized the graph structure embedding method and multiple attention graph neural networks to improve the prediction performance of stock prices.

2.2. Deeping Learning Approaches in Stock Prediction

Currently, deep learning approaches have become the predominant focus of research both domestically and internationally. Numerous experts and scholars have dedicated their efforts to exploring this field extensively. Rather et al. [27] used RNN with memory capability to predict stock returns. Minami [28] employed a variant RNN model called LSTM, which effectively alleviates common issues in neural networks such as gradient vanishing and explosions, as well as long-range dependence. Gupta et al. [29] augmented prediction speed by adopting GRU, a network structure with fewer gating mechanisms compared to LSTM, as the main network structure of the prediction model. Chandar et al. [30] used s wavelet neural network model to predict stock price trends. The aforementioned scholars have utilized historical stock price data at the input level of their models in order to conduct their prediction research. Additionally, deep learning techniques for stock prediction can incorporate a wider and more varied range of information sources, thereby enriching the factors influencing the prediction model. Cai et al. [31] integrated the characteristics of financial- and stock market-related news posts into a hybrid model consisting of a LSTM and Convolutional Neural Network (CNN) for forecasting. They constructed the prediction model using a multi-layer recurrent neural network. However, although the complexity of the model was increased, the prediction accuracy could be further improved. Ho et al. [32] combined candlestick charts with social media data, proposing a multi-channel collaborative network for predicting stock trends.

Furthermore, the current existing network models still have certain limitations. The selection of hyperparameters in these models is often based on existing research or experience, coming with a certain degree of subjectivity. Appropriate hyperparameters can improve the topology of the network model and enhance its generalization and fitting capabilities. Hence, the issue of eliminating the influence of human factors and finding the optimal network model hyperparameters is a matter of concern for scholars. Hu [33] employed a Bayesian algorithm to optimize the learning rate, the number of hidden layers, and the number of neurons within LSTM, with the goal of forecasting the stock prices of prominent stocks at specific stages of the Chinese stock market. The experimental

results demonstrated that the optimized model possesses high universality and efficacy. Optimization of important parameters in the proposed network by a swarm intelligent optimization algorithm can solve the problems of network hyperparameters selection with high randomness, difficult selection, and influence by human factors, improving the prediction accuracy [34,35]. Song et al. [36] utilized the Particle Swarm Optimization (PSO) with adaptive learning strategy to optimize the time step, the batch size, the number of iterations, and the number of hidden layer neurons in the LSTM network. The SSA has a stronger optimization ability than the Grey Wolf Optimization (GWO), Ant Colony Optimization (ACO), and PSO [14]. Li et al. [37] conducted a comprehensive investigation comparing the Bat Algorithm (BA), GWO, Dragonfly Algorithm (DA), Whale Optimization Algorithm (WOA), Grasshopper Optimization Algorithm (GOA), and SSA. The performance of these algorithms, including their convergence speed, accuracy, and stability, was compared using 22 standard CEC test functions. The results clearly demonstrated that the SSA surpassed the other five algorithms in all aspects. Liao [38] utilized the SSA to optimize the weights, bias, the number of neurons in the hidden layer, and the number of iterations within the LSTM network. This methodology has heightened the prediction accuracy of the model and has been effectively implemented to address load forecasting issues with favorable results.

3. Methodology

The full system diagram is shown in Figure 2. Firstly, the stock market data and candlestick are preprocessed separately. Subsequently, the CPBiGRU optimized by SSA is utilized to forecast the closing price of stocks. Finally, the evaluation metrics are employed to evaluate the model performance.

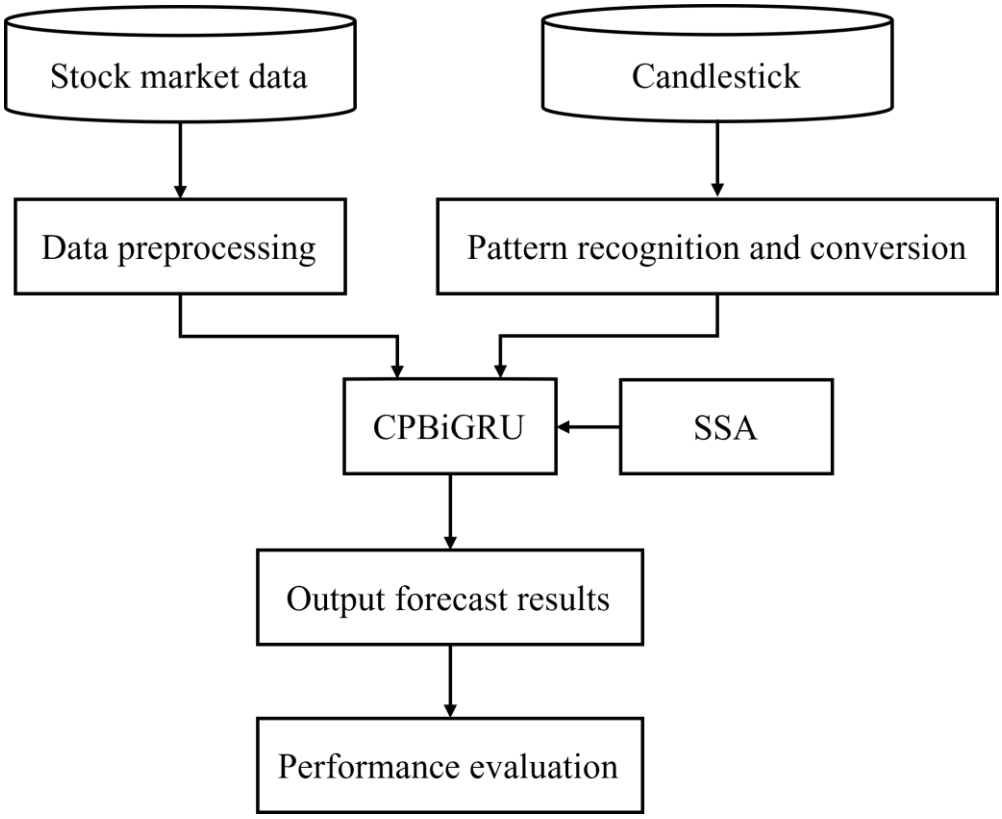


Figure 2. System diagram of the proposed method.

3.1. BiGRU Network

The GRU model proposed by Cho et al. [39] is a variant of LSTM. GRU simplifies the input gate and forget gate in LSTM into an update gate, which controls the input and previous output to be transferred to the next cell. Additionally, it sets a reset gate to regulate

the amount of historical information to be forgotten. It can effectively avoid the problems of long-range dependence and insufficient memory capacity of traditional recurrent neural networks. In contrast to LSTM, GRU exhibits a more simple architecture, as well as faster training and fitting speeds [40]. The architecture of the GRU is shown in Figure 3, and the operation formula is defined by Equations (1)–(4).

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \tag{1}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \tag{2}$$

$$\tilde{h_t} = tanh\left(W \cdot [r_t * h_{t-1}, x_t] + b_{\tilde{h}}\right) \tag{3}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h_t} \tag{4}$$

where x_t and h_t represent the input value and output value of the GRU network at moment t, respectively. In addition, h_{t-1} is the output at the previous moment, $\tilde{h_t}$ is the candidate value of the memory cell value at moment t, r_t represents the reset gate, and z_t represents the update gate. W_z, W_r, and W are the weight matrices of the update gate, reset gate and candidate hidden layer state, respectively. b_z, b_r, and $b_{\tilde{h}}$ are the bias terms of the update gate, reset gate and candidate hidden layer state, respectively, while σ is the sigmoid activation function.

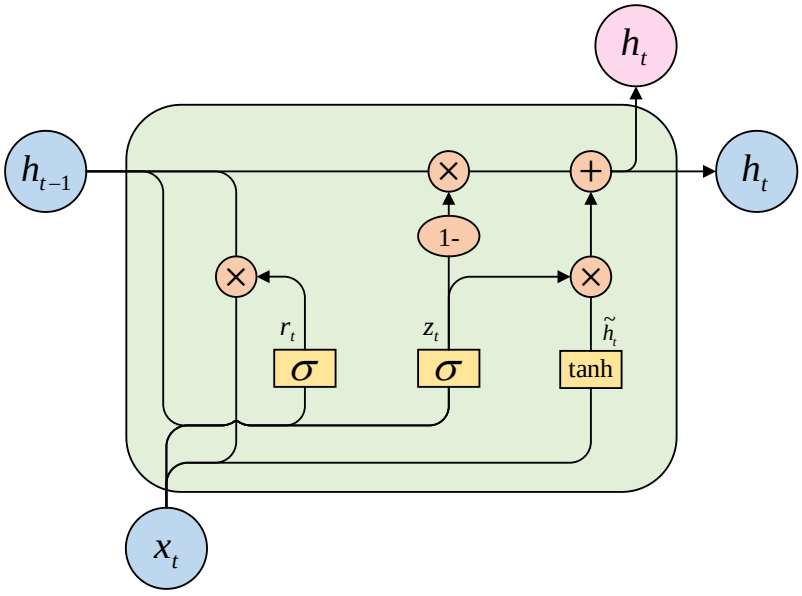


Figure 3. GRU architecture block diagram.

However, the GRU network only considers unidirectional information flow while disregarding the influence of other directions of information flow. The lack of sufficient influential factors and information characteristics for data at prediction points imposes certain limitations on the predictive performance of the network. BiGRU is a neural network architecture that considers information flow in both historical and future directions. Building upon the foundation of the unidirectional GRU, it incorporates an additional layer of reverse-GRU. It not only improves the issue of temporal dependencies in data, but also expands the number of neural units, enabling more precise prediction results. Therefore, we choose BiGRU as the foundational model for prediction methodology.

The output value of BiGRU network unit is obtained by combining the hidden unit output values in both backward and forward directions. The final output of BiGRU is calculated as shown in Equation (5): where w_t and v_t represent the weights corresponding to the forward hidden layer state $\overrightarrow{h_t}$ and the reverse hidden layer state $\overleftarrow{h_t}$ corresponding to

the BiGRU at moment t, b_t represents the offset corresponding to the hidden layer state at moment t. The structure diagram of BiGRU is shown in Figure 4.

$$h_t = w_t \overrightarrow{h_t} + v_t \overleftarrow{h_t} + b_t \tag{5}$$

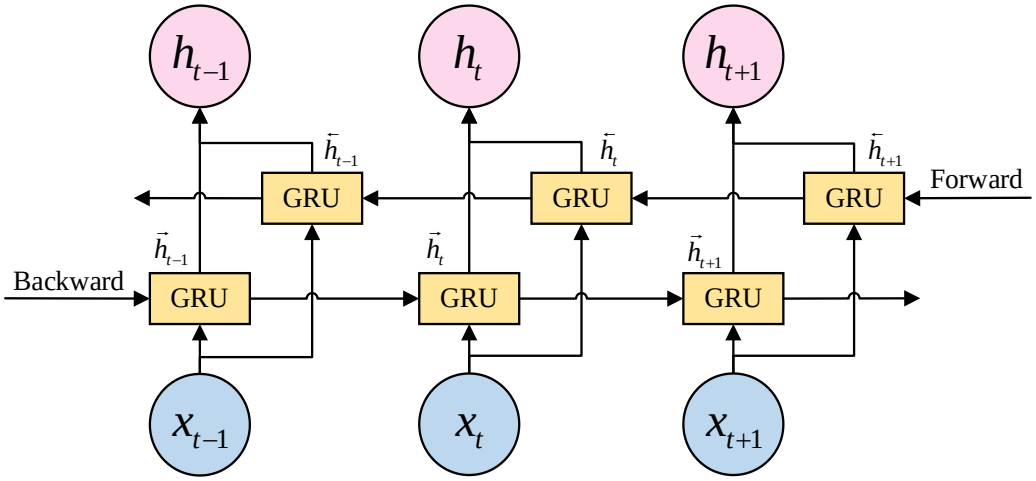


Figure 4. BiGRU structure diagram.

3.2. A Dual Port BiGRU Network Integrating Candlestick Patterns

Candlestick patterns reflect the market trend and price information, and they can be roughly divided into two categories: reversal patterns and continuation patterns. Talib is a Python quantitative indicator library, offering functions to identify 61 candlestick patterns. These functions return three values: 0, 100 and −100. Here, 100 indicates recognition of the pattern, while −100 indicates recognition of the pattern's inverse form, 0 indicates that the candlestick pattern is not recognized, 100 indicates the recognition of the pattern, and −100 indicates the recognition of the pattern's inverse form. Since the return values of different patterns indicate different market trends, they cannot be directly incorporated into the stock price prediction models. The experiment defines three kinds of stock trend prediction values: −1, 0, and 1. Here, −1 indicates a downward trend in the stock, 1 indicates an upward trend in the stock, and 0 indicates that the candlestick pattern is not detected. In this paper, the return values of 61 candlestick patterns are transformed into corresponding stock trend prediction values based on the pattern definition. The generated correlation table of stock trend prediction values is presented in Table 1. For example, the candlestick pattern of two crows predicts a falling stock price. When this pattern is identified, the function in Talib that recognizes this pattern will return a value of 100. When the reverse pattern is identified, the function will return a value of −100. Therefore, in the correlation table of stock trend predictions, the return values of 100 and −100 for the two crows pattern correspond to stock trend predictions of −1 and 1, respectively.

If multiple candlestick patterns are recognized on a certain day, the stock trend prediction for that day is the sum of the corresponding stock trend predictions for all identified patterns, e.g, if the engulfing pattern and belt-hold pattern are detected on a certain day, and the return values for both patterns are 100. Firstly, we refer to the correlation table to find the corresponding stock trend prediction value of 1 for these two patterns. Then, we add the trend prediction values for these two patterns together, resulting in a stock trend prediction value of 2 for this day. The stock trend prediction value and the stock market data are individually normalized and employed as inputs to the CPBiGRU network. The flowchart of CPBiGRU model is shown in Figure 5.

Table 1. The correlation table of stock trend prediction values.

Candlestick Patterns	Return Value	Predictive Trending	Candlestick Patterns	Return Value	Predictive Trending
Two Crows	100 −100	−1 1	Identical Three Crows	100 −100	−1 −1
Three Black Crows	100 −100	−1 1	In-Neck Pattern	100 −100	1 −1
Three Inside Up/Down	100 −100	1 −1	Inverted Hammer	100 −100	1 −1
Three Line Strike	100 −100	1 −1	Kicking	100 −100	1 −1
Three Outside Up/Down	100 −100	1 −1	Kicking-bull/bear determined by the longer marubozu	100 −100	1 −1
Three Stars In The South	100 −100	1 −1	Ladder Bottom	100 −100	1 −1
Three Advancing White Soldiers	100 −100	1 −1	Long Legged Doji	100 −100	1 −1
Abandoned Baby	100 −100	1 −1	Long Line Candle	100 −100	1 −1
Advance Block	100 −100	1 −1	Marubozu	100 −100	1 −1
Belt-hold	100 −100	1 −1	Matching Low	100 −100	1 −1
Breakaway	100 −100	1 −1	Mat Hold	100 −100	1 −1
Closing Marubozu	100 −100	1 −1	Morning Doji Star	100 −100	1 −1
Concealing Baby Swallow	100 −100	1 −1	Morning Star	100 −100	1 −1
Counterattack	100 −100	1 −1	On-Neck Pattern	100 −100	1 −1
Dark Cloud Cover	100 −100	−1 1	Piercing Pattern	100 −100	1 −1
Doji	100 −100	1 −1	Rickshaw Man	100 −100	1 −1
Doji Star	100 −100	1 −1	Rising/Falling Three Methods	100 −100	1 −1
Dragonfly Doji	100 −100	1 −1	Separating Lines	100 −100	1 −1
Engulfing Pattern	100 −100	1 −1	Shooting Star	100 −100	−1 1
Evening Doji Star	100 −100	1 −1	Short Line Candle	100 −100	1 −1
Evening Star	100 −100	1 −1	Spinning Top	100 −100	−1 1
Up/Down-gap side-by-side white lines	100 −100	1 −1	Stalled Pattern	100 −100	1 −1
Gravestone Doji	100 −100	1 −1	Stick Sandwich	100 −100	1 −1

Table 1. *Cont.*

Candlestick Patterns	Return Value	Predictive Trending	Candlestick Patterns	Return Value	Predictive Trending
Hammer	100 −100	1 −1	Takuri	100 −100	−1 1
Hanging Man	100 −100	1 −1	Tasuki Gap	100 −100	1 −1
Harami Pattern	100 −100	1 −1	Thrusting Pattern	100 −100	1 −1
Harami Cross Pattern	100 −100	1 −1	Tristar Pattern	100 −100	1 −1
High-Wave Candle	100 −100	−1 1	Unique 3 River	100 −100	1 −1
Hikkake Pattern	100 −100	1 −1	Upside Gap Two Crows	100 −100	−1 1
Modified Hikkake Pattern	100 −100	1 −1	Upside/Downside Gap Three Methods	100 −100	1 −1
Homing Pigeon	100 −100	1 −1			

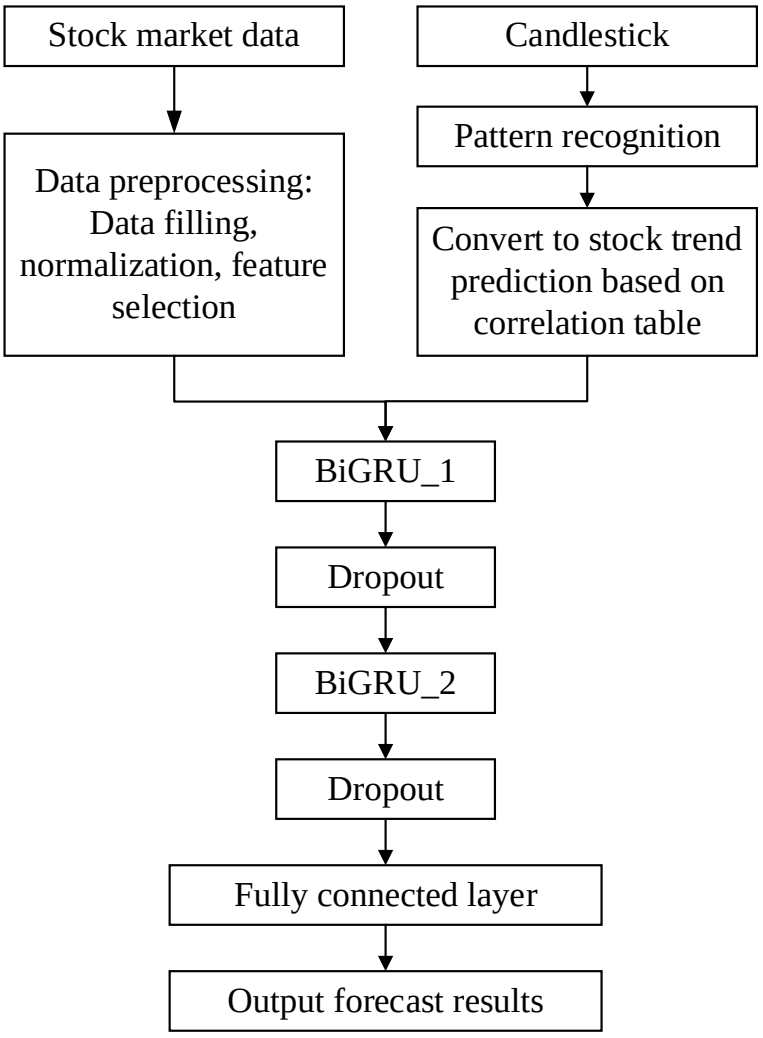


Figure 5. Flowchart for CPBiGRU.

3.3. Principle of Sparrow Search Algorithm

SSA is a novel swarm intelligence optimization technique. The algorithm simulates the foraging behavior of sparrow populations, dividing the sparrow population into discoverers and joiners. SSA calculates the fitness value of sparrows through the constructed fitness function, thus achieving the role and position transformation between individual sparrows, effectively avoiding the issue of traditional optimization algorithms easily getting stuck in local optima.

The population composed of n sparrows can be expressed as follows: where X is a randomly initialized sparrow population, x is an individual sparrow, d represents the dimensionality of the population, and n is the number of sparrows.

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,d} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,d} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n,1} & x_{n,2} & \cdots & x_{n,d} \end{bmatrix} \tag{6}$$

The fitness values of all sparrows can be expressed as follows: where F_x is the fitness matrix, f is the fitness value, expressed as the Root Mean Squared Error (RMSE) between the stock price prediction data and the real price data.

$$F_x = \begin{bmatrix} f([x_{1,1}\ x_{1,2} \ldots\ x_{1,d}]) \\ f([x_{2,1}\ x_{2,2} \ldots\ x_{2,d}]) \\ \vdots \\ \vdots \\ f([x_{n,1}\ x_{n,2} \ldots\ x_{n,d}]) \end{bmatrix} \tag{7}$$

The discoverers search for food and provide foraging direction to all joiners. During each iteration, the formula for the discoverer update location can be described using the following equation: where t is the current iteration number, $j = 1,\ 2,\ 3, \ldots, d$, $iter_{max}$ is the maximum number of iterations and is a constant, $X_{i,j}$ represents the position information of the i-th sparrow in the j-th dimension, α is a random number in [0, 1]. Q is a random number that follows a normal distribution, and L is a $1 \times d$ matrix in which each element is 1. $R_2 (R_2 \in [0,\ 1])$ and $ST (ST \in [0.5,\ 1.0])$ are the alarm value and the safe value, respectively. When $R_2 < ST$, it indicates the absence of predators in the environment, allowing the discoverers to conduct extensive searches. When $R_2 \geq ST$, it indicates that certain individuals within the population have detected predators and they will immediately emit an alarm signal. Following the reception of these alarm signals, the population is required to move to a secure location.

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^{t} \cdot \exp\left(\frac{-i}{\alpha \cdot iter_{max}}\right) & R_2 < ST \\ X_{i,j}^{t} + Q \cdot L & R_2 \geq ST \end{cases} \tag{8}$$

During the foraging process, joiners continuously revise their positions in order to acquire food while simultaneously keeping an eye on discovers and competing with them for food. Equation (9) outlines the approach used by joiners to update their positions. When $i > n/2$, it indicates that the i-th joiner with a lower fitness value has not obtained food and needs to fly to other places to search for more. Where X_P represents the location of the optimal discoverer, X_{worst} represents the current global worst position, n is the population size, while A is a $1 \times d$ matrix where each element is randomly assigned as 1 or -1, and $A^+ = A^T \left(AA^T\right)^{-1}$.

$$X_{i,j}^{t+1} = \begin{cases} Q \cdot \exp\left(\frac{X_{worst}^{t} - X_{i,j}^{t}}{i^2}\right) & i > \frac{n}{2} \\ X_P^{t+1} + \left|X_{i,j}^{t} - X_P^{t+1}\right| \cdot A^{+} \cdot L & i \leq \frac{n}{2} \end{cases} \tag{9}$$

In the algorithm, it is assumed that between 10% and 20% of sparrows in the population will become aware of danger and emit alarm signals when the danger occurs. The initial positions of these sparrows are randomly generated in the population, and their positions are updated based on Equation (10). Where X_{best} represents the current global optimal position, β is a step size control parameter following a normal distribution with a mean of 0 and a variance of 1. K is a random number in $[-1, 1]$ that indicates the direction of sparrow movement and also serves as a step size control parameter. Additionally, f_i represents the fitness value of the i-th sparrow individual, while f_g, f_w represent the current global optimal and worst fitness values, respectively, and ε is an extremely small constant to avoid the error of zero division.

$$X_{i,j}^{t+1} = \begin{cases} X_{best}^{t} + \beta \cdot \left|X_{i,j}^{t} - X_{best}^{t}\right| & f_i > f_g \\ X_{i,j}^{t} + K \cdot \left(\frac{\left|X_{i,j}^{t} - X_{worst}^{t}\right|}{(f_i - f_w) + \varepsilon}\right) & f_i = f_g \end{cases} \tag{10}$$

When $f_i > f_g$, it means that this sparrow is at the edge of the population and is extremely vulnerable to predators. When $f_i = f_g$, it indicates that the sparrow in the middle of the population perceives the danger and needs to approach the other sparrows in time in order to reduce the risk of predation.

The flowchart of the SSA-CPBiGRU model is shown in Figure 6. It can be divided into the following five steps:

1. Initialization: We take the learning rate, iteration times and the number of units in two hidden layers of the CPBiGRU network as the hyperparameter objectives to be optimized by the SSA. The position information of the population and related parameters are randomly initialized after setting the value range of the optimized hyperparameter, the population size of sparrows, optimization iteration times, and initial safety threshold value;
2. Fitness value: We use the RMSE between the predicted value of the network model and the real value as the fitness function for SSA and the loss function for CPBiGRU, to determine the fitness values of each sparrow;
3. Update: We update the sparrow position by Equations (8)–(10) and obtain the fitness value of the sparrow population. Simultaneously, we record the optimal individual position and global optimal position value in the population;
4. Iteration: We ascertain if the maximum value of the number of update iterations has been attained. In such a case, conclude the loop and yield the optimal individual solution, signifying the determination of the optimal parameters for the network structure. Otherwise, return to step (3);
5. Optimization results output: The optimal hyperparameter values output by the SSA algorithm are employed as the learning rate, iteration number, and number of units in two hidden layers of the CPBiGRU network. Afterwards, the network is reconstructed and we proceed with subsequent procedures such as inverse normalization and evaluation analysis.

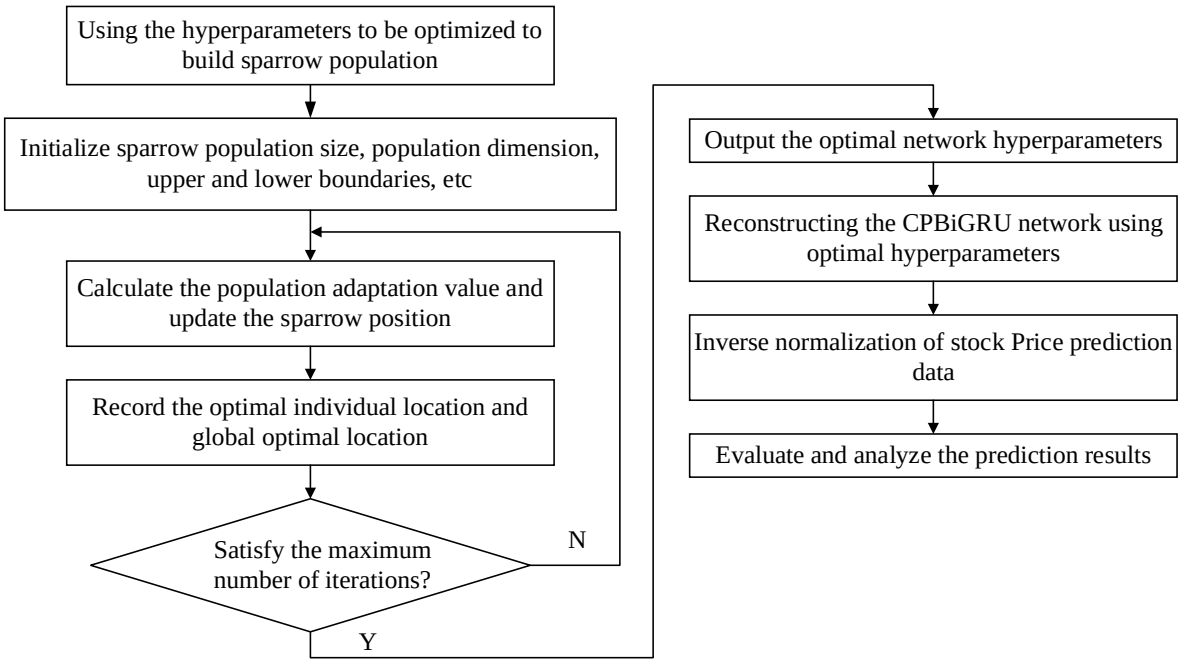


Figure 6. Flowchart for SSA-CPBiGRU.

4. Experiments and Discussions

4.1. Experimental Environment

The operating system of the experimental equipment is Windows 11, the graphics card is Nvidia GeForce RTX 3060, the code is written in Python language, and the prediction model is simulated and experimented with in the Keras framework, which is integrated with Tensorflow 2.9. The learning rate, number of iterations, and number of units in two hidden layers of the CPBiGRU network are obtained by SSA optimization. The search ranges for the hyperparameters are as follows: the learning rate is defined between [0.001, 0.01], the search range for the number of iterations is set as [10, 100], and the number of units in the hidden layers is between [1, 100]. The settings for other hyperparameters are outlined in Table 2.

Table 2. Model hyperparameter settings.

Hyperparameter	Value
Batch size	128
Optimizer	Adam
Number of hidden layer units	2
SSA population size	10
SSA iterations	20
SSA discoverer ratio	0.2

4.2. Datasets

The dataset used in this article consists of two parts: stock market data and stock trend prediction value, both collected on a daily basis, with a time span from 1 January 2017 to 1 August 2022.

To ensure the accuracy of the experimental data source, the stock market data are downloaded from the BaoStock data interface package in Python. In order to verify the universality and effectiveness of the model, this experiment selected six representative stocks from different sectors and industries as the research objects, as shown in Table 3.

Table 3. Stock dataset statistics.

Stock	Plate	Sector
SH.600000	Main Board of the Shanghai Stock Exchange	Bank
SH.603589	Main Board of the Shanghai Stock Exchange	Liquor
SH.600104	Main Board of the Shanghai Stock Exchange	Automobile
SZ.300294	Shenzhen Stock Exchange GEM	Medicine
SZ.002415	Shenzhen Stock Exchange SME	Computer
SZ.000725	Main Board of the Shenzhen Stock Exchange	Electron

The Pearson correlation coefficient is a method used to measure the correlation between two variables, and its values are in the range of $[-1, 1]$ [41]. The formula is represented as Equation (11). Where $\rho_{X,Y}$ is the correlation coefficient, σ_X and σ_Y are the standard covariance of variables X and Y, respectively. Additionally, $cov(X, Y)$ is the covariance between X and Y, and μ_X and μ_Y represent the means of X and Y, respectively. When $\rho_{X,Y} > 0$, it indicates a positive correlation between X and Y. When $\rho_{X,Y} < 0$, it indicates a negative correlation between X and Y. By computing the Pearson correlation coefficients between various market data, a heatmap of the stock market data, as shown in Figure 7, can be generated. From the heatmap, it is evident that the price-to-cashflow ratio (pcfNcfTTM) is negatively correlated with the closing price (close). The objective of this experiment is to predict the closing price of an individual stock. Therefore, we selected 11 kinds of stock market data, including open, high, low, close, pctChg, volume, turn, amount, psTTM, peTTM, and pbMRQ.

$$\rho_{X,Y} = \frac{cov(X, Y)}{\sigma_X \sigma_Y} = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y} \tag{11}$$

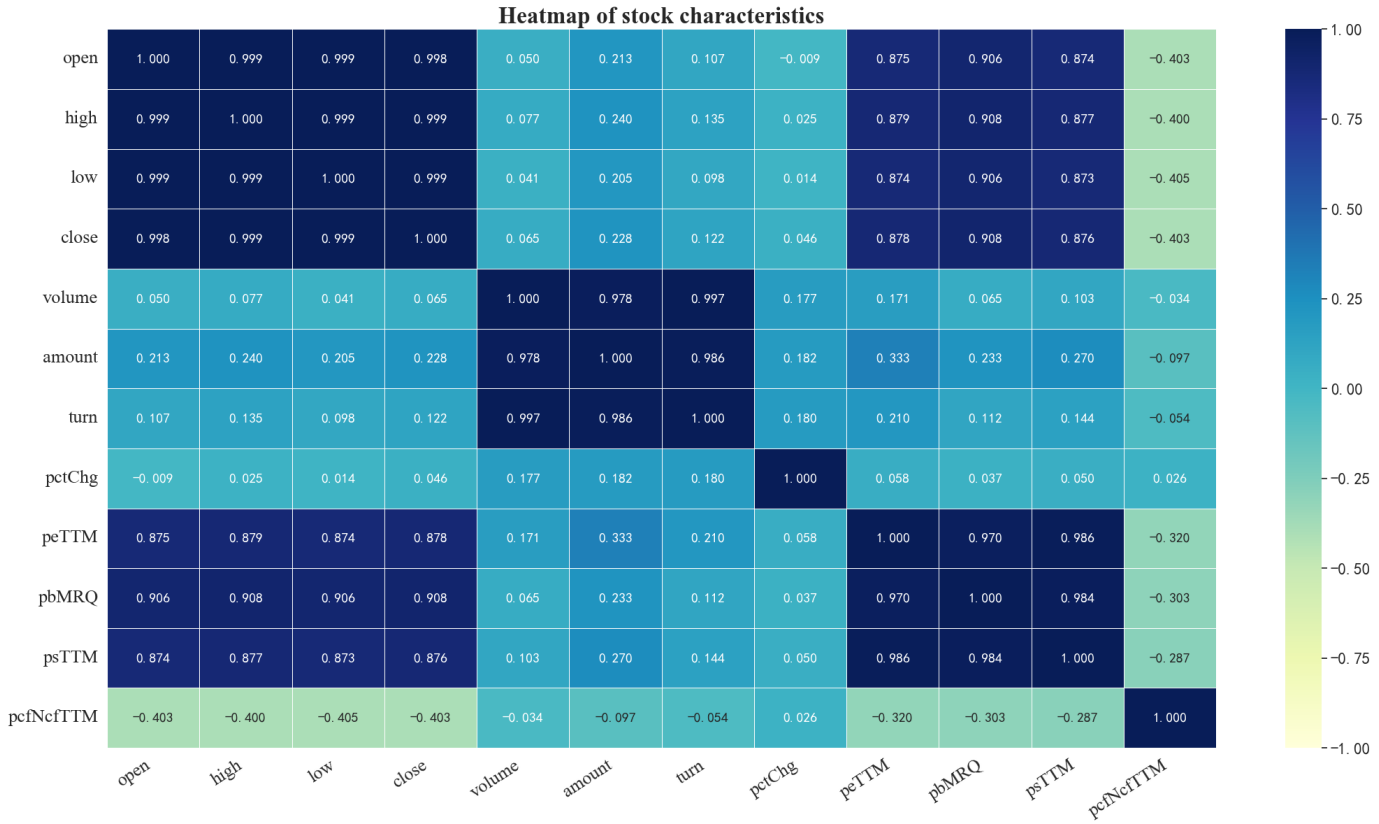


Figure 7. A heatmap of stock market data.

At the same time, the daily trend prediction value of the target stock is obtained based on the correlation table of stock trend prediction values. The stock market data and the trend prediction value are synchronized based on trading time, serving as input data to train SSA-CPBiGRU network for forecasting the closing price of stocks on the following day. The dataset comprises a total of 1356 days, arranged in chronological order. The top 65%, 10%, and the remaining 25% of the dataset are respectively allocated as the training set, validation set, and test set.

4.3. Data Preprocessing

4.3.1. Missing Value Processing

The obtained stock data may have the problem of missing data and suspension of trading, and these missing data may have an impact on stock price prediction. We employ the mean-filling method to fill in missing data. For the data on the suspension date, we adopt the data from the day before the suspension date for filling, thereby ensuring the integrity of the data.

4.3.2. Data Standardization

Due to the large magnitude of stock data and the different magnitudes of input features, directly inputting them into the network model will lead the model bias towards features with large magnitudes, resulting in features with smaller magnitudes not being effective. Therefore, in order to eliminate the effect of different magnitudes between features, the paper uses the maximum minimum standardization method to normalize the stock data to the range of [0, 1]. The formula is as follows:

$$y_i^{'} = \frac{y_i - y_{min}}{y_{max} - y_{min}} \tag{12}$$

where y_i is the denoised stock data, $y_i^{'}$ is the normalized data as the input value of the model, y_{max} and y_{min} are the maximum and minimum values in the data, respectively.

$$y = y^*(y_{max} - y_{min}) + y_{min} \tag{13}$$

where y is the predicted value of the inverse normalization of the model, y^* is the predicted value of the model, y_{min} is the minimum value of the original data in the test set, and y_{max} is the maximum value of the original data in the test set.

4.4. Evaluation Criteria

In order to show the prediction effect of each model, we evaluate the performance of each model by using four metrics: Mean Absolute Percentage Error (MAPE), RMSE, Mean Absolute Error (MAE), and Coefficient of Determination (R^2). The smaller the values of MAPE, RMSE and MAE, the higher the accuracy of the prediction model and the better the prediction effect. As the value of R^2 approaches closer to one, the model's fitting performance grows increasingly superior. The formulas for the four indicators are defined by Equations (14)–(17). Where y_i is the true value of stock closing price, $y_i^{'}$ is the predicted value of stock closing price, $\overline{y}$ is the average value of the true value of the sample, and n is the sample size of the test dataset.

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - y_i^{'}}{y_i}\right| \tag{14}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_i^{'}\right)^2} \tag{15}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - y_i^{'}\right| \tag{16}$$

$$R^2 = 1 - \frac{\sum_i \left(y_i{}' - y_i\right)^2}{\sum_i (\overline{y} - y_i)^2} \tag{17}$$

4.5. Comparing the Prediction Accuracy

4.5.1. Experimentation and Analysis of Stock Price Prediction Model

In order to better evaluate the performance of the models and examine the effectiveness of incorporating candlestick patterns into stock price prediction, LSTM, GRU, BiGRU, CPBiGRU, SSA-BiGRU and SSA-CPBiGRU models are selected for comparative experiments in this paper. To mitigate the element of chance in the experiments and ensure the reliability of the experimental results, we conducted five experiments for each model to obtain the average value. The specific experimental results are shown in Table 4, where the bolded data represent the best predicted experimental results. The prediction results of each model on all stock data test sets selected for the experiment are shown in Figures 8–13.

Table 4. Predictive performance of different models.

Model	Stock	MAPE	RMSE	MAE	R^2
LSTM	SH.600000	0.0137	0.1489	0.1178	0.9756
	SH.603589	0.0277	2.3260	1.7570	0.9264
	SH.600104	0.0168	0.4178	0.3157	0.9392
	SZ.300294	0.0337	1.6722	1.2259	0.9088
	SZ.002415	0.0311	2.1352	1.6488	0.9538
	SZ.000725	0.0332	0.1845	0.1650	0.9649
GRU	SH.600000	0.0120	0.1313	0.1080	0.9810
	SH.603589	0.0262	2.2027	1.6451	0.9340
	SH.600104	0.0142	0.3813	0.2714	0.9493
	SZ.300294	0.0325	1.6306	1.1828	0.9132
	SZ.002415	0.0295	2.0208	1.5643	0.9585
	SZ.000725	0.0316	0.1802	0.1594	0.9665
BiGRU	SH.600000	0.0102	0.1152	0.0906	0.9854
	SH.603589	0.0249	2.1920	1.5678	0.9347
	SH.600104	0.0139	0.3720	0.2688	0.9518
	SZ.300294	0.0319	1.6146	1.1758	0.9149
	SZ.002415	0.0232	1.5993	1.2063	0.9741
	SZ.000725	0.0235	0.1421	0.1205	0.9792
CPBiGRU	SH.600000	0.0094	0.1079	0.0834	0.9871
	SH.603589	0.0245	2.1829	1.5543	0.9352
	SH.600104	0.0137	0.3657	0.2641	0.9536
	SZ.300294	0.0304	1.6077	1.1262	0.9157
	SZ.002415	0.0216	1.4820	1.1078	0.9777
	SZ.000725	0.0191	0.1248	0.0990	0.9839
SSA-BiGRU	SH.600000	0.0086	0.1066	0.0768	0.9875
	SH.603589	0.0241	2.0366	1.4912	0.9433
	SH.600104	0.0136	0.3639	0.2619	0.9539
	SZ.300294	0.0293	1.4882	1.0839	0.9277
	SZ.002415	0.0209	1.4449	1.0710	0.9788
	SZ.000725	0.0177	0.1160	0.0918	0.9861
SSA-CPBiGRU	SH.600000	**0.0077**	**0.0989**	**0.0689**	**0.9892**
	SH.603589	**0.0235**	**2.0330**	**1.4685**	**0.9439**
	SH.600104	**0.0134**	**0.3639**	**0.2601**	**0.9542**
	SZ.300294	**0.0282**	**1.4572**	**1.0290**	**0.9307**
	SZ.002415	**0.0193**	**1.3275**	**0.9654**	**0.9821**
	SZ.000725	**0.0155**	**0.1112**	**0.0834**	**0.9872**

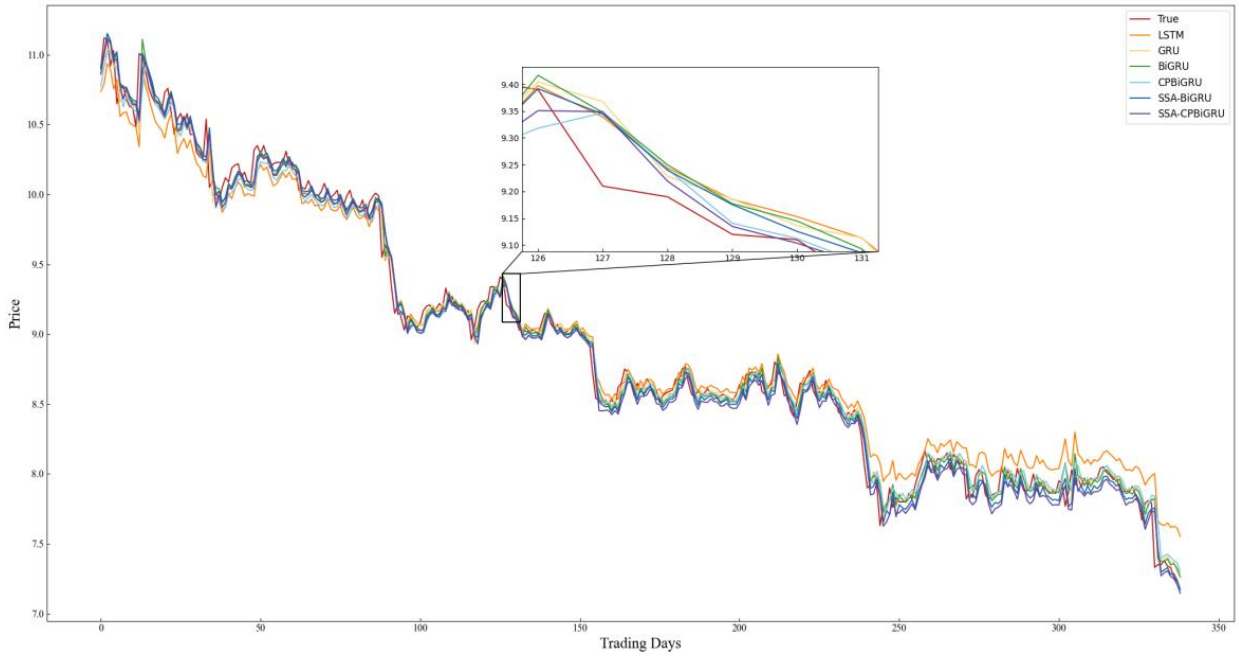


Figure 8. Comparison chart of SH.600000 prediction results.

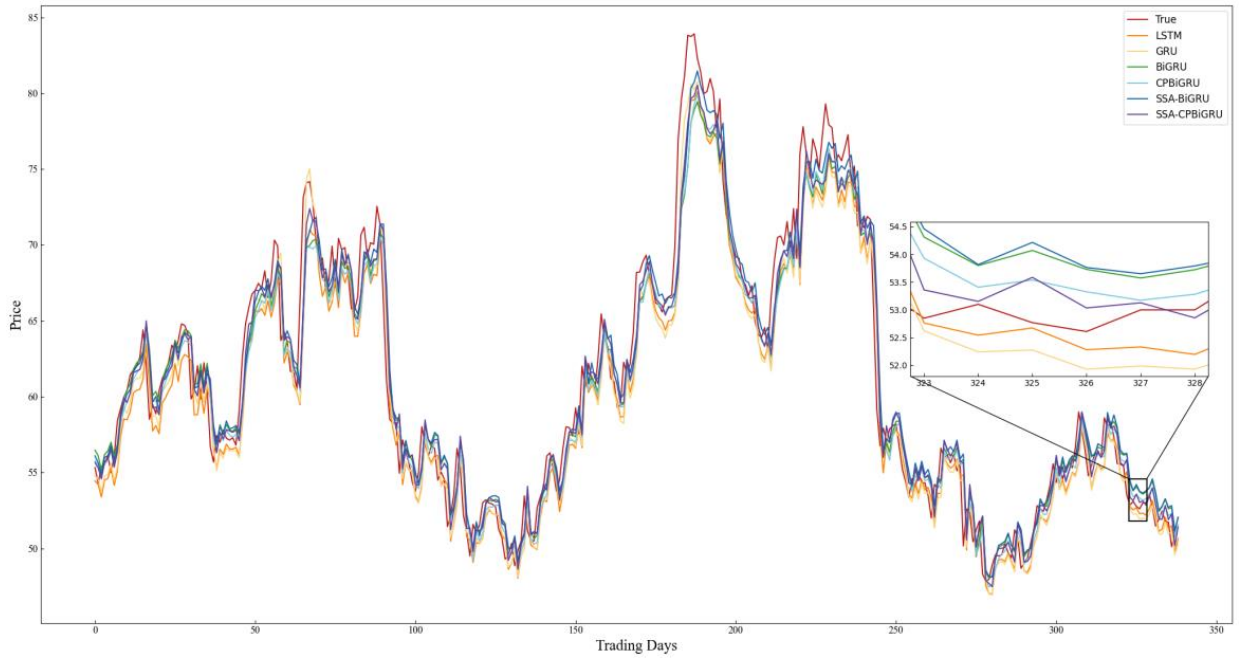


Figure 9. Comparison chart of SH.603589 prediction results.

Combining the prediction results in Table 4 and Figures 8–13, it can be seen that the SSA-CPBiGRU model has a better prediction performance than other models. From Table 4, it can be seen that the BiGRU model has better prediction results than LSTM and GRU. In comparison to the LSTM model, the BiGRU model reduces the test loss by an average reduction of 18.81%, 15.15%, and 17.77% in terms of MAPE, RMSE and MAE, respectively. Additionally, there is an average improvement of 1.24% in R^2. This is mainly due to the fact that BiGRU improves the problem of data time dependence and increases the number of neural units, which leads to more accurate prediction results.

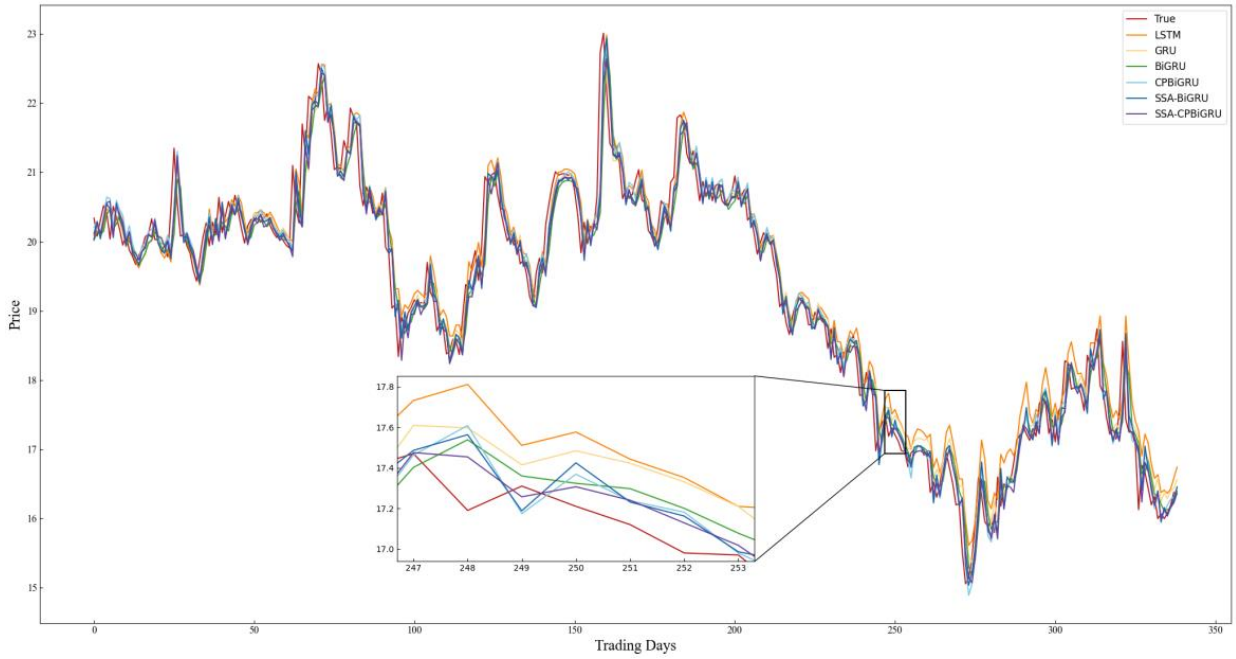


Figure 10. Comparison chart of SH.600104 prediction results.

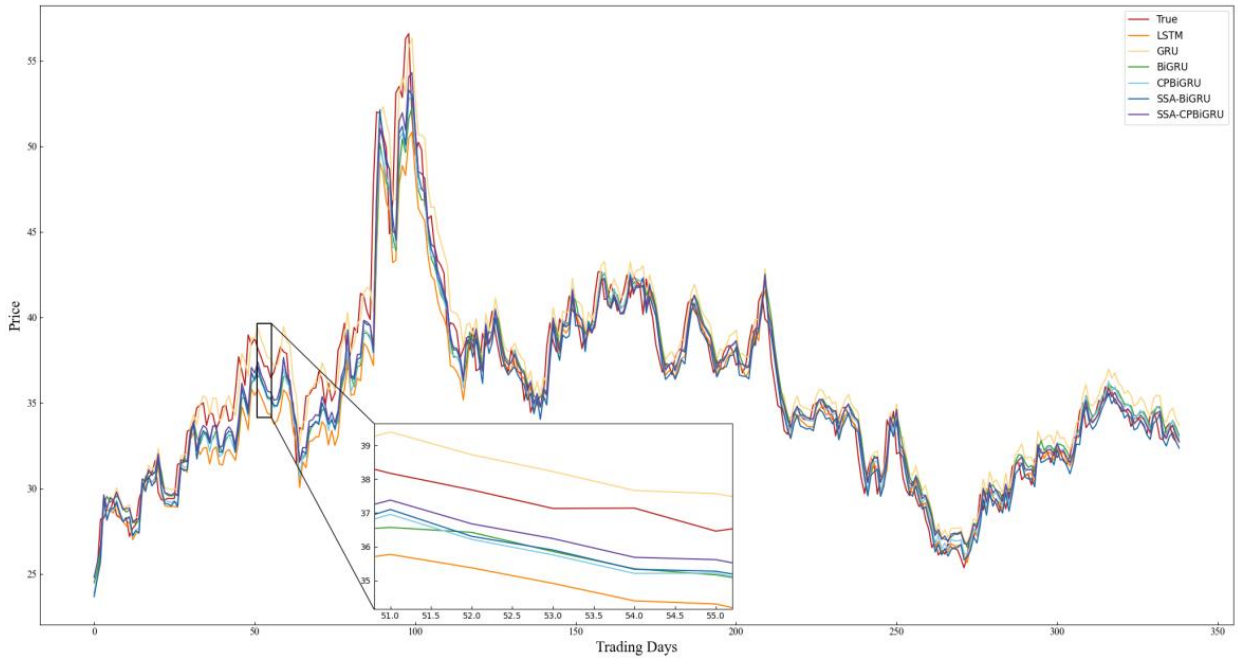


Figure 11. Comparison chart of SZ.300294 prediction results.

Compared with the single LSTM, GRU and BiGRU models, the CPBiGRU model exhibits a significant improvement in terms of MAPE, RMSE, and MAE. In comparison to the BiGRU model, the CPBiGRU model produces an average of 6.87%, 4.73%, and 6.80% less test loss in terms of MAPE, RMSE and MAE, respectively. This indicates that the addition of candlestick patterns contributes to an enhancement in stock price forecasting performance.

In addition, SSA-BiGRU and SSA-CPBiGRU, which use SSA to optimize the hyperparameters of the model, outperform the model with manually defined hyperparameter values in all evaluation criteria. Compared to the CPBiGRU model, the SSA-CPBiGRU model reduces the test loss by an average reduction of 10.18%, 7.73%, and 10.28% in terms of MAPE, RMSE and MAE, respectively. This indicates that the utilization of SSA can help to improve the prediction accuracy of the model.

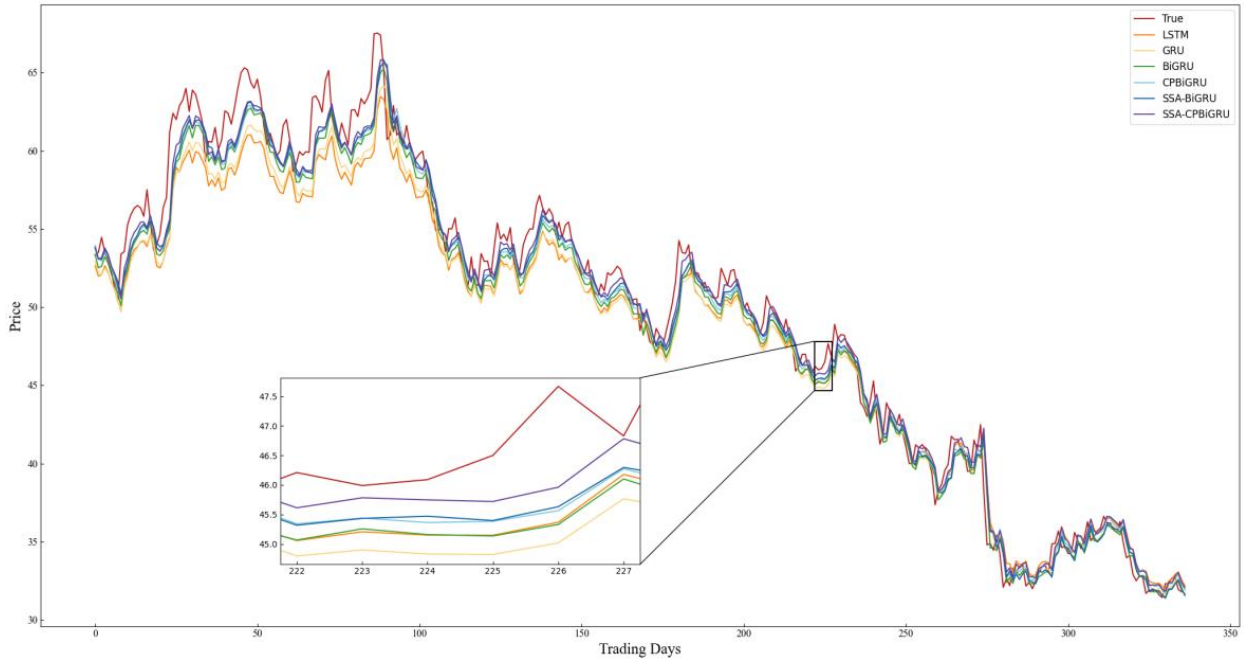


Figure 12. Comparison chart of SZ.002415 prediction results.

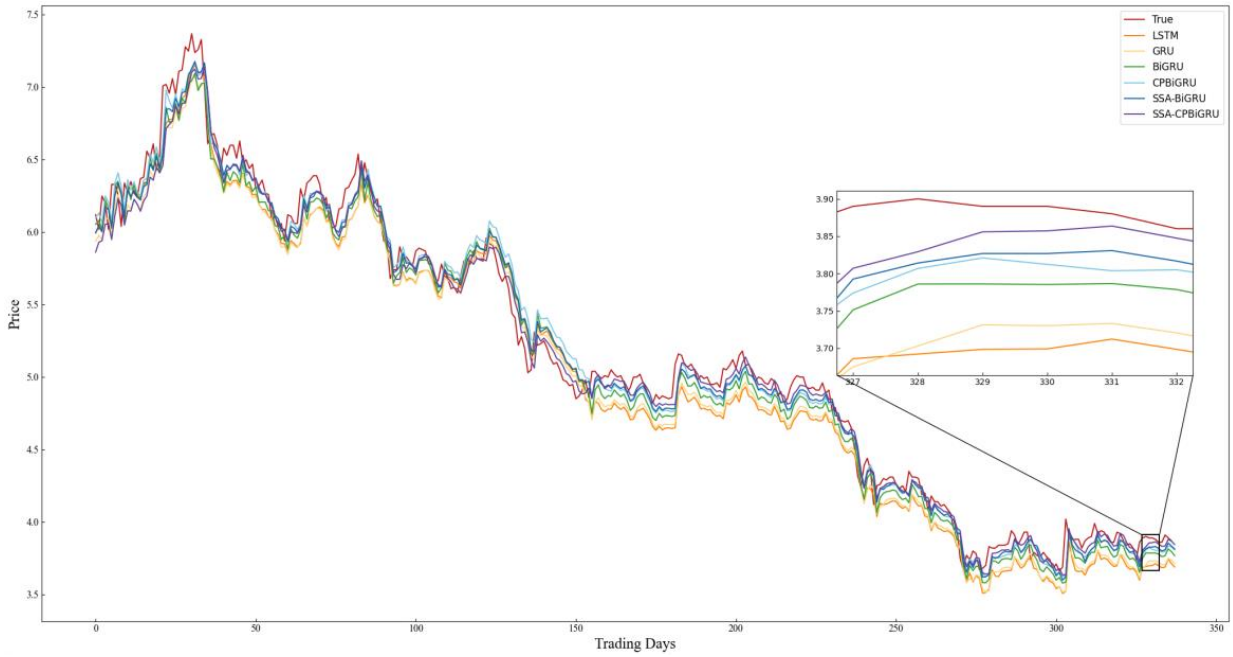


Figure 13. Comparison chart of SZ.000725 prediction results.

The SSA-CPBiGRU model proposed in this paper outperformed the previous five models in predicting the closing prices of stocks in six different sectors and industries. In comparison to the BiGRU model, the SSA-CPBiGRU model produces an average of 16.03%, 12.01%, and 16.13% less test loss in terms of MAPE, RMSE and MAE, respectively, and it also achieves an average improvement of 0.82% in R^2. Moreover, when compared to the GRU model, the SSA-CPBiGRU model reduces the test loss by an average reduction of 25.09%, 20.03%, and 25.01% in terms of MAPE, RMSE and MAE, respectively. Additionally, there is an average improvement of 1.46% in R^2. In comparison to LSTM model, the SSA-CPBiGRU model exhibits an average reduction of 31.13%, 24.92%, and 30.42% in test loss in terms of MAPE, RMSE, and MAE, respectively. Additionally, it achieves an average enhancement of 2.05% in R^2.

Based on the above analysis, the SSA-CPBiGRU model demonstrates superior efficacy in forecasting stock prices compared to traditional network models, exhibiting reduced prediction errors and enhanced fitting capabilities.

4.5.2. Experimentation and Analysis of the Effect of Trading Day on Stock Price Prediction

Based on the above experiments, we aim to determine how long the trading day of stock data can achieve the optimal stock price prediction effect. In Table 5, SSA-CPBiGRU (1), SSA-CPBiGRU (2), SSA-CPBiGRU (3), SSA-CPBiGRU (5), and SSA-CPBiGRU (10) represent the use of stock data from the first trading day, the first two trading days, the first three trading days, the first five trading days, and the first ten trading days of the predicted stock price date in the model for prediction. Bolded data in the table denote the experimental results of the optimal prediction.

Table 5. Predictive performance of different trading days.

Model	Stock	MAPE	RMSE	MAE	R^2
SSA-CPBiGRU (1)	SH.600000	0.0095	0.1149	0.0835	0.9854
	SH.603589	**0.0235**	**2.0330**	**1.4685**	**0.9439**
	SH.600104	**0.0134**	**0.3639**	**0.2601**	**0.9542**
	SZ.300294	**0.0282**	**1.4572**	**1.0290**	**0.9307**
	SZ.002415	0.0237	1.6354	1.2313	0.9728
	SZ.000725	0.0278	0.1597	0.1395	0.9737
SSA-CPBiGRU (2)	SH.600000	**0.0079**	**0.0996**	**0.0710**	**0.9892**
	SH.603589	0.0253	2.2558	1.6093	0.9308
	SH.600104	0.0139	0.3626	0.2662	0.9539
	SZ.300294	0.0303	1.5543	1.1087	0.9212
	SZ.002415	0.0222	1.4902	1.1234	0.9774
	SZ.000725	0.0261	0.1521	0.1294	0.9761
SSA-CPBiGRU (3)	SH.600000	0.0087	0.1071	0.0770	0.9873
	SH.603589	0.0255	2.2630	1.6195	0.9304
	SH.600104	0.0140	0.3739	0.2711	0.9513
	SZ.300294	0.0300	1.6116	1.1171	0.9145
	SZ.002415	0.0219	1.5373	1.1346	0.9759
	SZ.000725	0.0176	0.1212	0.0941	0.9849
SSA-CPBiGRU (5)	SH.600000	0.0098	0.1180	0.0868	0.9845
	SH.603589	0.0288	2.5816	1.8620	0.9095
	SH.600104	0.0142	0.3702	0.2725	0.9523
	SZ.300294	0.0346	1.9086	1.3105	0.8811
	SZ.002415	0.0212	1.3392	1.0357	0.9818
	SZ.000725	**0.0155**	**0.1112**	**0.0834**	**0.9872**
SSA-CPBiGRU (10)	SH.600000	0.0107	0.1277	0.0939	0.9816
	SH.603589	0.0276	2.4660	1.7691	0.9176
	SH.600104	0.0154	0.3934	0.2932	0.9463
	SZ.300294	0.0354	1.7844	1.3038	0.8944
	SZ.002415	**0.0193**	**1.3275**	**0.9654**	**0.9821**
	SZ.000725	0.0277	0.1609	0.1401	0.9733

As can be seen from Table 5, SH.603589, SH.600104, SH.300294 and SH.600000 exhibit enhanced proficiency in predicting stock price movements when employing data from the first and the first two trading days. This is due to the rapid fluctuations in the prices of these stocks, where utilizing longer periods of stock data does not provide assistance and may even contribute a negative interference to stock price prediction. SZ.000725 performs better when using data from the first five trading days for prediction. SZ.002415 achieves better prediction results by employing data from the first ten trading days. This indicates that stocks with relatively moderate price changes are better suited for using data from longer time windows for forecasting, which can better reflect the market trends.

5. Conclusions

In this paper, we propose a stock price forecasting model SSA-CPBiGRU, which integrates candlestick patterns and SSA to predict the next day's closing price. In order to enrich the input data source, this paper uses stock market data as input data and integrates candlestick patterns to make the input data have structural characteristics and temporal relationships. At the same time, SSA is employed to optimize the hyperparameters of the model, solving the problem of high randomness in hyperparameter selection and further improving the prediction accuracy of the model. Experimental analysis is conducted on the data from six different sectors and industries of stocks in recent 5 years, employing six models: LSTM, GRU, BiGRU, CPBiGRU, SSA-BiGRU, and SSA-CPBiGRU. The results demonstrate that the SSA-CPBiGRU model outperforms the traditional models in terms of smaller prediction errors and better fitting effects. Compared to the LSTM model, the SSA-CPBiGRU model reduces the test loss by an average of 31.13%, 24.92%, and 30.42% in terms of MAPE, RMSE, and MAE, respectively. Additionally, it achieves an average improvement of 2.05% in R^2. Furthermore, for stocks with rapid price changes, shorter time window data yield better predictive outcomes, while for stocks with gentle price changes, the data with longer time windows are more effective in forecasting. The SSA-CPBiGRU model possesses the capability to swiftly and accurately grasp the intricacies of data characteristics, thereby achieving precision in forecasting stock prices. This model has the potential to serve as a decision-making tool for investors, mitigating investment risks to some extent. Additionally, the model exhibits efficient processing abilities for time series data, thereby presenting potential applicability to other time series problems.

In future research, we will focus on two aspects. Firstly, in terms of prediction model, we intend to incorporate the attention mechanisms and improve the SSA algorithm, thus further reducing the risk of getting trapped in local optima and elevating the algorithm's capacity for global exploration. Secondly, in relation to the input data sources, we can improve the fusion method of candlestick patterns and introduce more characteristic factors that possess the ability to influence stock price trends. At the same time, we can take into consideration the particularity of trading days, such as the impact of events like the Spring Festival and National Day, on the fluctuations of stock prices, thereby enhancing the predictive performance of the model.

Author Contributions: Conceptualization, X.C. and W.H.; methodology, X.C.; software, X.C.; validation, X.C., W.H. and L.X.; formal analysis, X.C.; investigation, X.C.; resources, X.C., W.H. and L.X.; data curation, X.C.; writing—original draft preparation, X.C.; writing—review and editing, X.C., W.H. and L.X.; visualization, W.H.; supervision, W.H.; project administration, L.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The stock market data used in this paper is sourced from BaoStock (www.baostock.com), spanning from 1 January 2017 to 1 August 2022. Accessed on 20 September 2022.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Chung, H.; Shin, K.S. Genetic algorithm-optimized multi-channel convolutional neural network for stock market prediction. *Neural Comput. Appl.* **2020**, *32*, 7897–7914. [CrossRef]
2. Pang, X.W.; Zhou, Y.Q.; Wang, P.; Lin, W.W.; Chang, V. An innovative neural network approach for stock market prediction. *J. Supercomput.* **2020**, *76*, 2098–2118. [CrossRef]
3. Rounaghi, M.M.; Zadeh, F.N. Investigation of market efficiency and Financial Stability between S&P 500 and London Stock Exchange: Monthly and yearly Forecasting of Time Series Stock Returns using ARMA model. *Phys. A Stat. Mech. Its Appl.* **2016**, *456*, 10–21. [CrossRef]
4. Jarrett, J.E.; Kyper, E. ARIMA Modeling with Intervention to Forecast and Analyze Chinese Stock Prices. *Int. J. Eng. Bus. Manag.* **2011**, *3*, 17. [CrossRef]
5. Dempere, J.M.; Modugu, K.P. The impact of the Dubai International Airport's activity volume on the Emirati stock market. *Int. J. Bus. Perform. Manag.* **2022**, *23*, 118–134. [CrossRef]

6. Lin, A.J.; Shang, P.J.; Zhou, H.C. Cross-correlations and structures of stock markets based on multiscale MF-DXA and PCA. *Nonlinear Dyn.* **2014**, *78*, 485–494. [CrossRef]
7. Patel, J.; Shah, S.; Thakkar, P.; Kotecha, K. Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques. *Expert Syst. Appl.* **2015**, *42*, 259–268. [CrossRef]
8. Nti, I.K.; Adekoya, A.F.; Weyori, B.A. Random Forest Based Feature Selection of Macroeconomic Variables for Stock Market Prediction. *Am. J. Appl. Sci.* **2019**, *16*, 200–212. [CrossRef]
9. Fu, S.B.; Li, Y.W.; Sun, S.L.; Li, H.T. Evolutionary support vector machine for RMB exchange rate forecasting. *Phys. A-Stat. Mech. Its Appl.* **2019**, *521*, 692–704. [CrossRef]
10. Xu, Y.; Yan, C.J.; Peng, S.L.; Nojima, Y. A hybrid two-stage financial stock forecasting algorithm based on clustering and ensemble learning. *Appl. Intell.* **2020**, *50*, 3852–3867. [CrossRef]
11. Zhang, G.P. Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing* **2003**, *50*, 159–175. [CrossRef]
12. Yu, Z.X.; Qin, L.; Chen, Y.J.; Parmar, M.D. Stock price forecasting based on LLE-BP neural network model. *Phys. A-Stat. Mech. Its Appl.* **2020**, *553*, 124197. [CrossRef]
13. Udagawa, Y. Predicting Stock Price Trend Using Candlestick Chart Blending Technique. In Proceedings of the 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 10–13 December 2018; pp. 4709–4715.
14. Xue, J. Research and Application of A Novel Swarm Intelligence Optimization Technique: Sparrow Search Algorithm. Master's Thesis, Donghua University, Shanghai, China, 2020.
15. Tao, L.; Hao, Y.T.; Hao, Y.J.; Shen, C.F. K-Line Patterns' Predictive Power Analysis Using the Methods of Similarity Match and Clustering. *Math. Probl. Eng.* **2017**, *2017*, 3096917. [CrossRef]
16. Li, H.B.; Liang, M.X.; He, T. Optimizing the Composition of a Resource Service Chain With Interorganizational Collaboration. *IEEE Trans. Ind. Inform.* **2017**, *13*, 1152–1161. [CrossRef]
17. Iqbal, M.; Pao, H.-K. Mining non-redundant distinguishing subsequence for trip destination forecasting. *Knowl. -Based Syst.* **2021**, *211*, 106519. [CrossRef]
18. Li, H.; Li, Z.; Peng, S.; Li, J.; Tungom, C.E. Mining the frequency of time-constrained serial episodes over massive data sequences and streams. *Future Gener. Comput. Syst.* **2020**, *110*, 849–863. [CrossRef]
19. Wang, W.; Tian, J.; Lv, F.; Xin, G.; Ma, Y.; Wang, B. Mining frequent pyramid patterns from time series transaction data with custom constraints. *Comput. Secur.* **2021**, *100*, 102088. [CrossRef]
20. Birogul, S.; Temür, G.; Kose, U. YOLO Object Recognition Algorithm and "Buy-Sell Decision" Model Over 2D Candlestick Charts. *IEEE Access* **2020**, *8*, 91894–91915. [CrossRef]
21. Guo, S.J.; Hung, C.C.; Hsu, F.C.; IEEE. Deep Candlestick Predictor: A Framework Toward Forecasting the Price Movement from Candlestick Charts. In Proceedings of the 9th International Conference on Parallel Architectures, Algorithms and Programming (PAAP), Natl Taiwan Univ Sci & Technol, Taipei, Taiwan, 26–28 December 2018; pp. 219–226.
22. Chen, J.H.; Tsai, Y.C. Encoding candlesticks as images for pattern classification using convolutional neural networks. *Financ. Innov.* **2020**, *6*, 26. [CrossRef]
23. Fengqian, D.; Chao, L. An Adaptive Financial Trading System Using Deep Reinforcement Learning with Candlestick Decomposing Features. *IEEE Access* **2020**, *8*, 63666–63678. [CrossRef]
24. Wang, M.J.; Wang, Y.J. Evaluating the Effectiveness of Candlestick Analysis in Forecasting US Stock Market. In Proceedings of the 3rd International Conference on Compute and Data Analysis (ICCDA), Univ Hawaii Maui Coll, Kahului, HI, USA, 14–17 March 2019; pp. 98–101.
25. Madbouly, M.M.; Elkholy, M.; Gharib, Y.M.; Darwish, S.M. Predicting Stock Market Trends for Japanese Candlestick Using Cloud Model. In Proceedings of the International Conference on Artificial Intelligence and Computer Vision (AICV2020), Cairo, Egypt, 8–10 April 2020; pp. 628–645.
26. Wang, J.; Li, X.H.; Jia, H.D.; Peng, T.; Tan, J.H. Predicting Stock Market Volatility from Candlestick Charts: A Multiple Attention Mechanism Graph Neural Network Approach. *Math. Probl. Eng.* **2022**, *2022*, 4743643. [CrossRef]
27. Rather, A.M.; Agarvval, A.; Sastry, V.N. Recurrent neural network and a hybrid model for prediction of stock returns. *Expert Syst. Appl.* **2015**, *42*, 3234–3241. [CrossRef]
28. Minami, S. Predicting Equity Price with Corporate Action Events Using LSTM-RNN. *J. Math. Financ.* **2018**, *08*, 58–63. [CrossRef]
29. Gupta, U.; Bhattacharjee, V.; Bishnu, P.S. StockNet-GRU based stock index prediction. *Expert Syst. Appl.* **2022**, *207*, 117986. [CrossRef]
30. Kumar Chandar, S.; Sumathi, M.; Sivanandam, S.N. Prediction of Stock Market Price using Hybrid of Wavelet Transform and Artificial Neural Network. *Indian J. Sci. Technol.* **2016**, *9*, 1–5. [CrossRef]
31. Cai, S.; Feng, X.; Deng, Z.; Ming, Z.; Shan, Z. Financial News Quantization and Stock Market Forecast Research Based on CNN and LSTM. In *Smart Computing and Communication*; Springer: Cham, Switzerland, 2018; pp. 366–375.
32. Ho, T.T.; Huang, Y.N. Stock Price Movement Prediction Using Sentiment Analysis and CandleStick Chart Representation. *Sensors* **2021**, *21*, 7957. [CrossRef]
33. Hu, Y. Reserach on Stock Trend Forecasting Method Based on Multi Technical Indicators and Deep Learning Mode. Master's Thesis, Jiangxi University of Finance and Economics, Nanchang, China, 2021; pp. 1–64. [CrossRef]
34. Hinchey, M.G.; Sterritt, R.; Rouff, C. Swarms and Swarm Intelligence. *Computer* **2007**, *40*, 111–113. [CrossRef]

35. Bonabeau, E.; Meyer, C. Swarm intelligence—A whole new way to think about business. *Harv. Bus. Rev.* **2001**, *79*, 106.
36. Song, G.; Zhang, Y.; Bao, F.; Qin, C. Stock prediction model based on particle swarm optimization LSTM. *J. Beijing Univ. Aeronaut. Astronaut.* **2019**, *45*, 2533–2542.
37. Li, Y.; Wang, S.; Chen, Q.; Wang, X. Comparative study of several new swarm intelligence optimization algorithms. *Comput. Eng. Appl.* **2020**, *56*, 1–12. [CrossRef]
38. Liao, G.C. Fusion of Improved Sparrow Search Algorithm and Long Short-Term Memory Neural Network Application in Load Forecasting. *Energies* **2022**, *15*, 130. [CrossRef]
39. Cho, K.; Van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv* **2014**, arXiv:1406.1078.
40. Li, W.Y.; Wu, H.; Zhu, N.Y.; Jiang, Y.N.; Tan, J.L.; Guo, Y. Prediction of dissolved oxygen in a fishery pond based on gated recurrent unit (GRU). *Inf. Process. Agric.* **2021**, *8*, 185–193. [CrossRef]
41. Barnett, L.; Barrett, A.B.; Seth, A.K. Granger Causality and Transfer Entropy Are Equivalent for Gaussian Variables. *Phys. Rev. Lett.* **2009**, *103*, 238701. [CrossRef]

2021

Stock Trend Prediction Using Candlestick Charting and Ensemble Machine Learning Techniques with a Novelty Feature Engineering Scheme

Yaohu Lin

Shancun Liu

Haijun Yang

Harris Wu
Old Dominion University, hwu@odu.edu

Original Publication Citation

Lin, Y., Liu, S., Yang, H., & Wu, H. (2021). Stock trend prediction using candlestick charting and ensemble machine learning techniques with a novelty feature engineering scheme. *IEEE Access*, 9, 101433-101446. https://doi.org/10.1109/access.2021.3096825

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received May 29, 2021, accepted July 7, 2021, date of publication July 13, 2021, date of current version July 23, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3096825

Stock Trend Prediction Using Candlestick Charting and Ensemble Machine Learning Techniques With a Novelty Feature Engineering Scheme

YAOHU LIN[1], SHANCUN LIU[1,2], HAIJUN YANG[1,3], (Member, IEEE), AND HARRIS WU[4]

[1]School of Economics and Management, Beihang University, Beijing 100191, China
[2]Key Laboratory of Complex System Analysis, Management and Decision, Ministry of Education, Beihang University, Beijing 100191, China
[3]Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing 100191, China
[4]Strome College of Business, Old Dominion University, Norfolk, VA 23529, USA

Corresponding author: Haijun Yang (navy@buaa.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 71771006 and Grant 71771008.

ABSTRACT Stock market forecasting is a knotty challenging task due to the highly noisy, nonparametric, complex and chaotic nature of the stock price time series. With a simple eight-trigram feature engineering scheme of the inter-day candlestick patterns, we construct a novel ensemble machine learning framework for daily stock pattern prediction, combining traditional candlestick charting with the latest artificial intelligence methods. Several machine learning techniques, including deep learning methods, are applied to stock data to predict the direction of the closing price. This framework can give a suitable machine learning prediction method for each pattern based on the trained results. The investment strategy is constructed according to the ensemble machine learning techniques. Empirical results from 2000 to 2017 of China's stock market confirm that our feature engineering has effective predictive power, with a prediction accuracy of more than 60% for some trend patterns. Various measures such as big data, feature standardization, and elimination of abnormal data can effectively solve data noise. An investment strategy based on our forecasting framework excels in both individual stock and portfolio performance theoretically. However, transaction costs have a significant impact on investment. Additional technical indicators can improve the forecast accuracy to varying degrees. Technical indicators, especially momentum indicators, can improve forecasting accuracy in most cases.

INDEX TERMS K-line patterns, machine learning, ensemble strategy, eight-trigram, stock forecasting.

I. INTRODUCTION

The forecasting of the stock market is an important objective in the financial world and remains one of the most challenging problems due to the non-linear and chaotic financial nature [1], [2]. Investments in the stock market are often guided by different prediction methods which can be divided into two groups of technical analysis and fundamental analysis [3]. The fundamental analysis approach is concerned with the company which used the economic standing of the firm, employees, yearly reports, financial status, balance sheets, income reports and so on [4]. On the other hand, technical analysis, also called charting, predicts the future by studying the trends from the historical data [5]. Investors could build profitable trading strategies by using technical analysis techniques [6], [7]. Utilizing open-high-low-close prices, candlestick charting can reflect not only the changing balance between supply and demand [8], but also the sentiment of the investors in the market [9].

Nti *et al.* revealed that 66% of stock market prediction documents they reviewed were based on technical analysis [3]. Traditional technical analysis is mostly limited to analyzing the candlestick charting, performing statistical analysis from past historical data to obtain the probability of prediction. For example, Caginalp and Laurent performed a statistical test including eight kinds of three-day patterns and noted that the candlestick patterns have predictive power [10]. Then Lu *et al.* examined these eight three-day patterns with three definitions of trend and four holding strategies in the DJIA component data, and found that regardless of which definition of the trend was used, eight three-day reversal patterns with a Caginalp–Laurent holding strategy were profitable [11]. Chen *et al.* gave the quantitative definitions of four pairs of two-day candlestick patterns to study their predictive

The associate editor coordinating the review of this manuscript and approving it for publication was Diego Oliva.

power in Chinese stock market. Results showed that these two-day candlestick patterns have different predictive capabilities [12]. Zhu *et al.* examined the effectiveness of five different candlestick reversal patterns in Chinese stock market. Statistical analysis suggested that bearish harami, and cross signals perform well in predicting head reversals for stocks of low liquidity, while bullish harami, engulfing, and piercing patterns were profitable when applied to highly liquid, small companies' stocks [13]. Lu examined the predictive power of single-day candlestick charting by using the daily data for the Taiwan stocks for the period from 4 January 1992 to 31 December 2009. Statistical results revealed that four patterns were profitable for the Taiwan stock market after transaction costs [14]. Lv *et al.* testing the predictive power of the Three Inside Up pattern and Three Inside Down pattern with the testing dataset of the K-line series data of Shanghai 180 index component stocks over the latest 10 years [15].

Andrew *et al.* provided evidence that technical analysis can be improved by using automated algorithms [16]. Recently, artificial intelligence (AI) has been applied to address the chaotic time series data [17], [18]. The intense computational use of intelligent predictive models has commonly been studied under the title of machine learning [19]. Machine learning uses historical data for parameter fitting to predict new data [20]. Many machine techniques have already been applied to forecast the stock market. For example, logistic regression (LR) and Neural Network (NNs) [21], [22], deep neural networks (DNN) [23], decision trees (DTs) [24]–[26], support vector machines (SVM) [27], k-nearest neighbors (KNN) [28], random forests (RFs) [29], [30] and long short-term memory networks (LSTMs) [31], [32] have been used to predict the stock market. Moreover, many authors try to improve the prediction ability by combining machine learning models with other methods. Ahmad *et al.* proposed a forecasting model based on chaotic mapping, firefly algorithm and support vector regression to predict stock market price. Compared with genetic algorithm-based SVR, chaotic genetic algorithm-based SVR, artificial neural networks and adaptive neuro-fuzzy inference systems, the proposed model performs best on mean squared error and mean absolute percent error [33]. Zhang *et al.* proposed a stock price predicting system by combining SVR and ensemble adaptive neuro fuzzy inference system (ENANFIS). The experimental results showed that the SVR-ENANFIS model has superior prediction performance than ENANFIS, SVR-Linear, SVR-SVR and SVR-ANN [34]. However, forecast research based on AI methods and candlestick charting is still less.

In these applications of AI methods for financial market forecasting, many studies have used technical indicators as input features. Weng *et al.* developed a financial expert system that incorporated the historical stock prices, eight kinds of technical indicators, counts and sentiment scores of published news articles, trends in Google search and Wikipedia information to predict short term stock prices [35]. Kumar *et al.* used 15 kinds of technical indicators to construct 55 input features to predict the direction of stock indices [36]. Gocken *et al.* used 44 technical indicators in their hybrid soft computing models for 1, 2, 3, 5, 7 and 10 days ahead stock price prediction [37]. Patel *et al.* selected 10 technical indicators to predict the closing price using the fusion of machine learning techniques [38]. Zhou *et al.* developed a learning architecture LR2GBDT for forecasting and trading stock indices by adding 12 technical indicators as the initial variables [39]. Bao *et al.* used 10 technical indicators in a deep learning framework where wavelet transforms (WT), stacked autoencoders (SAEs) and LSTM are combined for stock price forecasting [40]. However, the existing research is limited to using these technical indicators as input parameters and it lacks a further discussion on these technical indicators.

When comparing different machine learning effects, Weng *et al.* used four machine learning methods, including boosted regression tree (BRT), NN, SVM and RF. Results showed that BRT and RF perform the best when predicting one-day ahead stock price [35]. Krauss *et al.* deployed a statistical arbitrage strategy based on DNN, GBDT and RF to S&P 500 constituents from December 1992 to October 2015, finding that the RF outperform GBDT and DNN in their study [41]. Patel *et al.* compared four prediction models, ANN, SVM, RF and naive-Bayes. Experimental results showed that the RF outperformed the other three prediction models when the evaluation was carried out on 10 years of historical data of two stocks and two stock price indices [42]. This article attempts to construct an adaptive automatic machine learning method selection framework while the prediction effects of different machine learning methods are inconsistent in different scenarios.

The main contributions of our research are as follows: (1) This article combines traditional candlestick charting with the latest machine learning methods to enrich the research content of stock market forecasting. (2) We developed a simple eight-trigram classification following the eight-trigram scheme. We also compare different types of technical indicators in short-term stock forecasting, and further clarified the role of technical indicators in machine learning models. (3) Then an adaptive machine learning method selection framework with a novelty feature engineering scheme was proposed. Appropriate forecasting machine learning methods can be automatically selected in different candlestick charting patterns. (4) We have constructed an investment strategy based on our prediction framework. Empirical results show that this paper's strategy makes good economic returns on both individual stocks and portfolios.

The remainder of this paper is organized as follows. Section 2 outlines the design of an ensemble prediction framework using machine learning techniques. Section 3 presents the empirical results on stock data and robustness checks. Section 4 concludes the paper.

II. METHODOLOGY

This paper proposes an adaptive prediction framework using an ensemble of machine learning models to predict the direction of the closing price, which is shown in FIGURE 1.

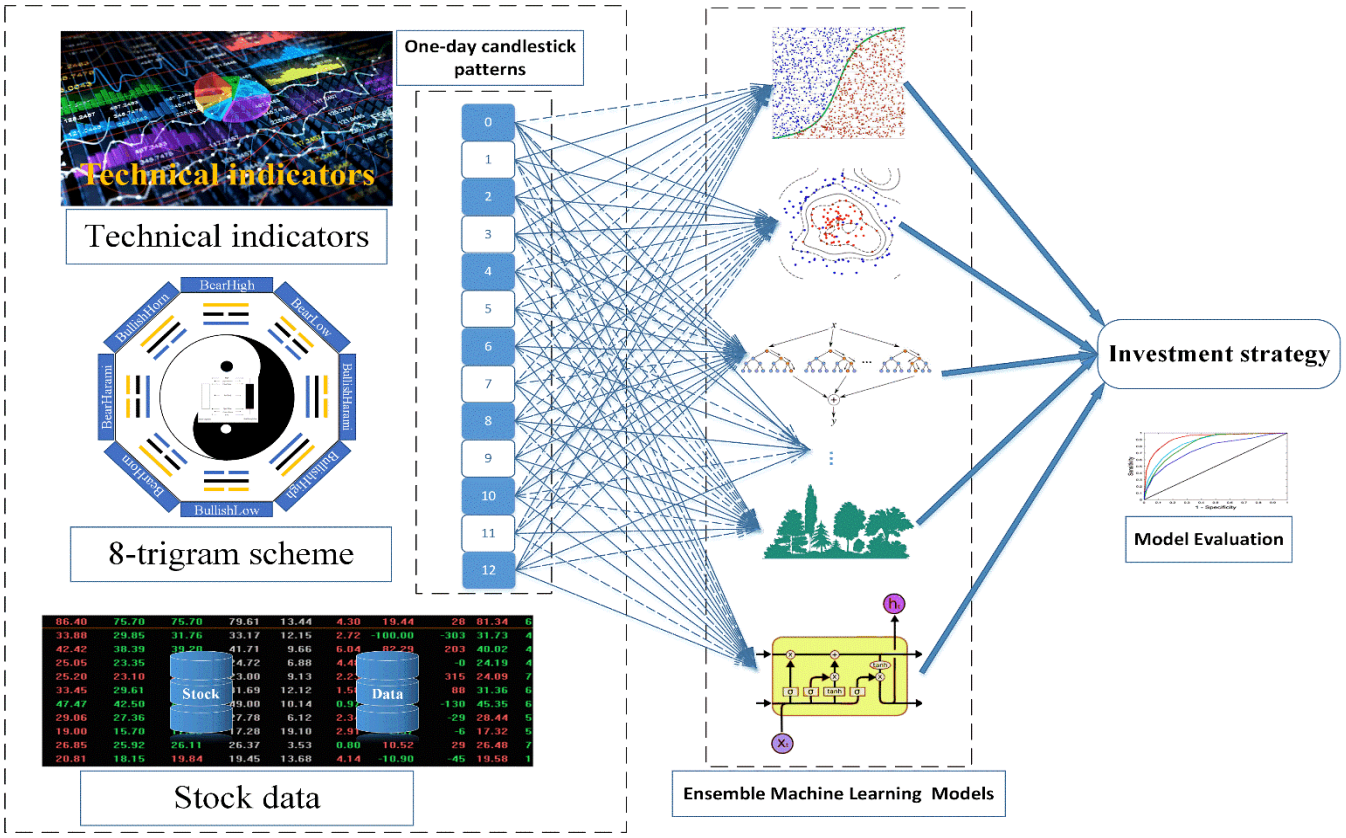


FIGURE 1. Overview of prediction framework.

First, 13 forms of one-day patterns are constructed and classified from 3,455 stocks used in this article, and then the corresponding technical indicators and eight-trigram information are calculated. Then, all feature data are passed as input to the ensemble machine learning model, which tests the prediction accuracy of each pattern. For each pattern, the machine learning method with the highest prediction will be recorded. Finally, the adaptive recommendation schedule gives corresponding stock prediction actions based on the evaluated results.

The main evaluation algorithm is shown as FIGURE 2.

A. FEATURE ENGINEERING USING AN EIGHT-TRIGRAM SCHEME

Key to our prediction framework is a simple eight-trigram scheme to represent inter-day stock price movements based on two-day candlestick patterns. The candlestick chart, also called K-line, is drawn by close, open, high, and low, where the part between the close and open is called real body. If the asset closed higher than it opened, the body is filled with red color in Chinese stock market while represented with white or green color in European and American stock markets. If the close price is lower than open price, the body is painted with green color in Chinese stock market while filled with black or red color in European and American stock markets. And then, the candlestick pattern classification is built, which is shown in FIGURE 3. We divide the candlestick patterns into 13 classes according to the candlestick basic elements: the opening, high, low, and closing prices.

Relative to yesterday's closing price, the opening position of the day reflects the accumulation of sentiments during

Algorithm: Model Evaluation

Input: Patterns data which includes feature engineering data and different indicators

Output: *BestModel*, *F1 score*

```
Evaluation (features):
  foreach p in patterns: Generate p_data of p;
      LogisticRegression (p_data);
      GridSearchCV of KNN (p_data);
      GridSearchCV of SVM (p_data);
      GridSearchCV of RF (p_data);
      GridSearchCV of GBDT (p_data);
      LSTM (p_data);
      BestModel = MAX_F1 (LR, KNN, SVM, RF,
GBDT, LSTM);
      Save the best performance model BestModel,
    and F1 score for pattern p
    Output: List of best performance model, F1 score
    for each pattern
```

FIGURE 2. Main evaluation algorithm.

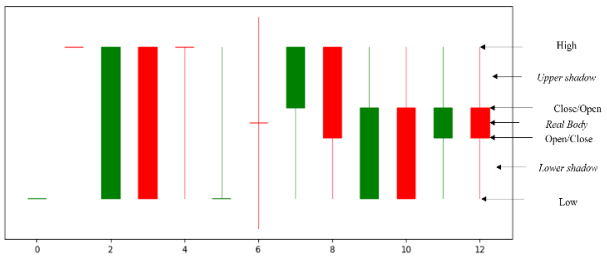


FIGURE 3. Candlestick patterns.

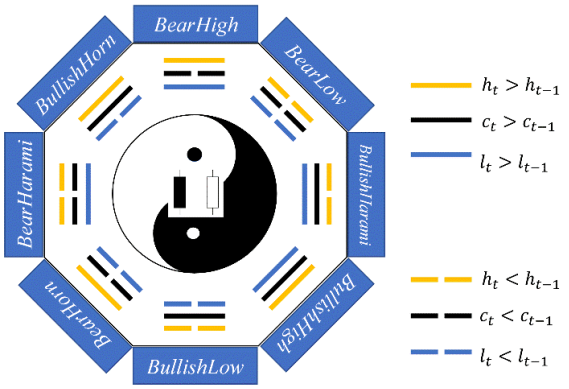


FIGURE 4. Eight-trigrams of inter-day price movement patterns.

non-trading period. We divide the inter-day price movement into eight categories based on the relative position of today's price range and yesterday's K-line patterns. The simple schematic diagram of eight-trigram feature engineering scheme is shown in FIGURE 4. The symbols used in eight-trigrams and the detailed expressions can be found in Appendix I.

On the other hand, trading volume is an important parameter which is not getting enough attention in academia. It is also a variable completely independent of price. The N-day moving average of volume on day t is defined by

$$MA_{n,v}(t) = \frac{1}{n}\sum_{i=0}^{n-1} V_{t-i} \tag{1}$$

The volume rate of change on day t is defined by

$$ROC(n) = \frac{V_t}{MA_{n,v}(t)} \tag{2}$$

Besides, the technical indicators could potentially have an impact on the stock price prediction [43], [44]. Four groups of technical indicators, including 21 indicators, were introduced in our research. As shown in Table 1 Detailed expressions for all indicators can be found in Appendix II.

B. GENERATION OF TRAINING AND TESTING SETS

We begin to extract the features after the data preprocessing process according to the feature engineering. The feature of single k-line patterns classification is generated on the basis price of Opening, High, Low and Closing price. A total of

TABLE 1. Technical indicators.

Group of indicators	Technical indicators
Overlap indicators	Moving Average (MA), Exponential Moving Average (EMA), Double Exponential Moving Average (DEMA), Kaufman's Adaptative Moving Average (KAMA), Simple Moving Average (SMA), Parabolic SAR (SAR)
Momentum indicators	Average Directional Movement Index (ADX), Price Oscillator - Absolute (APO), Balance of Power (BOP), Commodity Channel Index (CCI), Moving Average Convergence/Divergence (MACD), Money Flow Index (MFI), Momentum (MOM), Relative Strength Index (RSI)
Volume indicators	Chaikin A/D Line (AD), Chaikin Oscillator (ADOSC), On Balance Volume (OBV)True Range
Volatility indicators	(TRANGE), Average True Range (ATR), Normalized Average True Range (NATR)

13 K-line patterns was obtained after considering all the circumstances. Secondly, eight inter-day price move indicators including *BullishHorn, BearHorn, BullishHigh, BearHigh, BullishLow, BearLow, BullishHarami* and *BearHarami* are extracted. And then, the rate of volume feature which has not received enough attention is taken into consideration. Instead of simply placing the volume value directly into the forecasting model, we use the variable of the rate of change as a predictive feature. Finally, according to the indicator formula, the values of 21 indicators as prediction parameters are calculated. The research in this paper focuses on short-term forecasting, we choose 5 days or 10 days as the parameters of indicators. Parameters of 5 days for *MA, DEMA, KAMA* and 10 days for *EMA, SMA, ADX, APO, CCI, DX, MFI, RSI, ATR, NATR* are employed.

After all the prediction features are ready, we begin to prepare the training sets and testing sets. We divide the entire data set into two parts, 80% of which are training sets and 20% are testing sets. The corresponding result is the next day's direction of stock price.

C. PREDICTION MODELS

The inference engine is introduced in this phase. Six machine learning models including Logistic Regression (LR), Support Vector Machine (SVM), k-NearestNeighbor (KNN), Random Forest (RF), Gradient Boosting Decision Tree (GBDT) and Long Short-term Memory (LSTM) are used to predict the direction of the closing price. The parameters used in these prediction models are shown as Table 2.

1) LOGISTIC REGRESSION (LR)

Logistic Regression is the most basic machine learning algorithm. The Logistic Regression model returns an equation that determines the relationship between the independent variables and the dependent variable. First, the model calculates linear functions and then converts the result into a probability. Finally, it converts the probability into a label.

TABLE 2. Parameters used in prediction models.

MLs	Parameters
LR	Regularized= *L2*, solver_parameter=*warn, C=1.0, iteration=100, criteria=0.0001*
SVM	C={1e3,5e3,1e4,5e4,1e5}, gamma={0.0001,0.0005,0.001,0.005,0.01,0.1}, optimizer= *GridSearchCV* cv=10
KNN	n_neighbors = range(1,10), weights = ['uniform','distance'], algorithm=['auto','ball_tree','kd_tree','brute'], leaf_size=range(1,2), optimizer= *GridSearchCV* cv=10
RF	n_estimators=range(10,100,5), criterion=[gini, entropy], min_samples_leaf= [2, 4, 6,50], max_depth=range(1,10) , optimizer= *GridSearchCV* cv=10
GBDT	n_estimators=range(10,100), max_features=range(0.6,0.9), max_depth=range(1,10), optimizer= *GridSearchCV* cv=10
LSTM	unit=64, dropout=0.2, activation=sigmoid, loss=binary_crossentropy, optimizer=adam, epochs=20

In the empirical stage, we use *L2* as a regularized parameter, specifying *warn* as the solver parameter which determines our optimization method for the logistic regression loss function. In terms of the termination parameters of the algorithm, the maximum number of iterations which is taken for the solvers to converge to *100* and tolerance for stopping criteria parameter is set to *0.0001*.

2) SUPPORT VECTOR MACHINE (SVM)

In machine learning, support vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data for classification and regression analysis. The SVM training algorithm builds a model when given a set of training examples and assigns the new example to one category or another. The SVM model is to map the example as a point map in space, so that the examples of the separate categories are divided by a clear gap that is as wide as possible. Lee gave a detailed formula for the SVM's two-class problem in his 2009 article [27].

In addition to performing linear classification, SVM can effectively perform nonlinear classification using so-called kernel techniques, implicitly mapping its inputs to high-dimensional feature spaces. There are some studies using the SVM to predict the financial data [45], [46].

In the empirical stage, we get the best performance from a grid search algorithm which sets different penalty coefficients, coefficients of kernel function and degrees. Different parameter combinations produce different clustering effects.

3) K-NEARESTNEIGHBOR (KNN)

K nearest neighbors (KNN) is another machine learning algorithm. The k-NN algorithm looks for 'k' nearest records within the training dataset and uses the majority of the classes of the identified neighbors for classifying. Subha (2012) used k-NN to classify the stock index movement [28]. In the empirical stage, we get the best performance from a grid search algorithm which sets different neighbors, leaves and weights. Different parameter combinations produce different clustering effects.

4) RANDOM FOREST (RF)

Random forests are a combination of tree predictors. Each tree depends on the values of a random vector sampled independently with the same distribution for all trees in the forest. Random forests (RFs) are a nonparametric and nonlinear classification and regression algorithm [47]. Random forests not only use a subset of the training set, but also selects only a subset of the feature set when the tree is established in the decision tree. Booth (2014) used RFs to construct a n automated trading mechanism [29].

In the empirical stage, different parameter combinations may produce different classification effects. We get the best performance from a grid search algorithm which sets different leaves, depth and estimators.

5) GRADIENT BOOSTING DECISION TREE (GBDT)

Gradient Boosting Decision Tree (GBDT) is a popular machine learning algorithm. Friedman (2002) gave a detailed expression of the gradient descent in his research [48]. The core idea of GBDT is that the subsequent model of the sequence no longer directly predicts the predicted value of the data set, but predicts the difference between the predicted value and the true value of the previous model.

In the empirical stage, different parameter combinations may produce different classification effects. We get the best performance from a grid search algorithm which sets different features, depth and estimators.

6) LONG SHORT-TERM MEMORY (LSTM)

Long-short term memory is one of the recurrent neural network (RNNs) architecture [49]. Hochreiter and Schmidhuber proposed a solution by using memory cells [50] which consists of three components, including input gate, output gate and forget gate. The gates control the interactions between neighboring memory cells and the memory cell itself. The input gate controls the input state while the output gate controls the output state which is the input of other memory cells. The forget gate can choose to remember or forget its previous state.

The LSTM network used Keras, an open-source neural-network library written in Python. First, we construct a three-layer LSTM network, including two main processing layers and one output layer (dense layer). In the prediction model, we apply dropout regularization within the recurrent layer. Hereby, a fraction of the input units are randomly dropped at each update during training time to reduce the risk of overfitting and it gets better generalization. And the early stopping mechanism is also used to reduce the risk of overfitting.

In the empirical stage, the hidden neurons are set to 64 and dropout to 0.2 in the first layer and set the hidden neurons to 64 in the second layer. At the output layer, we use the sigmoid activation function to generate the classification results. This configuration yields 17,920 parameters for the first layer,

33,024 parameters for the second layer and 65 parameters for the output layer.

D. MODEL EVALUATION

The inference engine is introduced in this phase. We use six machine learning models to forecast the stock direction of up and down. To evaluate the performance of the prediction models, two commonly used evaluation criteria are used in this study: *Accuracy,* and *F1 score*. *Accuracy* is used to evaluate the overall classification ability of the model. The formula is as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

TP (True Positives) representing model prediction is true, and the real sample is also true; *TN (True Negatives)* representing model prediction is false, and the real sample is also false; *FP (False Positives)* representing model prediction is true while the real sample is false; *FN (False Negatives)* representing model prediction is false while the real sample is true.

Precision is used to estimate the accuracy of positive samples in the prediction data and *recall* is used to evaluate the coverage of positive samples in the prediction data of the model. The formula is shown as (4) and (5).

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

F1 is an indicator used in statistics to measure the accuracy of a binary model, which also considers the accuracy and recalls of the classification model. The formula is as follows:

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{6}$$

And then, the evaluation progress tries to evaluate the results of the machine learning prediction model. In this study, our evaluation model not only considers the *Accuracy* index, but also considers the *Precision* and *Recall* indicators.

E. INVESTMENT STRATEGY

The investment strategy is constructed based on the above evaluation model. This paper considers two situations, including only long and long-short, and builds the corresponding investment strategy. This article assumes that we will invest at the closing price at time t and will be clear at the closing price at time $t + 1$. The flowchart of the investment strategy is shown as FIGURE 5. If the current stock trading market mechanism allows shorting, that means you can go long and short at the same time. The specific construction steps of the investment strategy are as follows:

First, the specific K-line pattern of the current stock is checked at time t. Next, a matching machine learning method is selected to predict the rise or fall of $t + 1$ based on the above evaluation model. If the predicted result is consistent with the real result, the t + 1 profit would be recorded. If the prediction is wrong, the negative profit of $t + 1$ *would be recorded* as loss value. Finally, the above steps will be repeated to calculate $t + 1$, $t + 2$, etc.

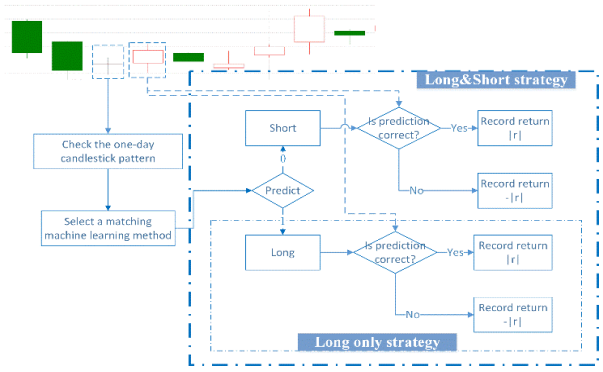


FIGURE 5. Flowchart of the investment strategy.

If the current stock trading market mechanism does not allow shorting, that is to say, you can only go long. The specific construction steps of the investment strategy are as follows:

First, the specific K-line mode of the current stock is checked at time t. Then, a matching machine learning method is selected to predict the rise or fall of $t + 1$ based on the above evaluation model. We only operate the investment when the forecasting result is up. If the predicted result is consistent with the real result, we record the t + 1 profit. Otherwise, we record the negative profit of $t + 1$ as loss value. Finally, the above steps will be repeated to calculate $t + 1$, $t + 2$, etc.

III. EMPIRICAL RESULTS

A. DATA AND TRAINING ENVIRONMENT

As the world's largest developing country and the world's second-largest economy, China's influence on the world is growing. The financial market of China has attracted the attention of domestic and foreign scholars and investors [12], [13], [39]. This paper selects the data of the Chinese stock market as the experimental data. The daily data of the China Stock Market from the 18-year period of 2000 to 2017 is used in this study. All the 3,455 stocks data is collected from CCER, a local data provider of China. First of all, we remove the daily data for a given stock if the trading volume is zero, which is a sign of stopped trading such as due to company reorganization. The distribution of 13 patterns in the data set is shown in Table 3. The distribution of these patterns in historical data is stable. Then, we generate feature information, which contains the date, intra-day pattern, inter-day pattern, 21 other indicator values and the next day's closing price, for each stock on each day t. In order to ensure effectiveness, three rounds of training were carried out. We randomly choose 5,000 rows of daily stock data for each of 13 intra-day patterns from the database, which yields 65,000 rows of data in each round. In order to ensure the balance of classification during training, for each intra-day pattern, we choose half of the training data with

TABLE 3. Distribution of 13 patterns.

Patterns	No.	Ratio	Ratio1	Ratio2
0	6,946	0.10%	0.10%	0.06%
1	15,268	0.21%	0.23%	0.13%
2	56,477	0.77%	0.79%	0.70%
3	57,910	0.79%	0.83%	0.63%
4	12,549	0.17%	0.19%	0.10%
5	7,501	0.10%	0.12%	0.05%
6	170,234	2.33%	2.34%	2.31%
7	630,482	8.63%	8.65%	8.56%
8	247,028	3.38%	3.53%	2.72%
9	186,261	2.55%	2.59%	2.39%
10	563,830	7.72%	7.60%	8.27%
11	2,442,444	33.45%	33.15%	34.77%
12	2,905,003	39.78%	39.89%	39.31%

The statistical data cycle is from Jan 1, 2000 to Dec 31, 2017. The *No.* column refers to the total number of occurrences of the specified pattern. The *Ratio* column indicates the proportion of the pattern in the total data set. *Ratio*1 column indicates the proportion of the pattern in the data set from

rising prices (closing price lower than next day's) and half of the training data with falling prices. Finally, the average is obtained based on three rounds of results.

B. MODEL COMPARISON AND EVALUATION

To clarify the role of technical indicators in machine learning models, the 21 technical indicators are divided into four groups, including Overlap indicators, Momentum indicators, Volume indicators and Volatility indicators. And then each pattern containing feature engineering information is put into the ensemble machine learning prediction framework that contains six machine-learning models for training. 80% of the data as training data and the remaining 20% as test data were set to verify the predictive validity of the model.

First of all, we train the machine learning models without indicators. 13 patterns and 6 machine learning models resulting in a total of 78 predictions shows the effectiveness of feature engineering, which is shown in the Fig. a. of FIGURE 6. The abscissa represents the 13 kinds of one-day candlestick patterns and the ordinate represents the *F1 score*. 49 of the 78 prediction models exceeded the random walk probability. Pattern 0 and pattern 1, which indicate that the maximum price limit has been reached, show strong long-short signals. In this case, pattern 4 and pattern 5 show significant prediction effects, and the *F1 score* value can reach about 0.57. In all prediction models, SVM, RF and GBDT showed good predictions for all patterns, while LR, KNN and LSTM only performed well in individual patterns.

Fig. c. of FIGURE 6 shows the prediction results of the introduction of the Overlap indicators which contain 6 indicators of MA, DEMA, EMA, KAMA, SMA and SAR. 60 of the 78 predictions exceeded the random walk probability. And after the introduction of this indicator group, the predicted maximum *F1 score* of each pattern are all improved slightly. Among them, pattern 1, which indicates a board daily limit, shows a strong long signal, the *F1 score* is more than 0.80. This Overlap indicator group has a significant effect on the

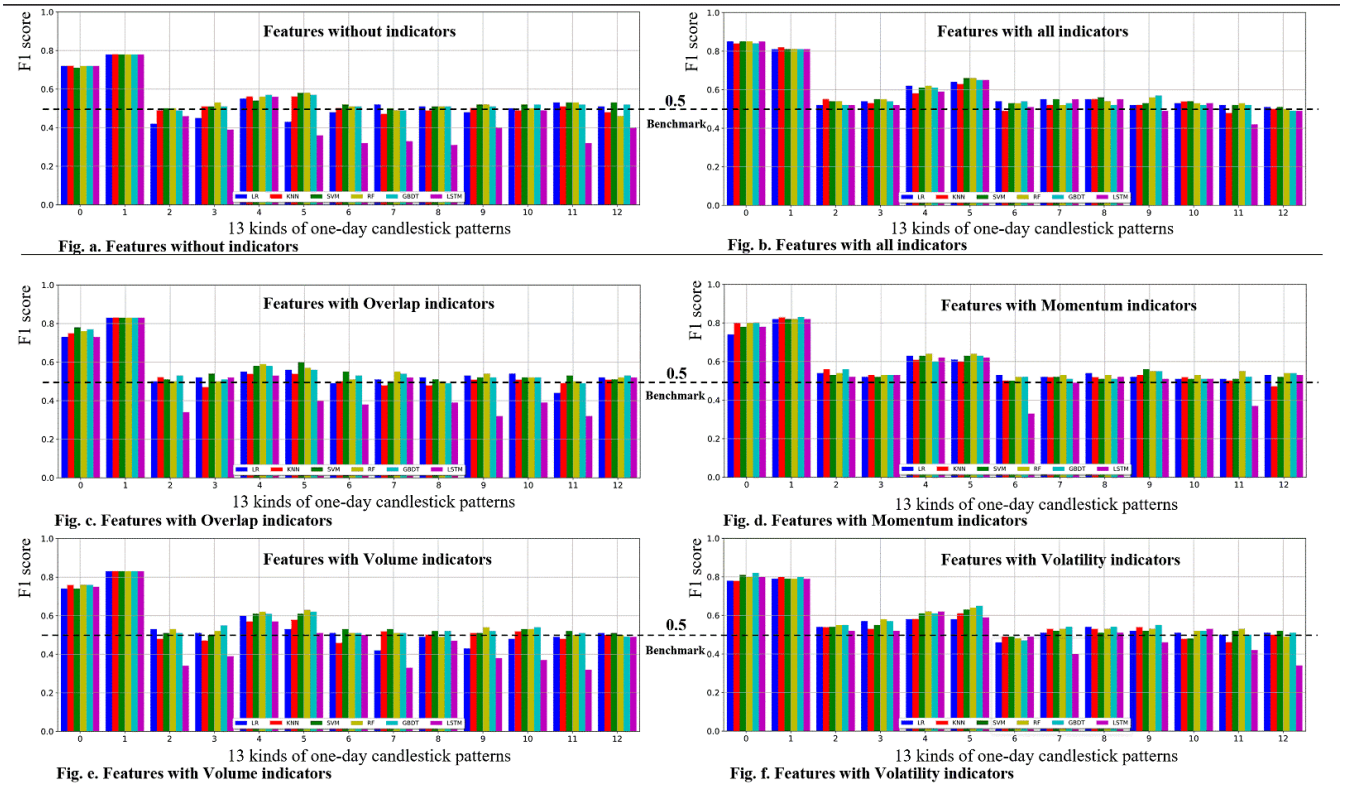


Fig. a. Features without indicators
Fig. b. Features with all indicators
Fig. c. Features with Overlap indicators
Fig. d. Features with Momentum indicators
Fig. e. Features with Volume indicators
Fig. f. Features with Volatility indicators

FIGURE 6. Forecast performance of different combinations of indicator groups.

TABLE 4. Best performance parameters.

Parameters	One-day candlestick patterns												
	0	1	2	3	4	5	6	7	8	9	10	11	12
ML Models	RF	KNN	GBDT	RF	RF	SVM	SVM	GBDT	SVM	GBDT	SVM	RF	RF
Indicators	All	Mom	Mom	Volatility	Mom	All	MA	All	All	All	All	Mom	Mom

improvement of the predictive ability of the SVM model, and the predictive ability of the LR model has also been improved to a certain extent.

Fig. d. of FIGURE 6 shows the forecast results of the introduction of the Momentum indicators which contain 9 indicators of ADX, APO, BOP, CCI, DX, MACD, MFI, MOM and RSI. 73 of the 78 predictions exceeded the probability of random walk. And after the introduction of this indicator, the predicted maximum *F1 score* of each pattern are all improved. Among them, the *F1 score* of pattern 0 and pattern 1 exceeds 0.80. The prediction effect of most prediction patterns has been improved after the introduction of these indicators, reflecting the obvious momentum characteristics in short-term prediction. The prediction ability of RF has been significantly improved, and the prediction effects of pattern 4 and pattern 5 have been significantly improved which reach about 0.62.

Fig. e. of FIGURE 6 shows the predicted results of the introduction of the Volume indicators which contain 3 indicators of AD, ADOSC and OBV. 57 of the 78 predictions exceeded the random walk probability. And after the introduction of this indicator, the predicted maximum *F1 score* of each pattern are slightly improved. Among them, only pattern 4 and pattern 5 show good prediction effects, with a *F1 score about 0.62*. RF and GBDT perform well in all prediction modes. From this we can find that the characteristic project has already included the information of short-term volume change, and the introduction of more volume characteristics cannot significantly improve the forecast level.

Fig. f. of FIGURE 6 shows the results of the introduction of the Volatility indicators which contain 3 indicators of ATR, NATR and TRANGE. 63 of the 78 predictions also exceeded the random walk probability. And after the introduction of the indicator, the highest prediction *F1 score* of each pattern are all improved, and different prediction models have a clear distinction between different pattern prediction effects. After the introduction of this indicator, all the patterns have been improved slightly. The overall prediction results reflect the obvious volatility characteristics in the short term.

In order to test whether more technical indicators can improve the forecast level, we introduce all the technical indicators into our forecasting framework. Fig. b. of FIGURE 6 shows the results of the introduction of all indicators which including 21 indicators. 69 of the 78 predictions exceeded the random walk probability. And after the introduction all indicators, the predicted maximum *F1 score* of each mode are improved. Among them, pattern 0 and pattern 1, that is, a board daily limit shows a strong signal, the *F1 score* is more than 0.80. Pattern 4 and pattern 5 still show good prediction results, *F1 score* is more than 0.62. These indicators have a significant effect on the prediction ability of the LR model, which means that more parameters can improve the linear fitting effect and verify that the stock market has complex nonlinear characteristics.

From the above results, we can see that RF and GBDT have good predictive ability in most cases in short-term prediction. KNN only showed relatively good predictive power in the first six patterns. Although SVM will take too long time to process big data, the prediction level is still significant in some cases. The advantage of the deep learning model LSTM in this scenario is not fully reflected. In addition, we can see that the increase in the number of parameters can increase the level of linear prediction, which also reflects the complexity and diversity of financial markets from another perspective. In most cases, an increase in the number of indicators can increase the level of prediction. However, in the short-term forecast, in the prediction of some models, using a smaller number of momentum indicators and volatility indicators can achieve satisfactory results. The best performance results is shown in Table 4.

C. ROBUSTNESS CHECKS

In order to test the validity of the prediction framework introduced in this paper, we selected the daily data of Shanghai and Shenzhen 300 Index constituent stocks of the China Stock Market from the 18-year period of 2000 to 2017 for testing. At the same time, to ensure the validity of the data, we exclude stocks that are no longer constituents in the last five years. The resulting sample consists of 168 stocks, 553,028 rows of data during that period. We use the best performance parameters of each pattern to verify the predictive validity of the model.

The validity of the prediction results is shown in FIGURE 7. The upper part is the *F1 score* and the lower part is the accuracy rate. From the results, we can see that all the pattern prediction accuracy is greater than 52%, and the *F1 score* is basically more than 50% after using our prediction framework. Among them, pattern 0, pattern 1, pattern 3, pattern 4, pattern 5, and pattern 8 have obvious prediction effects.

D. EMPIRICAL RESULT OF INVESTMENT STRATEGY

Based on the prediction framework of this paper, we construct a trading strategy as follows:

1. Identify the k-line pattern at day t.

2. Use the prediction framework to predict rise or fall of stock closing price at day t + 1.

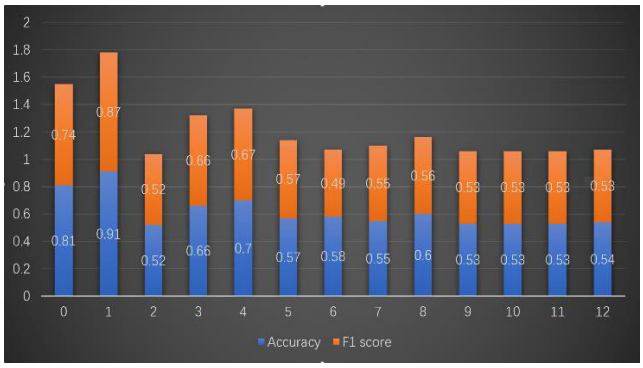


FIGURE 7. Robustness test.

3. If the prediction is correct, as validated by the empirical data, record a profit in the amount of price change (assuming short transactions are allowed and therefore profit can be made even when stock falls).

4. If the prediction is incorrect, record a loss in the amount of price change.

We construct a long-only strategy as a limited version to above, to record a profit only when the stock rises and the prediction correctly predicts the rise. These strategies can be executed at an X%-confidence, that is, be executed only when the confidence of prediction exceeds X%.

FIGURE 8 shows the forecast results of random stock '000001.SZ'. From the figure, we can see that before the 2008 financial crisis, a trading strategy based on the forecasting framework performs better than holding the stock itself. After the financial crisis, the prediction-based strategy has a smaller retraction and will soon outperform the market. However, the prediction-based strategy showed greater volatility during the 2015 stock market crash. The predicted maximum drawdown is 71.4%, which is less than 77.5% of the original stock. And the predicted Sharpe Ratio is 0.31, which is bigger than 0.25 of the original stock. The predicted Sortino Ratio is 0.0348, which is bigger than 0.033 of the original stock. Explain that there are fewer risks and greater benefits based on our forecasting framework. However, after considering the transaction costs, the profits are significantly reduced. FIGURE 8 shows the profitability under different transaction costs of 0.1%, 0.2%, and 0.3%.

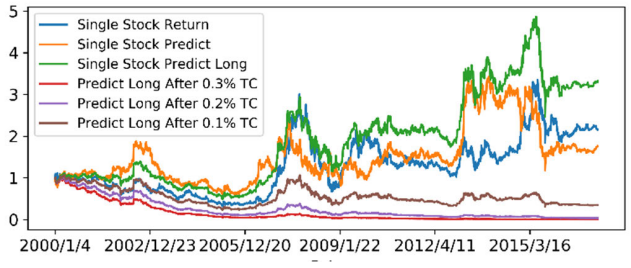


FIGURE 8. Return on single stock using prediction strategies.

Then, we build an equal-weighted portfolio based on the constituents of CSI 300. FIGURE 9 shows the historical

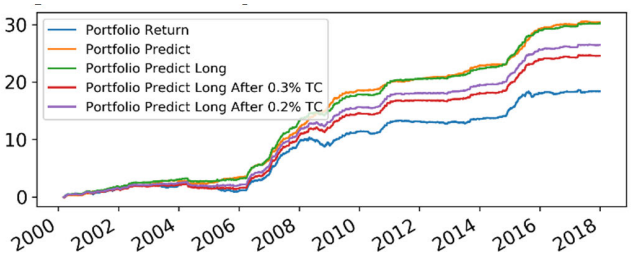


FIGURE 9. Return on CSI 300 using prediction strategies with 55% confidence threshold.

revenue and forecast for the portfolio. From the figure, we can see that the predicted results are significantly better than the portfolio itself. The predicted maximum drawdown is 43.7%, which is less than 63.5% of the original stock. And the predicted Sharpe Ratio is 0.62, which is bigger than 0.35 of the original portfolio, the predicted Sortino Ratio is 0.808, which is bigger than 0.368 of the original portfolio supporting that there are fewer risks and greater benefits based on our forecasting framework. After considering the transaction costs, although the return has declined, it is still possible to obtain excess return, which is higher than the investment portfolio itself. The figure shows the return on the portfolio considering 0.2% and 0.3% transaction cost.

FIGURE 10 shows portfolio return based on the constituents of Shanghai Composite Index. The index component covers 1,380 stocks, and we selected the 2016-2017 data outside the sample for testing to verify the effectiveness of our forecasting framework.

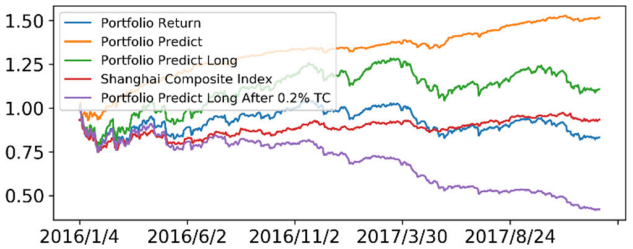


FIGURE 10. Return performance of portfolio from Shanghai composite index.

We can see that the return rate based on our research method is better than the market performance from the figures. The effect will be more prominent if we can short-sell stocks. However, after considering the transaction costs, the profit is significantly reduced. Table 5 shows the finance performance.

TABLE 5. Finance performance.

	Portfolio	Predict1	Predict2	Predict 3	Index
Max drawdown	26.1%	9.7%	24.1%	59.04%	21.0%
Sharpe Ratio	-0.36	1.65	0.24	-1.76	-0.08
Sortino Ratio	-0.023	0.174	0.029	-0.139	-0.009

Predict 1 means predict the portfolio including long and short. Predict 2 means the portfolio only go long. Predict 3 means the portfolio only go long after the 0.2% transaction cost. The index means Shanghai Composite Index.

IV. CONCLUSION

This paper develops an ensemble machine learning prediction model that automatically selects appropriate prediction methods for each daily k-line pattern. The empirical results show that the forecasting framework of this paper has predictive power, and the investment strategy based on the forecasting model can generate superior returns.

This study makes contributions into four aspects. Firstly, this article combines traditional candlestick charting with the latest artificial intelligence methods to enrich the forecasting research of the stock market. By studying the prediction effects of all 13 one-day candlestick patterns under different machine learning methods, we combine traditional technical analysis methods with AI technology. We also concluded that certain candlestick patterns, for example, pattern 4 and pattern 5, have apparent predictive effects in the stock market.

Secondly, in feature engineering, an eight-trigram classification of two-day k-line patterns is developed, in addition to the 13 patterns of daily k-line patterns and volume change features. The simple eight-trigram classification follows the eight-trigram scheme, or Bagua, a key concept in Taoism cosmology. The eight-trigram classification provides a simple set of features based on opening, closing, high and low prices of two consecutive trading days. To improve the forecast level, we introduced four sets of technical indicators: overlap, momentum, volume, and volatility as auxiliary feature variables. We find that the momentum indicators are significantly better via empirical testing than other indicators in short-term forecasting. Additional technical indicators can improve forecasting in most cases. However, in the short-term forecast, in predicting some patterns, a smaller number of momentum or volatility indicators can achieve satisfactory results.

Thirdly, we introduce a framework for assembling multiple machine prediction models to select the optimal prediction method for different feature modes. The ensemble model includes six commonly-used effective prediction models (RF, GBDT, LR, KNN, SVM, LSTM) and optimizes the parameters of each model. In the empirical study, we find that RF and GBDT have a good predictive ability for short-term prediction in most cases. The prediction level of LR needs to be improved by adding features. KNN and SVM only fit in some patterns. The advantage of the deep learning model LSTM in this scenario is not fully reflected.

Finally, based on the prediction results of this paper, we have constructed an investment strategy. The empirical results show that this paper's strategy makes good economic returns on both individual stocks and portfolios theoretically. This also shows that through big data, multiple rounds of training, feature standardization, etc., the prediction of results is effective. The predicted maximum drawdown, Sharpe Ratio, and Sortino Ratio of this investment strategy are better than buying and holding the original stock. However, the transaction costs have a significant impact on actual transactions. In actual investment, other factors need to be considered to obtain excess returns.

The forecasting framework of this paper has predictive power, although it is difficult to profit from certain patterns due to the stop-trading rules of the Chinese market. One of the machine learning methods, the SVM method, is not suitable for predicting large-scale stock data. We intend to incorporate more suitable machine learning methods for prediction, such as reinforcement learning methods, into the ensemble model. Furthermore, we plan to utilize additional predictive factors, such as major news events and market sentiment, to improve forecasting results in the future.

APPENDIX I
DEFINITION OF EIGHT-TRIGRAM

Following table shows the symbols used in eight trigrams:

TABLE 6. Symbols in eight trigrams.

Symbols	Description
h	h_t represents the highest price at time t, h_{t-1} represents the highest price at time t-1
c	c_t represents the closing price at time t, c_{t-1} represents the closing price at time t-1
l	l_t represents the lowest price at time t, l_{t-1} represents the lowest price at time t-1

BullishHorn reflects that the oscillations of the day exceeded the previous cycle and reached a new high and a new low, reflecting the strong characteristics of an oscillation. *The BullishHorn* candlestick at time t should fulfill the following three conditions:

- $h_t > h_{t-1}$
- $l_t < l_{t-1}$
- $c_t > c_{t-1}$

where h_t, l_t, c_t represent highest price, lowest price and closing price at time t.

BearHorn reflects that the oscillations of the day exceeded the previous cycle and reached a new high and a new low, but the closing price was lower than the previous cycle, reflecting the weak characteristics of an oscillation. The *BearHorn* candlestick at time t should fulfill the following three conditions:

- $h_t > h_{t-1}$
- $l_t < l_{t-1}$
- $c_t < c_{t-1}$

BullishHigh shows that the candlestick creates a new high price and the lowest price is higher than the previous period, reflecting a rising strong feature. The *BullishHigh* candlestick at time t should fulfill the following three conditions:

- $h_t > h_{t-1}$
- $l_t > l_{t-1}$
- $c_t > c_{t-1}$

BearHigh reflects a weak rising feature. The candlestick creates a new high price while the closing price is lower than the previous period. *BearHigh* candlestick at time t should fulfill the following three conditions:

- $h_t > h_{t-1}$
- $l_t > l_{t-1}$
- $c_t < c_{t-1}$

BullishLow contains the information that the stock is getting stronger. The stock hit a new low, but the closing price exceeds the previous cycle. *The BullishLow* candlestick at time t should fulfill the following three conditions:

- $h_t < h_{t-1}$
- $l_t < l_{t-1}$
- $c_t > c_{t-1}$

BearLow reflects the weak characteristics. The stock hit a new low and the closing price is lower than the previous perios. The *BearLow* candlestick at time t should fulfill the following three conditions:

- $h_t < h_{t-1}$
- $l_t < l_{t-1}$
- $c_t < c_{t-1}$

BullishHarami shows the process of energy accumulation. The stock's amplitude is within the range of the previous period and the closing price is higher than the previous period. The *BullishHarami* candlestick at time t should fulfill the following three conditions:

- $h_t < h_{t-1}$
- $l_t > l_{t-1}$
- $c_t > c_{t-1}$

BearHarami reflects another process of energy accumulation. The stock's amplitude is within the range of the previous period while the closing price is lower than the previous period. *BearHarami* candlestick at time t should fulfill the following three conditions:

- $h_t < h_{t-1}$
- $l_t > l_{t-1}$
- $c_t < c_{t-1}$

APPENDIX II
DEFINITION OF TECHNICAL INDICATORS

TABLE 7. Classification of technical indicators.

Overlap indicators	MA, DEMA, EMA, KAMA, SMA, SAR
Momentum indicators	ADX, APO, BOP, CCI, DX, MACD, MFI, MOM, RSI
Volume indicators	AD, ADOSC, OBV
Volatility indicators	ATR, NATR, TRANGE

A. OVERLAP INDICATORS

1) MOVING AVERAGE (MA)

$$MA\,(t) = \frac{1}{n}\sum_{i=0}^{n-1} C_{t-i}$$

where n refers to the time interval, and C is the close price.

2) EXPONENTIAL MOVING AVERAGE (EMA)

The Exponential Moving Average is a staple of technical analysis and is used in countless technical indicators.

$$EMA\,(t) = \frac{2}{n+1}C_t + \frac{n-1}{n+1}EMA(t-1)$$

3) DOUBLE EXPONENTIAL MOVING AVERAGE (DEMA)

The DEMA is a smoothing indicator with less lag than a straight exponential moving average.

$$DEMA\,(t) = 2 * EMA\,(C_t) - EMA(EMA\,(C_t))$$

4) KAUFMAN's ADAPTATIVE MOVING AVERAGE (KAMA)

The KAMA automatically increases EMA's smoothing during weak trends and during ranging trends.

$$ER(t) = \frac{Abs(C_t - C_{t-n})}{\sum_{i=t-n}^{t} Abs(C_i - C_{i-1})}$$

$$\mathrm{sc(t)} = (\mathrm{ER(t)} * \left(\frac{2}{n1+1} - \frac{2}{n2+1}\right) + \frac{2}{n2+1})^2$$

$$\mathrm{KAMA\,(t)} = \mathrm{sc(t)} * (C_t - KAMA\,(t-1)) + KAMA(t-1)$$

where $n1$ refers to the fast period, $n2$ refers to the slow period and Abs indicates absolute value.

5) SIMPLE MOVING AVERAGE (SMA)

Moving Averages are used to smooth the data in an array to help eliminate noise and identify trends.

$$SMA\,(t) = \frac{1}{n}(m * MA\,(t) + (n-m) * SMA\,(t-1))$$

where m represents the weight.

6) PARABOLIC SAR (SAR)

The Parabolic SAR calculates a trailing stop.

$$SAR\,(t) = SAR\,(t-1) + af_t * (xp_{t-1} - SAR\,(t-1))$$

where af is acceleration factor and xp is the extreme point.

B. VOLUME INDICATORS

Directional Movement Index (+DI and −DI) The +DI is the percentage of the true range that is up. The −DI is the percentage of the true range that is down.

$$\Delta H = H_{t-1} - H_t$$
$$\Delta L = L_t - L_{t-1}$$

where H refers to the high price and L refers to the low price. The calculation logic of DI is as follows:

If $(\Delta H < 0$ *and* $\Delta L < 0)$ *or* $\Delta H = \Delta L$ *then*

$$plusDM = 0$$
$$minusDM = 0$$

If $\Delta H > \Delta L$ *then*

$$plusDM = \Delta H$$
$$minusDM = 0$$

If $\Delta L > \Delta H$ *then*

$$plusDM = 0$$
$$minusDM = \Delta L$$

Then

$$plusDMsum\,(t) = plusDMsum\,(t-1) - \frac{plusDMsum\,(t-1)}{n} + plusDM$$

$$minusDMsum\,(t) = minusDMsum\,(t-1) - \frac{minusDMsum\,(t-1)}{n} + minusDM$$

$$TR(t) = H_t - L_t$$

$$TRsum\,(t) = TRsum\,(t-1) - \frac{TRsum\,(t-1)}{n} + TR(t)$$

$$+DI(t) = 100 * \frac{plusDMsum(t)}{TRsum(t)}$$

$$-DI\,(t) = 100 * \frac{minusDMsum(t)}{TRsum(t)}$$

Directional Movement Index (DX) The DX is usually smoothed with a moving average.

$$DX(t) = \frac{(+DI(t)) - (-DI(t))}{(+DI(t)) + (-DI(t))}$$

1) AVERAGE DIRECTIONAL MOVEMENT INDEX (ADX)

The ADX is a Welles Wilder style moving average of the Directional Movement Index (DX).

$$ADX(t) = \frac{ADX\,(t-1) * (n-1) + DX(t)}{n}$$

2) PRICE OSCILLATOR-ABSOLUTE (APO)

The Price Oscillator shows the difference between two moving averages.

$$APO\,(t) = MA\,(t1) - MA(t2)$$

where *t1* is the slow-moving average and *t2* is the fast-moving average.

3) BALANCE OF POWER (BOP)

BOP attempts to measure the strength of buyers vs. sellers by assessing the ability of each to push price to an extreme level. BOP calculates raw values for each bar as:

$$BOP\,(t) = \frac{C_t - O_t}{H_t - L_t}$$

where *O* refers to the open price.

4) COMMODITY CHANNEL INDEX (CCI)

The CCI is designed to detect beginning and ending market trends.

$$CCI\,(t) = \frac{\frac{H_t+L_t+C_t}{3} - MA(n)}{0.015 * \frac{1}{n}\sum_{i-n}^{n} MA\,(i) - C_i}$$

5) MOVING AVERAGE CONVERGENCE/ DIVERGENCE (MACD)

The Moving Average Convergence Divergence (MACD) is the difference between two Exponential Moving Averages.

$$shortMA\,(t) = 0.15 * C_t + 0.85 * shortMA\,(t-1)$$

$$longMA\,(t) = 0.075 * C_t + 0.925 * longMA\,(t-1)$$

$$MACD(t) = 0.15{*}C_t + 0.85 * shortMA\,(t-1)$$

6) MONEY FLOW INDEX (MFI)

The Money Flow Index calculates the ratio of money flowing into and out of a security. T

$$typicalPrice(t) = \frac{H_t + L_t + C_t}{3}$$

$$moneyFlow(t) = typicalPrice(t) * V_t$$

The calculation logic of *MFI* is as follows:
If *typicalPrice(t) > typicalPrice(t-1)*

$$positiveMoneyFlow\,(t) = positiveMoneyFlow\,(t-1) + moneyFlow(t)$$

else

$$negativeMoneyFlow\,(t) = negativeMoneyFlow\,(t-1) + moneyFlow(t)$$

then

$$moneyRatio\,(t) = \frac{\sum_{t-n}^{t} positiveMoneyFlow(t)}{\sum_{t-n}^{t} negativeMoneyFlow(t)}$$

$$MFI\,(t) = 100 - \frac{100}{1 + moneyRatio(t)}$$

7) MOMENTUM (MOM)

The Momentum is a measurement of the acceleration and deceleration of prices.

$$\mathrm{MOM}\,(t) = C_t - C_{t-n}$$

8) RELATIVE STRENGTH INDEX (RSI)

The Relative Strength Index (RSI) calculates a ratio of the recent upward price movements to the absolute price movement. The *RSI* is generated as follows:

If $C_t > C_{t-1}$

$$up(t) = C_t - C_{t-1}$$

$$dn(t) = 0$$

else

$$up(t) = 0$$

$$dn(t) = C_{t-1} - C_t$$

then

$$upAvg(t) = \frac{(n-1)\,upAvg(t-1) + up(t)}{n}$$

$$dnAvg(t) = \frac{(n-1)\,dnAvg(t-1) + dn(t)}{n}$$

$$RSI\,(t) = 100 * \frac{upAvg(t)}{upAvg\,(t) + dnAvg(t)}$$

C. VOLUME INDICATORS

1) CHAIKIN A/D LINE (AD)

The *AD* Line is calculated as follows:

$$Clv(t) = \frac{2 * C_t - H_t - C_t}{H_t - L_t}$$
$$AD\,(t) = AD\,(t-1) + V_t * Clv(t)$$

2) CHAIKIN OSCILLATOR (ADOSC)

The Chaikin Oscillatoris essentially a momentum of the Accumulation/Distribution Line.

$$ADOSC\,(t) = EMA\,(AD\,(n1)) - EMA(AD\,(n2))$$

where *EMA* is the exponential moving average, n1 represents the fast period, n2 represents the slow period.

3) ON BALANCE VOLUME (OBV)

The OBV is a cumulative total of the up and down volume. The calculation logic of *OBV* is as follows:

If $C_t > C_{t-1}$

$$OBV\,(t) = OBV\,(t-1) + V_t$$

If $C_t < C_{t-1}$

$$OBV\,(t) = OBV\,(t-1) - V_t$$

Else

$$OBV\,(t) = OBV\,(t-1)$$

where V refers to the volume.

D. VOLATILITY INDICATORS

1) TRUE RANGE (TRANGE)

The True Range is a base calculation that is used to determine the normal trading range of a stock or commodity.

$$TR\,(t) = max\,(H_t, C_{t-1}) - min(L_t, C_{t-1})$$

2) AVERAGE TRUE RANGE (ATR)

The ATR is a moving average of the True Range.

$$ATR\,(t) = \frac{1}{n}\sum_{i=1}^{n} TR(t-i+1)$$

3) NORMALIZED AVERAGE TRUE RANGE (NATR)

The *NATR* is the normalized of *ATR*.

REFERENCES

[1] M. Kumar and M. Thenmozhi, "Forecasting stock index returns using ARIMA-SVM, ARIMA-ANN, and ARIMA-random forest hybrid models," *Int. J. Banking, Account. Financ.*, vol. 5, no. 3, pp. 284–308, 2014, doi: 10.1504/IJBAAF.2014.064307.

[2] P. C. S. Bezerra and P. H. M. Albuquerque, "Volatility forecasting via SVR–GARCH with mixture of Gaussian kernels," *Comput. Manage. Sci.*, vol. 14, no. 2, pp. 179–196, Apr. 2017, doi: 10.1007/s10287-016-0267-0.

[3] I. K. Nti, A. F. Adekoya, and B. A. Weyori, "A systematic review of fundamental and technical analysis of stock market predictions," *Artif. Intell. Rev.*, vol. 53, pp. 1–51, Aug. 2019.

[4] A. Ghaznavi, M. Aliyari, and M. R. Mohammadi, "Predicting stock price changes of tehran artmis company using radial basis function neural networks," *Int. Res. J. App. Basic Sci*, vol. 10, no. 8, p. 972, 2016.

[5] E. Ahmadi, M. Jasemi, L. Monplaisir, M. A. Nabavi, A. Mahmoodi, and P. A. Jam, "New efficient hybrid candlestick technical analysis model for stock market timing on the basis of the support vector machine and heuristic algorithms of imperialist competition and genetic," *Expert Syst. Appl.*, vol. 94, pp. 21–31, Mar. 2018.

[6] C. L. Osler, "Currency orders and exchange rate dynamics: An explanation for the predictive success of technical analysis," *J. Finance*, vol. 58, no. 5, pp. 1791–1819, Oct. 2003, doi: 10.1111/1540-6261.00588.

[7] Y. Zhu and G. Zhou, "Technical analysis: An asset allocation perspective on the use of moving averages," *J. Financial Econ.*, vol. 92, no. 3, pp. 519–544, 2009, doi: 10.1016/j.jfineco.2008.07.002.

[8] H. Bessembinder and K. Chan, "Market efficiency and the returns to technical analysis," *Financial Manage.*, vol. 27, no. 2, p. 5, 1998, doi: 10.2307/3666289.

[9] B. R. Marshall, M. R. Young, and L. C. Rose, "Candlestick technical trading strategies: Can they create value for investors?" *J. Banking Finance*, vol. 30, no. 8, pp. 2303–2323, Aug. 2006, doi: 10.1016/j.jbankfin.2005.08.001.

[10] G. Caginalp and H. Laurent, "The predictive power of price patterns," *Appl. Math. Finance*, vol. 5, nos. 3–4, pp. 181–205, Sep. 1998, doi: 10.1080/135048698334637.

[11] T.-H. Lu, Y.-C. Chen, and Y.-C. Hsu, "Trend definition or holding strategy: What determines the profitability of candlestick charting?" *J. Banking Finance*, vol. 61, pp. 172–183, Dec. 2015, doi: 10.1016/j.jbankfin.2015.09.009.

[12] S. Chen, S. Bao, and Y. Zhou, "The predictive power of Japanese candlestick charting in Chinese stock market," *Phys. A, Stat. Mech. Appl.*, vol. 457, pp. 148–165, Sep. 2016, doi: 10.1016/j.physa.2016.03.081.

[13] M. Zhu, S. Atri, and E. Yegen, "Are candlestick trading strategies effective in certain stocks with distinct features?" *Pacific-Basin Finance J.*, vol. 37, pp. 116–127, Apr. 2016, doi: 10.1016/j.pacfin.2015.10.007.

[14] T.-H. Lu, "The profitability of candlestick charting in the taiwan stock market," *Pacific-Basin Finance J.*, vol. 26, pp. 65–78, Jan. 2014, doi: 10.1016/j.pacfin.2013.10.006.

[15] L. Tao, Y. Hao, H. Yijie, and S. Chunfeng, "K-line patterns' predictive power analysis using the methods of similarity match and clustering," *Math. Probl. Eng.*, vol. 2017, May 2017, Art. no. 3096917, doi: 10.1155/2017/3096917.

[16] A. W. Lo, H. Mamaysky, and J. Wang, "Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation," *J. Finance*, vol. 55, no. 4, pp. 1705–1765, Aug. 2000, doi: 10.1111/0022-1082.00265.

[17] D. Yan, Q. Zhou, J. Wang, and N. Zhang, "Bayesian regularisation neural network based on artificial intelligence optimisation," *Int. J. Prod. Res.*, vol. 55, no. 8, pp. 2266–2287, Apr. 2017, doi: 10.1080/00207543.2016.1237785.

[18] J.-J. Wang, J.-Z. Wang, Z.-G. Zhang, and S.-P. Guo, "Stock index forecasting based on a hybrid model," *Omega*, vol. 40, pp. 758–766, Dec. 2012, doi: 10.1016/j.omega.2011.07.008.

[19] B. M. Henrique, V. A. Sobreiro, and H. Kimura, "Literature review: Machine learning techniques applied to financial market prediction," *Expert Syst. Appl.*, vol. 124, pp. 226–251, Jun. 2019, doi: 10.1016/j.eswa.2019.01.012.

[20] Y. Xiao, J. Xiao, F. Lu, and S. Wang, "Ensemble ANNs-PSO-GA approach for day-ahead stock e-exchange prices forecasting," *Int. J. Comput. Intell. Syst.*, vol. 7, no. 2, pp. 272–290, 2014, doi: 10.1080/18756891.2013.864472.

[21] D. Brownstone, "Using percentage accuracy to measure neural network predictions in stock market movements," *Neurocomputing*, vol. 10, no. 3, pp. 237–250, 1996, doi: 10.1016/0925-2312(95)00052-6.

[22] F. Kamalov, "Forecasting significant stock price changes using neural networks," *Neural Comput. Appl.*, vol. 32, no. 23, pp. 17655–17667, Dec. 2020, doi: 10.1007/s00521-020-04942-3.

[23] P. Yu and X. Yan, "Stock price prediction based on deep neural networks," *Neural Comput. Appl.*, vol. 32, no. 6, pp. 1609–1628, Mar. 2020, doi: 10.1007/s00521-019-04212-x.

[24] M.-C. Wu, S.-Y. Lin, and C.-H. Lin, "An effective application of decision tree to stock trading," *Expert Syst. Appl.*, vol. 31, no. 2, pp. 270–274, Aug. 2006, doi: 10.1016/j.eswa.2005.09.026.

[25] J. Dopke, U. Fritsche, and C. Pierdzioch, "Predicting recessions with boosted regression trees," *Int. J. Forecasting*, vol. 33, no. 4, pp. 745–759, 2017.

[26] S. Barak, A. Arjmand, and S. Ortobelli, "Fusion of multiple diverse predictors in stock market," *Inf. Fusion*, vol. 36, pp. 90–102, Jul. 2017.

[27] M.-C. Lee, "Using support vector machine with a hybrid feature selection method to the stock trend prediction," *Expert Syst. Appl.*, vol. 36, no. 8, pp. 10896–10904, Oct. 2009, doi: 10.1016/j.eswa.2009.02.038.
[28] M. V. Subha and S. T. Nambi, "Classification of stock index movement using k-nearest neighbours (k-NN) algorithm," *WSEAS Trans. Inf. Sci. Appl.*, vol. 9, no. 9, pp. 261–270, 2012.
[29] A. Booth, E. Gerding, and F. McGroarty, "Automated trading with performance weighted random forests and seasonality," *Expert Syst. Appl.*, vol. 41, no. 8, pp. 3651–3661, Jun. 2014, doi: 10.1016/j.eswa.2013.12.009.
[30] C. Lohrmann and P. Luukka, "Classification of intraday S&P500 returns with a random forest," *Int. J. Forecasting*, vol. 35, no. 1, pp. 390–407, Jan. 2019.
[31] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 654–669, 2018, doi: 10.1016/j.ejor.2017.11.054.
[32] A. Mundra, S. Mundra, V. K. Verma, and J. S. Srivastava, "A deep learning based hybrid framework for stock price prediction," *J. Intell. Fuzzy Syst.*, vol. 38, no. 5, pp. 5949–5956, May 2020.
[33] A. Kazem, E. Sharifi, F. K. Hussain, M. Saberi, and O. K. Hussain, "Support vector regression with chaos-based firefly algorithm for stock market price forecasting," *Appl. Soft Comput.*, vol. 13, no. 2, pp. 947–958, 2013.
[34] J. Zhang, L. Li, and W. Chen, "Predicting stock price using two-stage machine learning techniques," *Comput. Econ.*, vol. 57, pp. 1237–1261, 2021, doi: 10.1007/s10614-020-10013-5.
[35] B. Weng, L. Lu, X. Wang, F. M. Megahed, and W. Martinez, "Predicting short-term stock prices using ensemble methods and online data sources," *Expert Syst. Appl.*, vol. 112, pp. 258–273, Dec. 2018.
[36] D. Kumar, S. S. Meghwani, and M. Thakur, "Proximal support vector machine based hybrid prediction models for trend forecasting in financial markets," *J. Comput. Sci.*, vol. 17, pp. 1–13, Nov. 2016.
[37] M. Göçken, M. Özçalıcı, A. Boru, and A. T. Dosdoğru, "Stock price prediction using hybrid soft computing models incorporating parameter tuning and input variable selection," *Neural Comput. Appl.*, vol. 31, no. 2, pp. 577–592, Feb. 2019.
[38] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, "Predicting stock market index using fusion of machine learning techniques," *Expert Syst. Appl.*, vol. 42, no. 4, pp. 2162–2172, Mar. 2015, doi: 10.1016/j.eswa.2014.10.031.
[39] F. Zhou, Q. Zhang, D. Sornette, and L. Jiang, "Cascading logistic regression onto gradient boosted decision trees for forecasting and trading stock indices," *Appl. Soft Comput.*, vol. 84, Nov. 2019, Art. no. 105747, doi: 10.1016/j.asoc.2019.105747.
[40] W. Bao, J. Yue, and Y. Rao, "A deep learning framework for financial time series using stacked autoencoders and long-short term memory," *PLoS ONE*, vol. 12, no. 7, Jul. 2017, Art. no. e0180944, doi: 10.1371/journal.pone.0180944.
[41] C. Krauss, X. A. Do, and N. Huck, "Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500," *Eur. J. Oper. Res.*, vol. 259, no. 2, pp. 689–702, Jun. 2017, doi: 10.1016/j.ejor.2016.10.031.
[42] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, "Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques," *Expert Syst. Appl.*, vol. 42, no. 1, pp. 259–268, 2015.
[43] K.-J. Kim and I. Han, "Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index," *Expert Syst. Appl.*, vol. 19, no. 2, pp. 125–132, Aug. 2000, doi: 10.1016/S0957-4174(00)00027-0.
[44] C.-F. Tsai and Y.-C. Hsiao, "Combining multiple feature selection methods for stock prediction: Union, intersection, and multi-intersection approaches," *Decis. Support Syst.*, vol. 50, no. 1, pp. 258–269, 2010, doi: 10.1016/j.dss.2010.08.028.
[45] H. Li, L.-Y. Hong, Y.-C. Mo, B.-Z. Zhu, and P.-C. Chang, "Restructuring performance prediction with a rebalanced and clustered support vector machine," *J. Forecasting*, vol. 37, no. 4, pp. 437–456, Jul. 2018, doi: 10.1002/for.2512.
[46] L. F. S. Vilela, R. C. Leme, C. A. M. Pinheiro, and O. A. S. Carpinteiro, "Forecasting financial series using clustering methods and support vector regression," *Artif. Intell. Rev.*, vol. 52, no. 2, pp. 743–773, Aug. 2019, doi: 10.1007/s10462-018-9663-x.
[47] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.
[48] J. H. Friedman, "Stochastic gradient boosting," *Comput. Statist. Data Anal.*, vol. 38, no. 4, pp. 367–378, 2002, doi: 10.1016/S0167-9473(01)00065-2.
[49] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2013, pp. 6645–6649, doi: 10.1109/ICASSP.2013.6638947.
[50] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

YAOHU LIN was born in Fujian, China, in April 1984. He received the B.S. and M.S. degrees in computer science and engineering from Beihang University, China, where he is currently pursuing the Ph.D. degree with the School of Economics and Management. His main research interests include machine learning, deep learning, and relation classification.

SHANCUN LIU was born in Hebei, China, in October 1964. He received the B.S. degree in mathematics from Lanzhou University, the M.S. degree in fundamental mathematics from Wuhan University, and the Ph.D. degree from Beihang University. He is currently a Full Professor with Beihang University. He has published more than 100 articles on different journals, such as, *Applied Economics Letters*, *Economics Letters*, *Journal of Systems Science and Complexity*, and *Finance Research Letters*.

HAIJUN YANG (Member, IEEE) was born in Tianjin, China, in January 1970. He received the B.S. and M.S. degrees in mathematics and management science from Nankai University, China, and the Ph.D. degree from Tianjin University.

He is currently a Full Professor with Beihang University. He is also a Fulbright Scholar. He has published more than 30 articles on different journals, such as, *Journal of Evolutionary Economics*, *Entropy*, and *International Journal of Information Technology & Decision Making*.

Dr. Yang is a member of AEA.

HARRIS WU was born in Shenyang, China. He received the Ph.D. degree from University of Michigan.

He is currently a Full Professor with the Department of Information Technology and Decision Sciences, College of Business and Public Administration, Old Dominion University. He has published some articles on *Decision Support Systems*, *Information and Management*, and *Applied Soft Computing Journal*. His research interests include data analytics, social media, cybersecurity, text mining, enterprise information systems, and system integration.

HUMAN COMPUTER INTERACT BASED EYE CONTROLLED MOUSE

Katta Shivani[1], Lekireddy Sreenivas Reddy[2], Buyyakar Karthik[3], Dr I Nagarju[4]

[1,2,3]B.Tech Student, Department Of CSE (Internet of Things), Malla Reddy College of Engineering and Technology, Hyderabad, India.

[4] Professor, Department Of CSE (Internet of Things), Malla Reddy College Of Engineering and Technology, Hyderabad, India.

ABSTRACT:

There are different reasons for which people need an artificial of locomotion such as a virtual keyboard. The number of people, who need to move around with the help of some article means, because of an illness. Moreover, implementing a controlling system in it enables them to move without the help of another person is very helpful. The idea of eye controls of great use to not only the future of natural input but more importantly the handicapped and disabled. Camera is capturing the image of eye movement. First detect pupil center position of eye. Then the different variation on pupil position get different command set for virtual keyboard. The signals pass the motor driver to interface with the virtual keyboard itself. The motor driver will control both speed and direction to enable the virtual keyboard to move forward, left, right and stop.

Key words: eye movement, virtual keypad, artificial of locomotion.

I INTRODUCTION

Nowadays, personal computer systems take a vast part in our day to day survival since they are used in areas such as at workplace etc. These applications have one thing in common i.e. the use of personal computers is mostly dependant on the data input methods such as mouse. But this is not a problem in case of a healthy individual, this may be a problem for people with less freedom of movement of their limbs [1]. In such cases, it might be preferable to use input methods which supports the abilities of the region such as eye movements. To enable such input method as a substitute, a system is designed which follows a low-cost approach to control cursor on a computer system without the use of mouse [6]. In the proposed system, the cursor movement of the computer system is controlled by the

eyeball movement using OpenCV. This system comprises of Raspberry pi [5]. It is interfaced with IP camera which detects the Eyeball movements and based on these eyeball movements the cursor can be controlled accordingly which are processed using the Open CV (Open Computer Vision).Nowadays personal computer systems are carrying a huge part in our everyday lives as they are used in areas such as work, education and enjoyment. What all these applications have in common is that the use of personal computers is mostly based on the input method via keyboard and mouse. While this is not a problem for a healthy individual, this may be an insurmountable bound for people with limited freedom of movement of their limbs. In these cases it would be preferable to use input methods which are based on more abilities of the region such as eye movements. To enable such substitute input methods a system was made which follows a low-price approach to control a mouse cursor on a computer system. The eye tracker is based on images recorded by a mutated webcam to acquire the eye movements. These eye movements are then graphed to a computer screen to position a mouse cursor accordingly. The movement of mouse by automatically adjusting the position of eyesight. Camera is used to capture the image of eye movement. In general, any digital image processing algorithm consists of three stages: input, processor and output. In the input stage image is captured by a camera. It sent to a particular system to focus on a pixel of image that's gives, its output as a processed image. Embedded system is combination of hardware and software.An embedded system can be an independent system or it can be a part of a large system. An embedded system is a microcontroller or microprocessor based system which is designed to perform a specific task. For example, a fire alarm is an embedded system; it will sense only smoke Python is a high-level language. This means that Python code is written in largely recognizable English, providing the Pi with commands in a manner that is quick to learn and easy to follow. This is in marked contrast to low-level languages, like assembler, which are closer to how the computer ―thinks‖ but almost impossible for a human to follow without experience.

II RELATED STUDY

There are two components to the human visual line-of-sight: pose of human head and the orientation of the eye within their sockets. Investigated these two aspects but will concentrate on the eye gaze estimation in this concpt. The present of novel approach called the ―one-circle‖ algorithm for

measuring the eye gaze using a monocular image that zooms in on only one eye of a person. Observing that the iris contour is a circle, Estimate the normal direction of this iris circle, considered as the eye gaze, from its elliptical image. From basic projective geometry, an ellipse can be back-projected into space onto two circles of different orientations. However, by using a geometric constraint, namely, that the distance between the eyeball's center and the two eye corners should be equal to each other, the correct solution can be disambiguated. This allows us to obtain a higher resolution image of the iris with a zoom-in camera, thereby achieving higher accuracies in the estimation. A general approach that combines head pose determination with eye gaze estimation is also proposed. The searching of the eye gaze is guided by the head pose information. The robustness of our gaze determination approach was verified statistically by the extensive experiments on synthetic and real image data. The two key contributions in this concept are that show the possibility of finding the unique eye gaze direction from a single image of one eye and that one can obtain better accuracy as a consequence of this. The first technique is proposed to estimate the 3-D eye gaze directly. In this technique, the cornea of the eyeball is modelled as a convex mirror. Via the properties of convex mirror, a simple method is proposed to estimate the 3-D optic axis of the eye. The visual axis, which is the true 3-D gaze direction of the user, can be determined subsequently after knowing the angle deviation between the visual axis and optic axis by a simple calibration procedure. Therefore, the gaze point on an object in the scene can be obtained by simply intersecting the estimated 3-D gaze direction with the object. In addition, a dynamic computational head compensation model is developed to automatically update the gaze mapping function whenever the head moves. Hence, the eye gaze can be estimated under natural head movement. Furthermore, it minimizes the calibration procedure to only one time for a new individual. The advantage of the proposed techniques over the current state of the art eye gaze trackers is that it can estimate the eye gaze of the user accurately under natural head movement, without need to perform the gaze calibration every time before using it. Our proposed methods will improve the usability of the eye gaze tracking technology, and believe that it represents an important step for the eye tracker to be accepted as a natural computer input device.

III METHODOLOGY

The user has to sits in front of thedisplay screen of private computer or pc, a specialised video camera

established above the screen to study the consumer's eyes. The laptop constantly analysis the video photo of the attention and determines wherein the consumer is calling at the display screen. not anything is attached to the consumer's head or body. To "pick out" any key, the user seems at the key for a exact period of time and to "press" any key, the consumer just blink the eye. On this device, calibration procedure is not required. For this system enter is simplest eye. No outside hardware is connected or required. Camera gets the input from the eye. After receiving these streaming movies from the cameras, it'll spoil into frames. After receiving frames, it will check for lights conditions because cameras require enough lighting fixtures from external sources in any other case blunders message will show at the screen. The captured frames which can be already in RGB mode are transformed into Black 'n' White. Five. Pics (frames) from the enter supply focusing the eye are analysed for Iris detection (middle of eye).

IV RESULTS EXPLANATION

The current android application is developed using Xml, Java, SQL with Firebase connectivity. It can be used by every individual who are in a need of fulfilling their household services.

At the time of submission of my application was capable of doing the following:

- Displaying thehome screen with different fragments.
- Authentication of user by using login screen using Firebase.
- Home screen to display based on user or service provider.
- After successful login of user, they can choose the service and book a slot of their particular service provider from the displayed list.
- Add, update, view, delete the user details.
- After successful login of service provider, they can view all the bookings that are booked by the users and can attend them one by one.
- Service provider can also set his preferences to not available, if he's too busy or many users had already booked him.
- Service provider has the ability to change their particular radius of location for servicing.
- He can set up to 10 km radius.

- Logout and end the session.

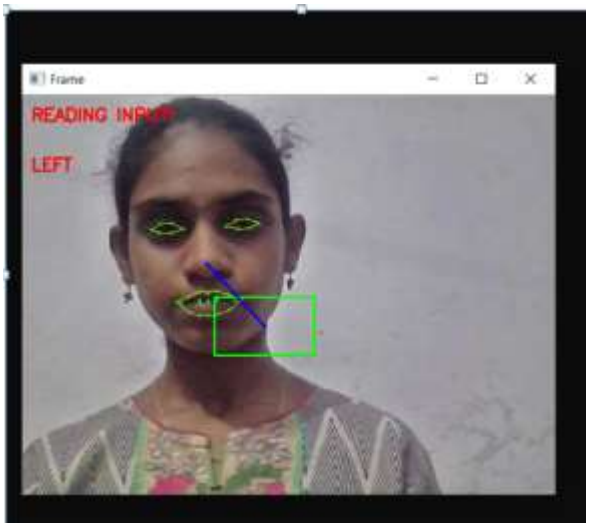


Fig.4.1. OUTPUT results.

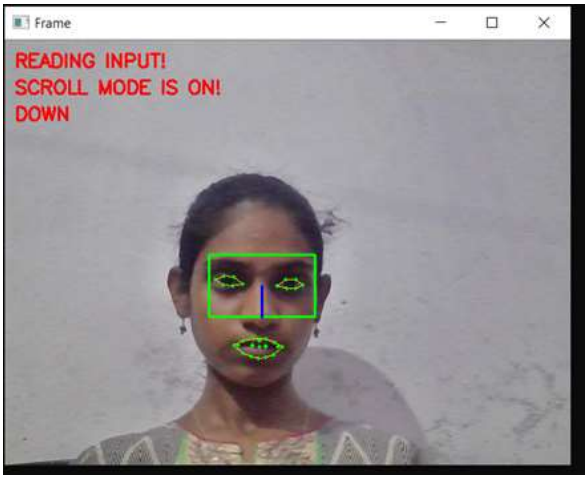


Fig.4.2. Detection of Eyeball.

V CONCLUSION

First detect pupil center position of eye. Then the different variation on pupil position get different command set for virtual keyboard. The signals pass the motor driver to interface with the virtual keyboard itself. The motor driver will control both speed and direction to enable the virtual keyboard to move forward, left, right and stop.

VI REFERENCES

1. Brooks,R.E.(1997) ―Towards a theory of the cognitive processes in computer programming,‖ Int. J. Man-Mach. Studies, vol. 9, pp. 737–751.

2. Cheng- Chih Wu, Ting-Yun Hou(2015)‖Tracking Students' Cognitive Processes During Program Debugging‖—An Eye-Movement Approach, IEEE.

3. Ehrlich,K. and Soloway,E.(1983) ―Cognitive strategies and looping constructs: An empirical study,‖ Commun. ACM, vol. 26, no. 11, pp. 853–860.

4. Eric Sung and Jian-Gang Wang (2002)―Study on Eye Gaze Estimation‖, IEEE, VOL. 32, NO. 3, JUNE .

5. Murphy,L. (2008)―Debugging: The good, the bad, and the quirky—A qualitative analysis of novices' strategies,‖ SIGCSE Bull., vol. 40, no. 1, pp. 163–167.

6. QiangJi and Zhiwei Zhu (2007)‖Novel Eye Gaze Tracking Techniques Under Natural Head Movement‖, Senior Member, IEEE, VOL. 54, NO. 12, DECEMBER .

7. Rajlich,V. and Xu,S.(2004) ―Cognitive process during program debugging,‖ in Proc. 3rd IEEE ICCI, pp. 176– 182.

8. Renumol,V.(2009) ―Classification of cognitive difficulties of students to learn computer programming,‖ Indian Inst. Technol. India.

9. Seung-Jin Baek and Young-Hyun Kim(2013)‖Eyeball Model-based Iris Center Localization for Visible Imagebased Eye-Gaze Tracking Systems‖,IEEE.

10. Tianchi Liu, Yan Yang.Driver (2015)‖Distraction Detection Using Semi-Supervised Machine Learning‖, IEEE.