ENHANCED STOCK PRICE PREDICTION USING CRNN, SENTIMENT ANALYSIS, IABC OPTIMIZATION, AND ADVANCED TEXTUAL AND TECHNICAL FEATURE EXTRACTION

[1]R. Rathiga* and [2]Dr. T. Rathimala

[1]Research Scholar, Department of Computer and Information Science, Faculty of Science, Annamalai University, Annamalainagar, Tamilnadu, India.
Email: rajrathics@gmail.com
[2]Assistant professor/Programmer, Department of Computer and Information Science, Faculty of Science, Annamalai University, Annamalainagar, Tamilnadu, India.
Email: rathimalat@gmail.com
Corresponding author email: rajrathics@gmail.com

KEYWORDS

Stock Price Prediction; Sentiment Analysis; CRNN; IABC; ITF-IDF.

ABSTRACT

Stock price prediction is a very crucial yet discouraging task in financial analytics because of the complexity and nonlinearity of the market. The paper proposes a hybrid prediction model that combines Convolutional Recurrent Neural Networks (CRNN) with the Improved Artificial Bee Colony (IABC) optimization for improved accuracy. This model combines textual data from major stock forums and technical market indicators from the Shanghai Stock Exchange for effective stock price prediction. Preprocessing of the textual data for quality assurance: text cleaning, tokenization, stop word removal, stemming, and lemmatization. Feature extracting techniques include Bag-of-Words (BoW), N-grams, and Improved Term Frequency-Inverse Document Frequency (ITF-IDF) to represent in numerical form. As mentioned, a CRNN has performed sentiment analysis that takes convolutional layers for spatial feature extractions and recurrent layers in temporal dependency modeling. These classified sentiments are combined with technical indicators to form a unified feature set for stock price prediction. The IABC algorithm further optimizes the model by hyperparameter tuning and improving convergence. Experimental verification time intervals prove the superiority of the model over traditional methods. This hybrid approach has significantly improved the accuracy, scalability, and practical applicability for financial decision-making.

1. Introduction

Stock is a class of financial instrument that offers both great risks and great rewards. This makes the combination quite unique, thus making stocks an attractive choice for many investors in their investment portfolios. When investors correctly predict the direction of stock prices, they stand to reap considerable returns on their investments [1,2]. The trend or pattern is not, however, the sole indicator of the movement of stocks; it depends on the performance of a wide number of factors. These involve the state of the economy generally, the current market state, events that may occur related to society and economy in general, investor preferences and habits, and management decisions on companies [3]. All these factors are intertwined, making correct stock price predictions an ongoing challenge and thus the focus of concentrated research [4]. Traditionally, in stock price forecasting, statistical and econometric models have been used widely. Although these methods could provide some degree of insight, often they cannot satisfy the dynamics and multi-dimensionality of the stock market [5]. The development of computer technology has, since the 1970s, changed all this, and the application

of machine learning techniques in making stock price predictions has become possible for researchers [6,7]. These techniques have been helpful to investors because they provide tools that could be used to outline ways through which risks are minimized and returns maximized.
The stock market is characterized by behaviors of complex time series and normal dynamic traits that change extremely fast during trading hours [8]. Once the market opens, stock trading tends to increase and, consequently, stock prices face frequent changes. This high volatility is compounded by lots of factors that can unfold along an unexpected path, causing a stock price trajectory normally to be nonstationary [9]. Therefore, the prediction of stock prices is still one of the most challenging tasks in predictive research. From many sides, academics have attempted to predict stock prices during the last decades and paid considerable attention to the improvement in models and feature selection of these models. However, most of these econometric methods are deficient in considering many of the external factors that affect fluctuation in stock prices [10]. They also possess strong underpinning assumptions about the data which may not necessarily be the case in the real world. Because of these deficiencies, the application of machine learning methods has emerged recently as better alternatives for effective generation of stock price predictions. Several studies have evidenced that the deep learning model outperforms traditional methods, and more precisely, neural networks have shown to be particularly effective compared to regression and discriminant models [11]. The researchers, in the hunt for better predictions, have also explored the relationships between new features and stock prices. This includes considerations of factors such as political climate, macroeconomic indicators, and overall investor sentiment that have been integrated into predictive models to enhance their accuracy [12].
While existing research has made considerable strides in stock price prediction, two key limitations remain in current methodologies [13]. While most of these models adopt the features of text from social media to capture important information, traditional text mining techniques-like the bag-of-words model-sometimes fail to realize the rich semantic information in the texts of social media. Consequently, such lapses may dampen the quality of the prediction models. The dimensionality of the features has to be reduced during the balancing process with respect to the text features with financial metrics in stock price prediction [14]. However, most of the previously related methods adopt PCA and LDA for this purpose. Unfortunately, PCA is vulnerable to information loss and is unsuitable for nonlinear data. Likewise, LDA fails effectively exploring semantic nuances from social media expressions. Therefore, it turns out that these models will not be very relevant or effective in stock price predictions. Various factors prevail in stock market price determination, which are influenced by investor reactions to financial news and other developments [15]. Understanding these elements is essential for developing more accurate prediction models.
The contributions of this paper are given below,

- This work combines textual sentiment analysis from stock forums with technical market indicators sourced from the Shanghai Stock Exchange. Such a method links qualitative market sentiments to quantitative data for better predictions of stock prices.
- In this work, a new hybrid framework combining CRNN for spatial and temporal feature extraction and Improved Artificial Bee Colony optimization for hyperparameter tuning is proposed. This synergy will improve the prediction accuracy and convergence of the model.
- Advanced data preprocessing methods such as text cleaning, tokenization, stemming, and lemmatization techniques have been employed in this work, along with powerful feature extraction methods like Bag-of-Words, N-grams, and ITF-IDF, to ensure quality inputs, hence better predictive performance.

This paper is further divided into the following sections. Part 2 presents both related works and problem statement. The suggested method is implemented and illustrated in part 3. The result and discussion are then presented in the part 4, followed by the conclusion in the part 5.

2. Related Works

In 2022, Li et al. [16] proposed the PCC-BLS framework, a multi-indicator feature selection method combining the Pearson Correlation Coefficient (PCC) and Broad Learning System (BLS). PCC selects key features from 35 inputs, including stock prices, technical, and financial indicators, which are then processed using BLS for feature extraction and rapid training.

In 2020, Vijh et al. [17] used ANN and RF techniques to forecast the next day's closing price for five companies from different sectors. The models take financial data, including Open, High, Low, and Close prices, and generate new input variables.

In 2024, Bandhu et al. [18] used LSTM networks for forecasting Apple Inc.'s stock prices using historical data from Tiingo. It involves multiple stages, including data cleaning, feature selection, feature scaling, model building, evaluation, and improvement. Data preprocessing prepares the raw data, while feature engineering extracts relevant features to enhance accuracy.

In 2022, Liu and Ma [19] proposed a method combining the Elman neural network and quantum mechanics to improve market forecasting. The network's sensitivity to dynamic information is enhanced by incorporating an internal self-connection signal, crucial for system modeling. The Double Chains Quantum Genetic Algorithm is used to optimize learning rates. The model is validated by forecasting closing prices of six stock markets, with simulation results demonstrating its feasibility and effectiveness.

In 2023, Md et al. [20] introduced a novel approach for stock price prediction using a Multi-Layer Sequential Long Short-Term Memory (MLS LSTM) model with the Adam optimizer. By leveraging normalized time series data and time steps, the model captures dependencies between past and future stock prices, overcoming the vanishing gradient issue typical in recurrent neural networks, and providing more accurate predictions based on past trends.

In 2023, Yun et al. [21] addressed limitations in feature-importance methods by considering time-dependency and collective behavior of features. We propose a two-stage feature selection process combining internal technical indicators and external market prices. Using Savitzky-Golay smoothing for piecewise curve fitting, our method improves forecasting accuracy and provides flexible, interpretable insights by selecting the most relevant features for different data segments.

In 2020, Xiao et al. [22] proposed a combined model to improve stock price prediction accuracy. It integrates a cumulative auto-regressive moving average (ARI-MA) with a least squares support vector machine (LS-SVM). The results demonstrate that the ARI-MA-LS-SVM model outperforms traditional single models, offering a more reliable approach for forecasting stock market trends and aiding investment decision-making.

In 2023, Zhao and Yang [23] developed a novel hybrid model, SA-DLSTM, for stock market prediction and simulation trading. The model combines an emotion-enhanced convolutional neural network (ECNN), denoising autoencoder (DAE), and LSTM. ECNN extracts sentiment from user-generated comments, while DAE improves prediction accuracy by extracting key features from stock data. LSTM then uses these features and sentiment indexes to make more reliable stock market predictions.

In 2022, Srijiranon et al. [24] introduced a hybrid model, PCA-EMD-LSTM, combining Principal Component Analysis (PCA), Empirical Mode Decomposition (EMD), and Long Short-Term Memory (LSTM) to predict one-step-ahead closing prices in the Thai stock market. Additionally, news sentiment analysis using FinBERT is incorporated to enhance the model’s performance. This approach leverages financial and economic news to improve the original LSTM model's predictive accuracy.

In 2021, Ho and Huang [25] developed a multichannel collaborative network that integrates candlestick-chart and social-media data for stock trend predictions. Social media sentiment features are extracted using sentiment analysis on Twitter, while stock data is transformed into candlestick charts. The network has two branches: a one-dimensional CNN for sentiment classification and a two-dimensional CNN for image classification of candlestick charts.

2.1. Problem Statement

Stock price prediction remains one of the most challenging tasks in the financial sector due to the complexity and volatility of financial markets. The price of stocks is influenced by a multitude of factors, including market trends, economic indicators, company performance, and investor sentiment. Traditional stock price prediction models primarily rely on historical price data, which often fail to capture the intricate, non-linear relationships among these factors. A key aspect that has gained attention in recent years is sentiment analysis, which analyzes public opinion, typically through social media or news, to gauge the emotions or sentiments that could impact market behavior. Incorporating sentiment analysis into stock price prediction models provides valuable insights into how public perception can influence stock movements. However, combining sentiment data with traditional time-series data, such as financial reports, poses challenges in terms of data integration, feature extraction, and model performance, requiring the development of more robust hybrid models.

3. Proposed Methodology

Stock price prediction involves forecasting future prices based on historical data, market trends, and external factors like economic indicators. Key challenges include market volatility, where unpredictable factors cause price fluctuations; non-linear relationships, as stock prices are influenced by complex patterns that are difficult to model; poor data quality, which can introduce noise and inaccuracies; and overfitting, where models perform well on historical data but fail to generalize to unseen data. To overcome these challenges, this work proposed a hybrid deep learning model combining CRNN. Fig. 1 depicts the overall proposed architecture.

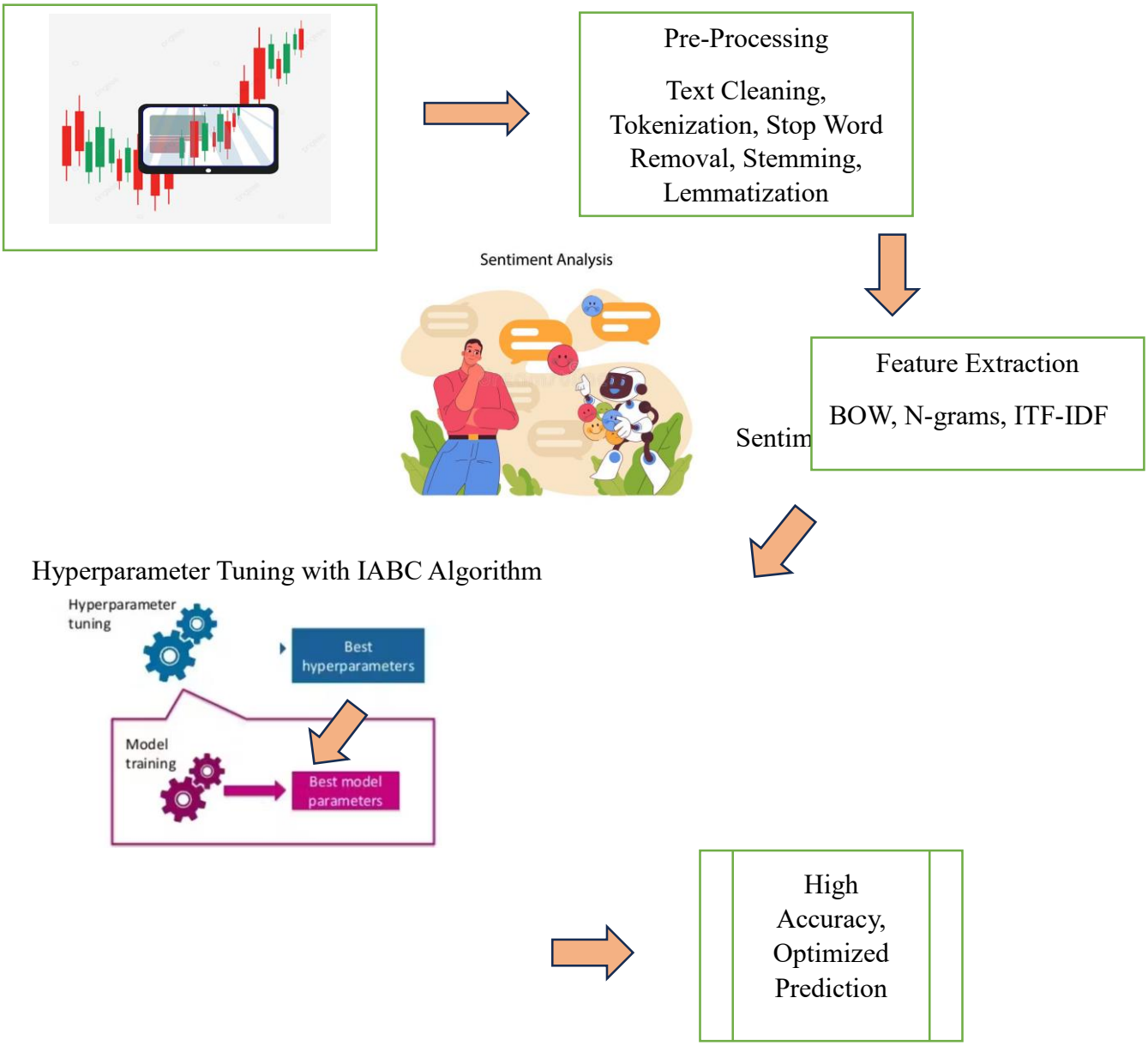


Figure 1: Overall Proposed Architecture

3.1. Data Preprocessing

This research applies data preprocessing techniques including text cleaning, tokenization, stop words removal, and stemming/lemmatization to prepare data for analysis, ensuring accuracy and relevance.

3.1.1. Text Cleaning

Text cleaning is the process of refining raw text data to enhance its quality and usability for analysis. It involves removing unwanted characters, such as punctuation and special symbols, correcting spelling errors, and standardizing text formats. This process also includes handling whitespace, removing extra spaces, and normalizing text to a consistent case. Additionally, text cleaning may involve filtering out irrelevant information, such as HTML tags or metadata, and dealing with encoding issues. By performing text cleaning, researchers can ensure that the data is standardized, consistent, and free from noise, making it suitable for further analysis and interpretation.

3.1.2. Tokenization

Tokenization is a fundamental concept in NLP and computer science. It involves breaking down a piece of text into smaller components, or tokens. These tokens can be words, phrases, symbols, or even individual characters, depending on the specific requirements of the task. In NLP, tokenization is often the first step in processing raw text data. By breaking down the text into tokens, it becomes easier to analyze and manipulate the data programmatically. For example, tokenization allows for tasks like counting word frequencies, identifying grammatical structures, or detecting patterns in text. There are several common tokenization techniques. Whitespace tokenization splits text based on spaces or other whitespace characters, effectively separating words. Word tokenization further refines this process by considering punctuation marks, hyphens, and other delimiters when breaking the text into words. Text can be divided into individual characters via a process called character tokenization, which is helpful for tasks like sentiment analysis and text production. Different languages, writing styles, and text formats may require different tokenization strategies to achieve optimal results. Overall, tokenization serves as a foundational step in text processing tasks, enabling computers to understand and work with human language more effectively.

3.1.3. Stop Word Removal

Stop word removal is a vital preprocessing step in NLP aimed at filtering out common words that lack substantive meaning and are thus deemed irrelevant to the analysis. These words, known as stop words, typically include articles, conjunctions, prepositions, and pronouns. By eliminating stop words from text data, NLP models can focus more on the essential content words, which carry the main message of the text. Removing stop words helps in reducing noise and improving the efficiency of subsequent NLP tasks such as text classification, sentiment analysis, and information retrieval. The process of stop word removal involves comparing each word in the text against a predefined list of stop words specific to the language or domain. Words found in this list are then removed from the text, leaving behind only the significant terms. However, it's crucial to note that the list of stop words may vary depending on the application. Some stop words might be retained if they hold specific significance.

3.1.4. Stemming

Stemming is a linguistic process used in NLP to reduce words to their base or root form, known as the stem. It involves removing affixes such as prefixes, infixes, postfixes, and suffixes from words, thereby transforming different forms or derivatives of a word into a single unified form. For example, words like running, runs, and ran would all be stemmed to the root run. Stemming aims to simplify and standardize word variants to improve text analysis and retrieval processes. By reducing words to their base forms, stemming helps in normalizing vocabulary and reducing data redundancy. However, stemming algorithms do not always produce actual dictionary words, as they prioritize linguistic reduction over linguistic accuracy. Common stemming algorithms include the Porter stemming algorithm, Snowball stemming algorithm, and Lancaster stemming algorithm, each employing heuristic rules to determine the stem of a word. While stemming is a useful preprocessing technique in NLP for tasks such as information retrieval and text mining, it may occasionally lead to inaccuracies or loss due to the aggressive nature of the stemming process.

3.1.5. Lemmatization

Lemmatization is a linguistic process used in natural language processing to reduce words to their base or root form, known as the lemma. Unlike stemming, which simply chops off prefixes or suffixes to derive the root word, lemmatization applies morphological analysis to determine the lemma, ensuring it is a valid word in the language. This technique helps in standardizing word forms, reducing the complexity of text data, and improving the accuracy of tasks like text analysis, information retrieval, and machine learning, by treating different inflected forms of a word as a single entity.

3.2. Feature Extraction

In this study, feature extraction methods are utilized to transform textual data into numerical representations. These methods include BOW, which represents text as a collection of unique words and their frequencies; n-grams, which capture sequences of adjacent words; and ITF-IDF, which weighs the importance of words based on their frequency in a document relative to the entire dataset.

3.2.1. Bag-of-Words

BoW model is a foundational technique in NLP used for text representation. In BoW, a text, such as a sentence or a document, is converted into a vector of numbers. This vector representation is created by counting the frequency of each word in the text while disregarding grammar, word order, and even context. The model gets its name from the concept of treating the text as a "bag" of words, where the importance lies in the presence and frequency of words rather than their sequence. To build a BoW model, the following steps are typically followed:

- Text Preprocessing: This involves converting the text to lowercase to ensure uniformity and removing punctuation, special characters, and stopwords (common words like 'is', 'the', 'and' that do not carry significant meaning).
- Tokenization: The text is split into individual words or tokens.
- Vocabulary Creation: A list of all unique words (vocabulary) in the dataset is compiled.
- Vector Creation: Each document is represented as a vector where each position corresponds to a word from the vocabulary. The value at each position indicates the frequency of the word in the document.

Consider two sentences: "Welcome to Great Learning, Now start learning" and "Learning is a good practice". After preprocessing (lowercasing, removing stopwords and punctuation), the sentences become "welcome great learning now start learning" and "learning good practice". The combined vocabulary might be: ["welcome", "great", "learning", "now", "start", "good", "practice"]. The BoW vectors for these sentences would then be created by counting the occurrences of each word in the vocabulary for each sentence. Thus, the vectors would be:

- Sentence 1: [1, 1, 2, 1, 1, 0, 0]
- Sentence 2: [0, 0, 1, 0, 0, 1, 1]

The BoW model is advantageous due to its simplicity and efficiency. However, it has limitations, such as ignoring word order and context, which can be significant in understanding the meaning of the text. Additionally, it treats all words equally, which may not be ideal since some words are more informative than others.

3.2.2. N-grams

An n-gram is a contiguous sequence of 'n' items from a given sample of text or speech, where 'n' denotes the number of items in the sequence. These items can be words, characters, or syllables, though in text processing, n-grams typically refer to sequences of words. There are various types of n-grams based on the value of 'n': unigrams (N=1) consist of single words, bigrams (N=2) are pairs of consecutive words, trigrams (N=3) are triplets of consecutive words, and four-grams (N=4) are sequences of four consecutive words, and so on. Consider the sentence "The cow jumps over the moon." Unigrams would be "The", "cow", "jumps", "over", "the", "moon"; bigrams would be "The cow", "cow jumps", "jumps over", "over the", "the moon"; and trigrams would be "The cow jumps", "cow jumps over", "jumps over the", "over the moon".

To calculate the number of n-grams in a sentence with 'X' words, the formula used is $X-N+1$. For instance, in a 6-word sentence with bigrams (N=2), the number of bigrams would be $6-2+1=5$. N-grams have several practical applications in natural language processing and text mining. They are crucial in language modelling, where they predict the next word in a sequence based on the previous 'n-1' words, aiding in tasks like speech recognition and text generation. They play a role in spelling correction by suggesting corrections through comparing n-grams

of misspelled words to those in a dictionary of correctly spelled words. In text summarization, n-grams help identify key phrases and summarize content, and in machine translation, they assist in predicting the most likely translation of a phrase by considering the context of surrounding words. The function splits a given text into tokens and then collects n-grams by joining 'n' consecutive tokens, thus providing a list of n-grams for further processing.

3.2.2. ITF-IDF

TF-IDF is a statistical measure used in natural language processing and information retrieval to evaluate the importance of a word in a document relative to a collection of documents (corpus). It aims to reflect how relevant a word is to a document within a collection of documents. Here's a breakdown of the components of TF-IDF:

Term Frequency (TF): Term Frequency measures how often a term (word) appears in a document. It's calculated by dividing the number of occurrences of a term in a document by the total number of terms in the document. Mathematically, it's represented as per Eq. (1).

$$TF(t,d) = \frac{Number\ of\ times\ term\ t\ appears\ in\ document\ d}{Total\ number\ of\ terms\ in\ document\ d} \quad (1)$$

Inverse Document Frequency (IDF): IDF measures the significance of a term across a collection of documents. It diminishes the weight of terms that appear frequently across many documents and emphasizes terms that are rare in the corpus. IDF is calculated by taking the logarithm of the ratio of the total number of documents to the number of documents containing the term t. Mathematically, it's represented as per Eq. (2).

$$IDF(t,D) = log\left(\frac{Total\ number\ of\ documents}{Number\ of\ documents\ containing\ term\ t}\right) \quad (2)$$

TF-IDF Score: The TF-IDF score of a term in a document combines both TF and IDF to determine the weight of the term in the document. It's calculated by multiplying the TF of the term in the document by its IDF across the corpus. Mathematically, it's represented as per Eq. (3).

$$TF - IDF(t,d,D) = TF(t,d) \times IDF(t,D) \quad (3)$$

The traditional TF-IDF algorithm assesses the importance of feature words by considering their frequency within documents but overlooks how these frequencies vary across different categories, which can hinder classification accuracy. To address this, an enhanced IDF method is proposed in this study. This method uses existing discipline classifications as a reference to differentiate feature words that appear with varying frequencies and meanings across different disciplines. By referencing subject classifications, the improved algorithm aims to discern the meanings of feature words in different subjects, thus enhancing the confidence in the TF-IDF algorithm's results. The revised formula for IDF is given as per Eq. (4).

$$IDF = -lg\left(1 - \frac{f(n_i)}{f(n_i)+f(m_i)}\right) = lg\left(1 + \frac{f(n_i)}{f(m_i)}\right) \quad (4)$$

Here $f(n_i)$ represents the likelihood of feature word i being in class m, $f(m_i)$ denotes the likelihood of feature word i being in other categories. These other categories refer to the classifications obtained after excluding class m from the document. Although two feature words may appear the same number of times in the document, $f(n_i)$ and $f(m_i)$ will differ due to the varying distributions of these words across different categories, leading to different IDF values. Consequently, this ITF-IDF algorithm can more effectively distinguish between the distributions of distinct feature words.

3.3. Sentiment Analysis Using CRNN

In this work, sentiment analysis using a CRNN is combined with technical indicators to create a unified feature set. By merging the context from text data with numerical technical indicators, the approach enhances predictive accuracy, leveraging the CRNN's ability to capture temporal dependencies and patterns for comprehensive decision-making insights.

3.3.1. CRNN

CNNs are a subset of deep learning models that are skilled in identifying and deriving meaningful patterns from unprocessed input. Convolutional, pooling, activation, completely connected, and an output layer are among them. While pooling layers down sample features, convolutional layers use filters to extract features. Fully connected layers produce predictions, and activation functions incorporate non-linearities. CNNs are trained to minimize a selected loss function by optimizing their parameters using methods like gradient descent and backpropagation. CNNs are able to adjust and enhance their recognition and classification skills of objects in images through this iterative process. Three different sorts of layers commonly make up an ANN's architecture: input, hidden, and output layers. There are neurons, or nodes, in every layer, and information is carried by the connections between these neurons. A neural network built for sequential data is called an RNN. RNNs, in contrast to feedforward neural networks, have connections that form directed cycles, which allows them to handle sequential input and store hidden state information. Because the order of the input pieces matters, they are especially useful for jobs involving time series data or natural language processing. Recurrent units, such Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), are used by RNNs to update hidden states over time and manage long-term dependencies. They use methods such as Backpropagation through Time (BPTT) to update parameters according to the data's sequential structure. Furthermore, RNNs are used in security frameworks to continually monitor incoming data and identify anomalies or departures from predicted patterns that could point to system faults or security risks. The hidden state hs_t at time t in an RNN is computed using Eq. (5).

$$hs_t = \sigma(w_{hsi} i_t + w_{hshs} hs_{t-1} + bi_{hs}) \quad (5)$$

i_t represents the input at time, w_{hsi} is the weight matrix for the input, w_{hshs} is the weight matrix for the hidden state, bi_{hs} is the bias term, and σ denotes the activation function, typically a non-linear function such as the sigmoid or hyperbolic tangent ($tanh$). The output $o_t$at time t is computed based on the hidden state and expressed as per Eq. (6).

$$o_t = softmax(w_{ohs} hs_t + bi_o) \quad (6)$$

Where, w_{ohs} is the weight matrix connecting the hidden state to the output, bi_o is the bias term, $softmax$ is the activation function, commonly used for multi-class classification problems. The RNN processes sequences by iterating through time steps, updating the hidden state at each step based on the current input and the previous hidden state.

3.4. Optimization with IABC

The paper applies an optimization technique known as IABC, which increases the quality and convergence speed of the solution. IABC optimizes parameters effectively, ensuring that the entire system performs well through improved exploratory and exploitation phases.

3.4.1. IABC

ABC is a population-based metaheuristic optimization algorithm that has been inspired by the foraging behavior of honey bees. The main components of the ABC algorithm include employed, onlooker, and scout bees. Employed bees search for food sources and share information with onlooker bees. Onlooker bees probabilistically choose sources based on quality. Meanwhile, some employed bees turn into scouts to explore new regions when abandoning their current sources, adding exploration for preventing entrapment in local optima. The ABC algorithm effectively emulates social cooperation and intelligent foraging in refining the solutions iteratively to obtain the optimal or near-optimal solutions within the search space. Therefore, the algorithm is widely applicable in various optimization problems.

In the ABC algorithm, the swarm consists of employed and onlooker bees, each comprising half of the total swarm size, which equals the number of solutions. It starts with a randomly distributed population of ss solutions (food sources), where ss is the swarm size.

Each solution, $A_i = \{a_{i,1}, a_{i,2}, \ldots, a_{i,ds}\}$, with ds as the dimension size, generates candidate solutions C_i in its neighborhood to optimize as per the proposed Eq. (7).

$$c_{i,j} = a_{i,j} + \emptyset_{i,j} \cdot w \cdot (a_{i,j} - a_{k,j}) \quad (7)$$

Where w represents the weight factor to control the impact of the current solution's quality, defined as per Eq. (8).

$$w = w_{max} - \frac{(w_{max} - w_{min}) \cdot t}{t_{max}} \quad (8)$$

Linearly Decreasing Weight (LDW) in the IABC algorithm gradually shifts the search process from exploration to exploitation as iterations progress. Initially, higher weight promotes broad exploration to find diverse solutions, while a decreasing weight prioritizes fine-tuning in later stages for optimal convergence. This dynamic adjustment enhances solution accuracy and prevents premature convergence, improving the algorithm's overall performance in complex optimization problems. The algorithm randomly selects a candidate solution A_k (where, $i \neq k$), and a dimension index j from $\{1, 2, \ldots, ds\}$, then determines a random number $\emptyset_{i,j}$ within [-1, 1]. Using these, a new candidate solution C_i is generated. If C_i fitness surpasses A_i, $A_i$is updated; otherwise, it remains unchanged. Employed bees share their food source info through dances with onlooker bees. Onlookers evaluate this information and choose a food source probabilistically based on nectar amount, akin to roulette wheel selection as Eq. (9) suggests.

$$r_i = \frac{fit_i}{\sum_{j=1}^{ss} fit_j} \oplus Levy(\beta) \quad (9)$$

The incorporation of Lévy flight in the ABC algorithm enhances resource search efficiency through the allowance of a random walk during the employed bee phase. This new mechanism generates new solutions around the best food source, improving the direction of the search and algorithm performance. The probability of choosing the ith food source is based on its fitness value, which is referred to as fit_i. If the fitness value is higher, then it will have a higher chance of choosing the ith food source. If a position cannot be improved within a predetermined number of cycles, then that food source is abandoned. In such cases, represented by A_i, the scout bee replaces it with a new food source according to Eq. 10.

$$A_{i,j} = lob_j + ra(0,1) \cdot (upb_j - lob_j) \quad (10)$$

Within the range [0, 1], a normal distribution is used to create the random number $ra(0,1)$. The lower and upper bounds of the jth dimension is denoted by the symbols lob and upb, respectively. Algorithm 1 demonstrates the step-by-step process of implementing the proposed methodology, highlighting key operations, optimizations, and workflows.

Algorithm 1: Enhanced Stock Price Prediction
Input: Historical stock prices, financial news/articles, social media sentiment data Output: Predicted stock price trends 1. Data Collection - Collect historical stock price data 2. Preprocessing - Clean historical price data (remove missing or noisy values) - Tokenize and clean textual data by removing stop words, punctuation, and irrelevant information 3. Advanced Feature Extraction - Extract technical indicators - Use advanced NLP techniques (TF-IDF, word embeddings) to encode textual data - Perform sentiment scoring on textual data using pre-trained sentiment models 4. Sentiment Integration - Integrate sentiment scores with extracted features (combine sentiment data with technical indicators and historical prices)

5. Optimization with IABC
   - Initialize IABC parameters (population, food sources, iterations)
   - Use IABC to optimize hyperparameters of CRNN model
   - Evaluate fitness of each food source based on prediction error
   - Update best solution using a linearly decreasing weight strategy
   - Return optimized hyperparameters
6. CRNN Model Development
   - Design a CRNN architecture combining convolutional layers for feature extraction and recurrent layers for sequence learning
   - Train CRNN using optimized hyperparameters and the combined feature set
7. Model Training
   - Split data into training, validation, and testing sets
   - Train CRNN with the training dataset
8. Prediction
   - Use the trained CRNN model to predict future stock prices
   - Evaluate the model on the test set and compare predicted vs. actual prices
9. Output
10. End

4. Result and Discussion

In the stock price prediction using the proposed hybrid model, the performance metrics provide a comprehensive evaluation of the model's effectiveness. Observed key performance metrics that evaluate the effectiveness of the proposed model. The accuracy, precision, recall, and F1-score are critical indicators that assess how well the model performs in terms of prediction.

Accuracy of 91% reflects the overall percentage of correct predictions made by the model. It indicates that the model is highly reliable in correctly identifying both positive and negative instances across all classes. The accuracy is an essential metric for determining the general effectiveness of the model, though it does not differentiate between types of errors. Precision at 89% signifies the proportion of true positive predictions among all instances predicted as positive. This indicates that when the model classifies a sample as positive, it is correct 89% of the time. A high precision suggests that the model is good at avoiding false positives, making it useful in applications where the cost of a false positive is significant.

Recall of 92% represents the proportion of actual positives correctly identified by the model. This indicates that the model is successful in identifying most of the true positive instances, which is crucial for applications, where failing to detect a positive instance (false negatives) is costly or undesirable. A high recall suggests the model is sensitive to positive cases, minimizing the risk of missing them. The F1-score of 90% is the harmonic mean of precision and recall, offering a balanced view of the model's performance. A high F1-score means that the model performs well in both precision and recall, making it suitable for scenarios where both false positives and false negatives must be minimized.

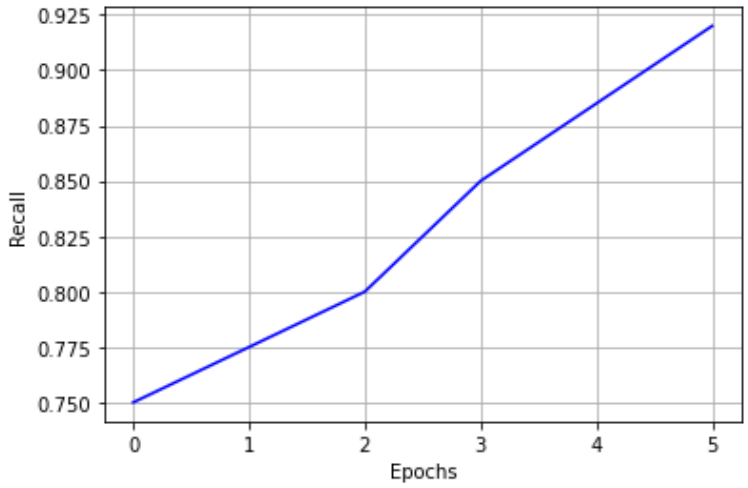
4.1. Dataset Collection

The 2022 Shanghai Stock Exchange 50 (SSE 50) dataset [26] contains trading data for the 50 largest stocks in China, selected based on market capitalization. It covers various financial indicators for these stocks during the year 2022. The dataset includes information such as the stock's exchange date, stock code (instrument), rate of return on equity (ROE), free cash flow, earnings per share (EPS), quarterly net profit growth, price-to-earnings ratio (PE_TTM), market capitalization, and average household shareholding ratio. Additional columns track short-term stock performance, including returns over the last five trading days and transaction amounts. The dataset is valuable for financial analysis, investing, and data visualization, providing insights into the performance of major Chinese stocks during 2022.

4.3. Graphical Representation of Existing and Proposed Model

Figure 2 illustrates how the performance metrics change with varying epochs during the training process. The increasing trend of accuracy, precision, recall, and F1-score across epochs indicates that the model is effectively learning from the data and improving its predictions over time. Figure 3, the Accuracy vs. Loss graph, reveals the relationship between model accuracy and the loss function. As the model progresses through epochs, the loss decreases, while accuracy improves, which is a typical indication of successful model convergence. This trend suggests that the model is gradually minimizing errors and aligning more closely with the true stock price predictions.

(a)



(b)

(c)

(d)

Figure 2: Performance Metrics Varying Epochs

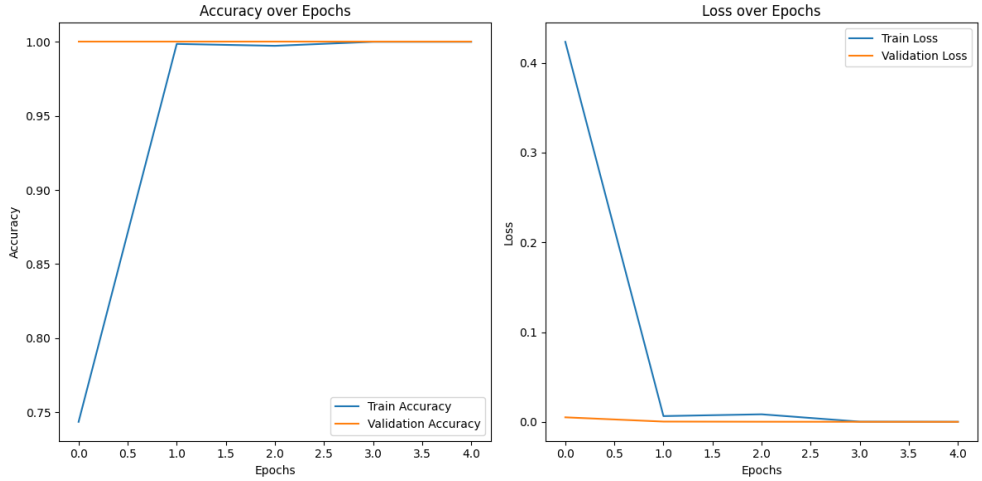


Figure 3: Accuracy vs Loss Graph

The Confusion Matrix in Figure 4 presents a detailed evaluation of the model's classification performance. The confusion matrix shows how many instances were correctly classified as true positives (correctly predicted price increases) and true negatives (correctly predicted price decreases), as well as the false positives and false negatives. The high values of accuracy,

precision, recall, and F1-score suggest that the model is highly proficient in distinguishing between the predicted price movements.

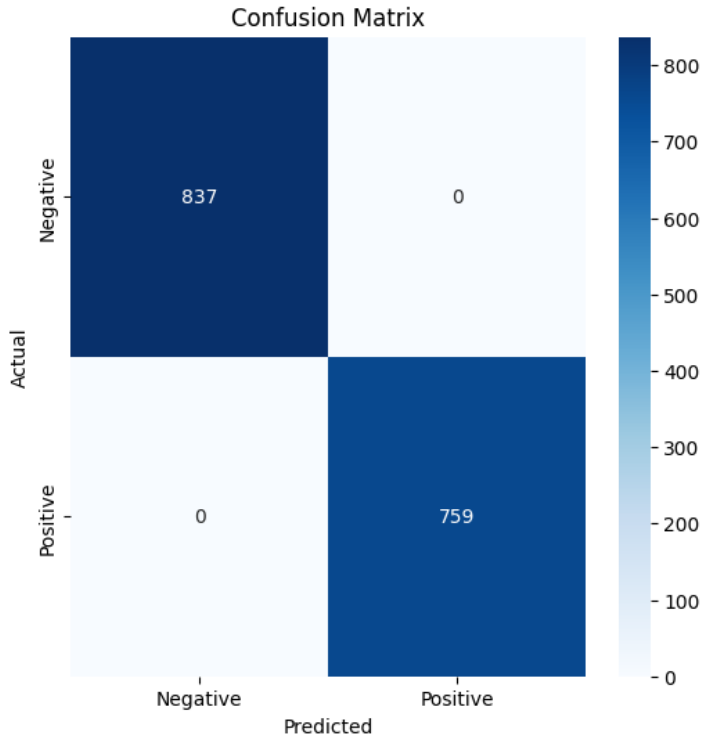


Figure 4: Confusion Matrix

These results suggest that the hybrid CRNN-IABC model, with its integration of sentiment analysis and technical indicators, provides accurate and reliable stock price predictions, outperforming traditional methods. The optimization through IABC further refines the model, enhancing its predictive power and general applicability in real-world financial decision-making.

5. Conclusion

This study developed a hybrid prediction model that fused CRNN with an IABC method. To effectively anticipate stock prices, the program used technical market indicators from the Shanghai Stock Exchange and textual data from prominent stock forums. Several quality assurance procedures, including text cleaning, tokenization, stop word removal, stemming, and lemmatization, were used in the preparation of textual data. The data was transformed into numerical form using techniques such as Bag-of-Words, N-grams, and ITF-IDF for feature extraction. A single feature set was created by combining technical indications with the categorized attitudes. By adjusting hyperparameters and improving convergence, the optimization method significantly enhanced the model's performance. The model's advantage over conventional techniques was demonstrated through experimental verification throughout a range of time periods. The model achieved an impressive accuracy of 91%, demonstrating its effectiveness in stock price prediction.

References

[1] Jin, Z., Yang, Y. and Liu, Y., 2020. Stock closing price prediction based on sentiment analysis and LSTM. Neural Computing and Applications, 32, pp.9713-9729.

[2] Shen, J. and Shafiq, M.O., 2020. Short-term stock market price trend prediction using a comprehensive deep learning system. Journal of big Data, 7, pp.1-33.

[3] Rezaei, H., Faaljou, H. and Mansourfar, G., 2021. Stock price prediction using deep learning and frequency decomposition. Expert Systems with Applications, 169, p.114332.

[4] Yu, Z., Qin, L., Chen, Y. and Parmar, M.D., 2020. Stock price forecasting based on LLE-BP neural network model. Physica A: Statistical Mechanics and Its Applications, 553, p.124197.
[5] Lu, W., Li, J., Wang, J. and Qin, L., 2021. A CNN-BiLSTM-AM method for stock price prediction. Neural Computing and Applications, 33(10), pp.4741-4753.
[6] Ji, X., Wang, J. and Yan, Z., 2021. A stock price prediction method based on deep learning technology. International Journal of Crowd Science, 5(1), pp.55-72.
[7] Wu, S., Liu, Y., Zou, Z. and Weng, T.H., 2022. S_I_LSTM: stock price prediction based on multiple data sources and sentiment analysis. Connection Science, 34(1), pp.44-62.
[8] Gülmez, B., 2023. Stock price prediction with optimized deep LSTM network with artificial rabbits optimization algorithm. Expert Systems with Applications, 227, p.120346.
[9] Kanwal, A., Lau, M.F., Ng, S.P., Sim, K.Y. and Chandrasekaran, S., 2022. BiCuDNNLSTM-1dCNN—A hybrid deep learning-based predictive model for stock price prediction. Expert Systems with Applications, 202, p.117123.
[10] Jing, N., Wu, Z. and Wang, H., 2021. A hybrid model integrating deep learning with investor sentiment analysis for stock price prediction. Expert Systems with Applications, 178, p.115019.
[11] Wu, J.M.T., Li, Z., Herencsar, N., Vo, B. and Lin, J.C.W., 2023. A graph-based CNN-LSTM stock price prediction algorithm with leading indicators. Multimedia Systems, 29(3), pp.1751-1770.
[12] Chen, Y., Wu, J. and Wu, Z., 2022. China's commercial bank stock price prediction using a novel K-means-LSTM hybrid approach. Expert Systems with Applications, 202, p.117370.
[13] Chen, W., Zhang, H., Mehlawat, M.K. and Jia, L., 2021. Mean–variance portfolio optimization using machine learning-based stock price prediction. Applied Soft Computing, 100, p.106943.
[14] Liu, T., Ma, X., Li, S., Li, X. and Zhang, C., 2022. A stock price prediction method based on meta-learning and variational mode decomposition. Knowledge-Based Systems, 252, p.109324.
[15] Yu, P. and Yan, X., 2020. Stock price prediction based on deep neural networks. Neural Computing and Applications, 32(6), pp.1609-1628.
[16] Li, G., Zhang, A., Zhang, Q., Wu, D. and Zhan, C., 2022. Pearson correlation coefficient-based performance enhancement of broad learning system for stock price prediction. IEEE Transactions on Circuits and Systems II: Express Briefs, 69(5), pp.2413-2417.
[17] Vijh, M., Chandola, D., Tikkiwal, V.A. and Kumar, A., 2020. Stock closing price prediction using machine learning techniques. Procedia computer science, 167, pp.599-606.
[18] Bandhu, K.C., Litoriya, R., Jain, A., Shukla, A.V. and Vaidya, S., 2024. An improved technique for stock price prediction on real-time exploiting stream processing and deep learning. Multimedia Tools and Applications, 83(19), pp.57269-57289.
[19] Liu, G. and Ma, W., 2022. A quantum artificial neural network for stock closing price prediction. Information Sciences, 598, pp.75-85.
[20] Md, A.Q., Kapoor, S., AV, C.J., Sivaraman, A.K., Tee, K.F., Sabireen, H. and Janakiraman, N., 2023. Novel optimization approach for stock price forecasting using multi-layered sequential LSTM. Applied Soft Computing, 134, p.109830.
[21] Yun, K.K., Yoon, S.W. and Won, D., 2023. Interpretable stock price forecasting model using genetic algorithm-machine learning regressions and best feature subset selection. Expert Systems with Applications, 213, p.118803.

[22] Xiao, C., Xia, W. and Jiang, J., 2020. Stock price forecast based on combined model of ARI-MA-LS-SVM. Neural Computing and Applications, 32(10), pp.5379-5388.
[23] Zhao, Y. and Yang, G., 2023. Deep Learning-based Integrated Framework for stock price movement prediction. Applied Soft Computing, 133, p.109921.
[24] Srijiranon, K., Lertratanakham, Y. and Tanantong, T., 2022. A hybrid framework using PCA, EMD and LSTM methods for stock market price prediction with sentiment analysis. Applied Sciences, 12(21), p.10823.
[25] Ho, T.T. and Huang, Y., 2021. Stock price movement prediction using sentiment analysis and CandleStick chart representation. Sensors, 21(23), p.7957.
[26] Dataset collected from: “https://www.kaggle.com/datasets/liqiang2022/2022-sse-50-dataset”, dated 1/12/2024.