Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

A Hybrid RNN-LSTM Model Optimized with EGWCS for Sentiment Analysis and Stock Price Prediction Using Shap

R. Rathiga[1], Dr. T. Rathimala[2]

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

Predicting stock prices based on sentiment analysis is a challenging task due to the complex and volatile nature of financial data. Traditional methods struggle with the high-dimensional, noisy, and dynamic textual data derived from social media platforms and financial news. This study addresses these challenges by proposing a hybrid model integrating Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks, optimized using the Enhanced Grey Wolf Sand Cat Swarm (EGWSCS) algorithm, which combines Grey Wolf Optimization (GWO) and Sand Cat Swarm Optimization (SCSO). Sentiment features are extracted using STOCKBERT, a domain-specific language model, to capture nuanced financial sentiment. The textual data undergo preprocessing steps including text cleaning, tokenization, and lemmatization. Improved Principal Component Analysis (I-PCA) reduces feature dimensionality for efficient processing. EGWSCS fine-tunes hyperparameters, improving prediction accuracy and avoiding local optima. Simulation results demonstrate the proposed model's superior performance in sentiment analysis-driven stock price prediction, offering a robust framework for decision-making in financial markets.
Keywords: Sentiment Analysis; Stock Price Prediction; EGWSCS; LSTM; RNN; I-PCA.

1. Introduction

Stock price prediction has emerged as a critical area of research and application in finance, leveraging advanced computational techniques to anticipate future stock movements. The dynamic nature of financial markets, driven by numerous factors such as economic indicators, geopolitical events, and investor sentiment, makes predicting stock prices a challenging yet valuable endeavor [1,2]. Accurate predictions can provide investors with a competitive edge, enabling informed decision-making and optimized portfolio management. Traditionally, stock price prediction relied on fundamental and technical analysis [3]. Fundamental analysis evaluates a company's intrinsic value based on financial statements, market conditions, and

industry trends, while technical analysis examines past price and volume data to identify patterns and trends [4,5]. While these approaches offer valuable insights, they often fall short in capturing the complexity of modern financial markets characterized by high volatility, interdependencies, and massive data flows. With the advent of big data and machine learning, stock price prediction has witnessed a paradigm shift [6]. Machine learning models, capable of processing and analyzing vast amounts of data in real time, have proven to be highly effective in forecasting stock prices. These models use historical data, market sentiment, macroeconomic indicators, and other features to identify patterns that are often imperceptible to traditional methods [7,8]. Techniques such as regression, support vector machines (SVM), and decision trees are commonly used for this purpose.

In recent years, deep learning techniques, a subset of machine learning, have gained prominence in stock price prediction [9]. Models such as RNN and LSTM networks are particularly suitable for time-series forecasting due to their ability to capture sequential dependencies and temporal patterns in data [10,11]. Additionally, Convolutional Neural Networks (CNN) and hybrid architectures combining CNN and LSTM have demonstrated success in improving prediction accuracy by extracting spatial and temporal features from stock market data. Another significant advancement in this field is the integration of sentiment analysis and natural language processing (NLP). By analyzing news articles, social media posts, and earnings call transcripts, researchers and practitioners can incorporate market sentiment as a feature in predictive models [12]. Sentiment-driven predictions add a layer of contextual understanding, allowing models to better anticipate market reactions to events and trends [13]. However, the task of predicting stock prices is fraught with challenges. Financial markets are influenced by numerous unpredictable factors, such as sudden political developments, natural disasters, or changes in regulatory policies. The efficient market hypothesis (EMH) posits that stock prices reflect all available information, suggesting that markets are inherently unpredictable. Despite this, machine learning and deep learning approaches strive to uncover exploitable patterns and inefficiencies, albeit with varying degrees of success [14].

This study focuses on the application of advanced machine learning and deep learning techniques for stock price prediction, aiming to improve forecasting accuracy and reliability. By exploring innovative methods, such as ensemble learning, attention mechanisms, and reinforcement learning, it seeks to contribute to the growing body of knowledge in financial technology. The ultimate goal is to provide robust predictive tools that cater to the needs of individual investors, institutional traders, and financial analysts in navigating the complexities of modern financial markets [15].

The contributions of this paper are manifested below,

- This study introduces a novel hybrid model combining RNN-LSTM, optimized using the EGWSCS algorithm. This model effectively addresses the challenges of predicting stock prices based on sentiment analysis, handling the complex and volatile nature of financial data.
- The study leverages STOCKBERT, a domain-specific language model, for extracting sentiment features from financial news and social media data. This method captures the nuanced financial sentiment, improving the accuracy of stock price predictions compared to generic sentiment analysis techniques.

- To enhance processing efficiency, the study employs I-PCA for reducing the dimensionality of feature data. This step ensures faster processing while retaining the critical information needed for accurate stock price prediction.

- The study utilizes the EGWSCS algorithm, a combination of Grey Wolf Optimization (GWO) and Sand Cat Swarm Optimization (SCSO), to fine-tune the hyperparameters of the model. This optimization improves prediction accuracy and helps avoid local optima, ensuring a more robust and reliable stock price prediction model.

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

2. Literature Review

In 2024, Zhang and Chen [16] introduced a novel two-stage model integrating a decomposition algorithm, a nonlinear ensemble strategy, and three machine learning models. First, variational mode decomposition (VMD) decomposes stock price time series into sub-series. Support vector regression (SVR), extreme learning machine (ELM), and deep neural networks (DNN) predict these sub-series independently, with their outputs aggregated into preliminary predictions. In the second stage, an ELM-based nonlinear ensemble combines these predictions to enhance overall stock price prediction accuracy.

In 2023, Li et al. [17] proposed a clustering-enhanced deep learning framework utilizing LSTM, RNN, and Gated Recurrent Unit (GRU). The framework incorporates clustering as a pre-processing step to improve model training quality. A novel similarity measure, Logistic Weighted Dynamic Time Warping (LWDTW), is introduced, modifying Weighted Dynamic Time Warping (WDTW) with a logistic probability density function to better capture return observation importance.

In 2023, Velu et al. [18] used financial market forecasting frameworks by integrating sentiment analysis of selected tweets into a long-short recurrent neural channel. A novel sentiment analysis approach combines psychological labelling with valence ratings to quantify sentiment strength. The analysis generates features such as 2-level and 3-level polarization, gross reactivity, and total valence. The emotional polarity explicitly annotated in the dataset aligns well with the results of the innovative lexicon-based approach, enhancing the predictive capability of the forecasting model.

In 2023, Wang and Zhu [19] developed a two-stage deep learning prediction system integrating intelligent optimization and feature clustering to enhance prediction accuracy. Multi-factor analysis and XGBoost eliminate weakly correlated factors, while clustering organizes features for prediction. Optimized GA-LSTM models predict each cluster, with their results nonlinearly integrated for final predictions applied to the test set.

In 2023, Yang et al. [20] proposed a neural network-based model optimized using an improved Particle Swarm Optimization (PSO) algorithm. By introducing adaptive inertial weight adjustments, the approach maintains particle diversity, enhancing global search during early

evolution and local search in later stages. The improved algorithm is integrated with a neural network to construct a robust prediction system.

In 2023, Ji et al. [21] developed an attention-based LSTM (ALSTM) model incorporating stock prices, technical indicators, and social media sentiment. Using a fine-tuned BERT sentiment classifier and lexicon-based methods, we analyze three years of data from 28 stocks. Results show superior performance in Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and accuracy, with BERT-based sentiment features and a 5-day input window yielding optimal results.

In 2023, Yan et al. [22] presented a hybrid approach combining AdaBoost feature selection with a deep learning model for predicting stock index futures prices. Specifically, the sklearn-wrapped AdaBoost regressor is employed for feature selection, while a two-layer Long Short-Term Memory (LSTM) predictor is developed for forecasting. Performance evaluations demonstrate that the proposed model consistently outperforms other popular prediction models, including random forest, multilayer perceptron, gated recurrent unit, deep belief network, and stacked denoising autoencoder.

In 2023, Wang et al. [23] used a novel approach combining knowledge graphs (KG) and graph convolutional networks (GCNs). By constructing a stock KG, quantifying relationships, and merging K-means, community detection, and GCNs, we predict trends using historical data. Testing on 4684 Chinese A-share stocks (2013–2019) validates the approach.

In 2022, Srivinay et al. [24] proposed a hybrid model combining Prediction Rule Ensembles (PRE) and Deep Neural Networks (DNN). The model first utilizes moving averages (20, 50, and 200 days) to identify trends. Then, PRE generates rules based on RMSE scores, which are refined by fine-tuning DNN hyperparameters. The final model combines PRE and DNN results, evaluated with MAE and RMSE metrics.

In 2021, Zhang and Lou [25] applied neural networks and the Backpropagation (BP) algorithm to classify and predict stock price patterns. Using the BP algorithm, transaction data from five consecutive days serves as input (20 input nodes), with the next day's closing price as the output. After training the model, test data is used to evaluate prediction performance by analyzing the error between actual and predicted stock prices.

2.1. Problem Statement

Stock price prediction is a complex and challenging task due to the inherent volatility and unpredictability of financial markets. Traditional methods, such as fundamental and technical analysis, often fail to capture the dynamic and high-dimensional nature of stock data. With the growing influence of social media and news platforms, sentiment analysis has gained attention as an effective tool for understanding market movements. However, predicting stock prices based on sentiment is further complicated by the noisy, unstructured, and ever-changing nature of textual data. Existing models struggle to handle the high volume of data generated daily and often face issues like overfitting and local optima during training. This study aims to develop an advanced hybrid model that combines RNN and LSTM networks optimized using the EGWSCS algorithm, to improve the accuracy and robustness of stock price predictions based on sentiment analysis.

3. Proposed Methodology

Stock price prediction is a complex task due to the volatile and dynamic nature of financial markets. Traditional methods struggle with high-dimensional, noisy, and unstructured data, particularly from social media platforms like Twitter. The challenge lies in accurately interpreting sentiment from textual data and predicting stock prices amid unpredictable market influences. This study overcomes these challenges by using a hybrid model that integrates RNN and LSTM networks, optimized through the EGSWCS algorithm. Preprocessing techniques, and dimensionality extraction, and selection of features help improve prediction accuracy and handle noisy data effectively. Fig. 1 depicts the overall proposed architecture.

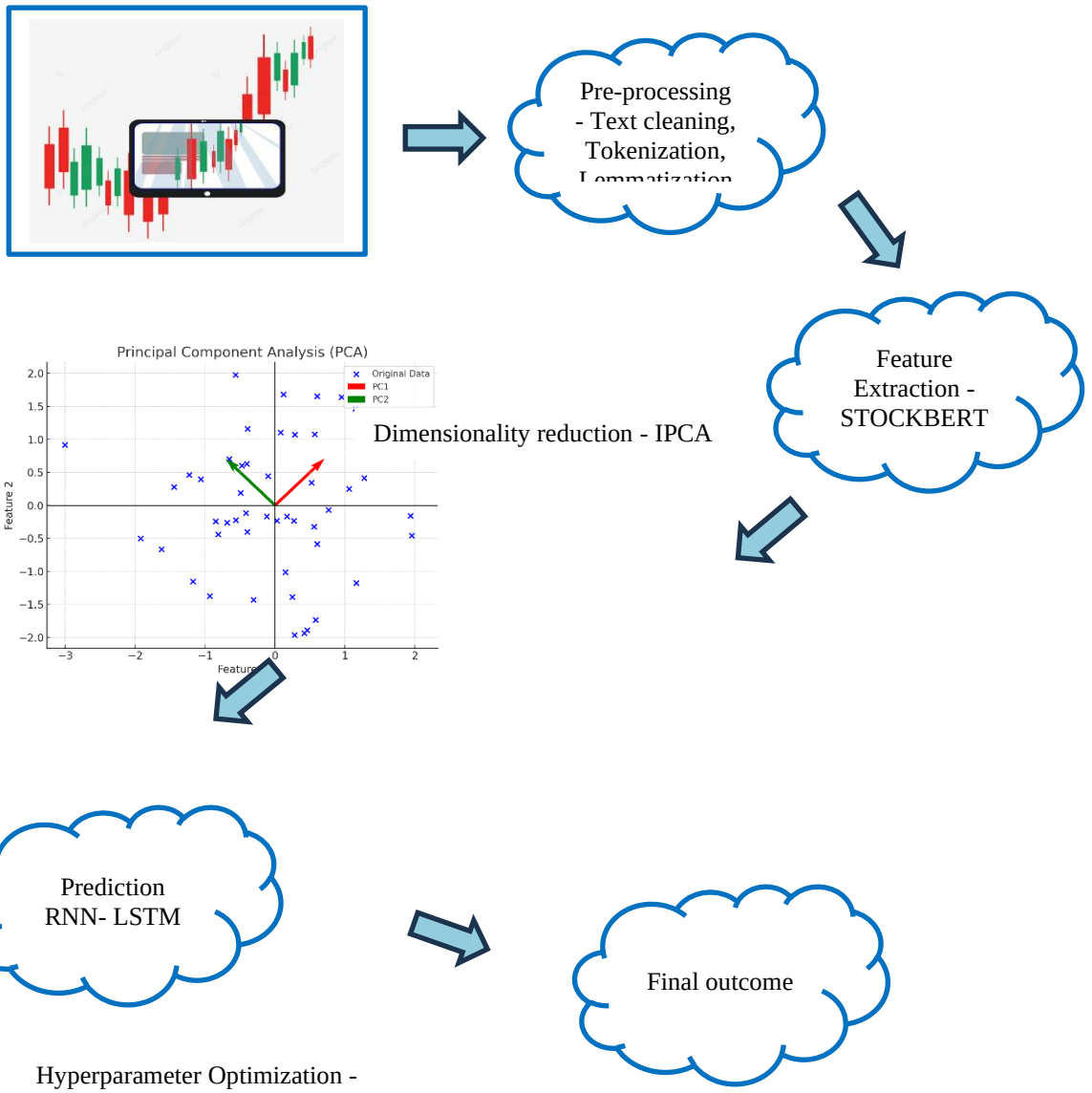


Figure 1: Overall Proposed Architecture

3.1. Data Preprocessing

This research applies data preprocessing techniques including text cleaning, tokenization, and 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 many text processing tasks, enabling computers to understand and work with human language more effectively.

3.1.3. 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. Sentiment Feature Extraction using STOCKBERT

STOCKBERT is a domain-specific natural language processing (NLP) model designed for financial sentiment analysis, particularly in the context of stock market data. Built upon the foundation of Bidirectional Encoder Representations from Transformers (BERT), STOCKBERT is fine-tuned to process and analyze stock market-related content, such as

financial news, social media posts, earnings reports, and market analyses. The main objective of STOCKBERT is to better understand and classify the sentiment expressed in stock market texts, enabling more accurate predictions of stock price movements and assisting investors in making data-driven decisions. BERT, developed by Google, is a transformer-based model that utilizes bidirectional attention to understand the context of a word within a sentence. This allows it to capture complex semantic relationships in text better than traditional models that analyze text in one direction. However, BERT is not tailored to any specific domain, and thus, for financial applications like stock market sentiment analysis, it requires fine-tuning on domain-specific data. STOCKBERT addresses this need by training the model on a large corpus of stock-related texts to enhance its ability to understand financial language.

The training process for STOCKBERT involves two main phases: pre-training and fine-tuning. In the pre-training phase, STOCKBERT learns general language patterns from a large corpus of stock market content, including news articles, social media posts, and other relevant textual sources. This step enables the model to understand basic financial terminology, sentence structures, and relationships between different entities, such as stock tickers, companies, and market events. The fine-tuning phase is where STOCKBERT adapts specifically to the task of sentiment classification. It is trained on labelled datasets, where texts are categorized into sentiments—positive, negative, or neutral. This phase helps STOCKBERT specialize in predicting the market sentiment towards specific stocks or sectors, enhancing its ability to forecast stock price movements.

One of the key advantages of STOCKBERT over general models like BERT is its ability to handle the unique linguistic patterns of financial texts. Financial content often contains jargon, abbreviations, and context-dependent terms that differ significantly from everyday language. STOCKBERT is trained to recognize these nuances, enabling it to classify stock-related sentiments more accurately. Furthermore, STOCKBERT can be used in conjunction with other predictive models, such as time-series forecasting or machine learning classifiers, to predict stock price movements. By analyzing market sentiment in real time, STOCKBERT helps investors stay ahead of market trends, providing actionable insights for trading strategies and risk management. This makes STOCKBERT an invaluable tool for anyone involved in the financial markets, offering a sophisticated approach to sentiment analysis and improving decision-making based on textual data.

3.3. Dimensionality Reduction with I-PCA

Dimensionality reduction aims to streamline modelling by reducing the number of variables. It encompasses feature selection, which involves choosing significant variables, and feature extraction, which transforms high-dimensional data into fewer dimensions. This process accelerates model training and enhances accuracy by mitigating overfitting. This study used PCA for dimensionality reduction.

3.3.1. I-PCA

I-PCA incorporates pixel pair weights to emphasize significant relationships during dimensionality reduction. Here, $w(i,j)$ assigns importance to pixel pairs, influencing the covariance matrix calculation, ensuring dominant features are retained for enhanced image analysis and representation. PCA is a statistical technique used for dimensionality reduction

and data compression. It works by transforming high-dimensional data into a lower-dimensional representation while preserving most of the important information. The main idea behind PCA is to identify the directions (or principal components) along which the data varies the most. These principal components are linear combinations of the original features and are orthogonal to each other. PCA accomplishes this by finding the eigenvectors and eigenvalues of the covariance matrix of the data. The eigenvectors represent the principal components, and the eigenvalues indicate the amount of variance explained by each principal component. After computing the principal components, PCA projects the original data onto these components, effectively reducing the dimensionality of the data. This projection retains the maximum amount of variability present in the original data. Here's a brief explanation of PCA.

1. Covariance Matrix Calculation

Given a dataset X with n observations and p features, calculate the covariance matrix C is expressed as per Eq. (1).

$$C = \frac{1}{n-1}(X - \overline{X})^T(X - \overline{X}) \quad (1)$$

2. Eigenvalue Decomposition

As per the proposed Eq. (2), compute the eigenvalues $(\lambda_1, \lambda_2, \ldots, \lambda_p)$ and corresponding eigenvectors $(v_1, v_2, \ldots, v_p)$ of the covariance matrixC.

$$Cv_i = \lambda_i v_i \cdot w(i,j) \quad (2)$$

3. Sort Eigenvalues

Sort the eigenvalues in descending order and rearrange the corresponding eigenvectors accordingly.

4. Select Principal Components

Choose the first k eigenvectors corresponding to the k largest eigenvalues to form the transformation matrix M.

5. Project Data

Project the original data X onto the new subspace defined by M to obtain the transformed datasetY. PCA reduces the dimensionality of the dataset while preserving the maximum variance. It finds the directions (principal components) along which the data varies the most and projects the data onto these components, effectively capturing the essential information in a lower-dimensional space.

3.4. Model Design

The chosen features are inputted into the classification stage utilizing RNN, and LSTM models.

3.4.1. RNN-LSTM

The hybridization of RNN and LSTM occurs by integrating their functionalities into a unified model architecture. The input data is first processed by an RNN layer. This layer captures the short-term dependencies and immediate sequential patterns in the data. The RNN computes

the hidden states hs_t iteratively for each time step using Eq. (3) and generates intermediate representations. The output from the RNN layer is passed as input to an LSTM layer. The LSTM focuses on retaining and analyzing long-term dependencies and contextual patterns in the sequential data. The LSTM computes activations l_t for each time step using Eq. (5), leveraging its gating mechanisms (input, forget, and output gates) to decide which information to retain or discard. The outputs from both RNN and LSTM layers are merged, either through concatenation, addition, or another suitable fusion technique, creating a hybrid representation of the sequential data. This fused representation incorporates both short-term patterns (from the RNN) and long-term dependencies (from the LSTM). The combined hybrid features are passed to the output layer, where predictions are made based on the task.

The Hybrid RNN-LSTM model integrates the strengths of RNN and LSTM networks to effectively process and analyze sequential data. This combination capitalizes on the RNN's ability to model sequential dependencies and the LSTM's capability to capture long-term dependencies, addressing challenges such as gradient vanishing in traditional RNN. An RNN is a neural network designed to process sequential data by retaining hidden state information through directed cycles in its architecture. This feature makes RNN effective for tasks involving time-series data and natural language processing. The hidden state hs_t at time t in an RNN is computed using Eq. (3).

$$hs_t = \sigma(w_{hsi} i_t + w_{hshs} hs_{t-1} + bi_{hs}) \quad (3)$$

Here, 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 as per Eq. (4).

$$o_t = \text{softmax}(w_{ohs} hs_t + bi_o) \quad (4)$$

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.

LSTM networks enhance RNN by incorporating memory cells that retain information over longer periods, solving the gradient vanishing problem. Each LSTM unit includes three gates,

- Input Gate: Controls the flow of new data into the memory cell.
- Forget Gate: Decides what information to discard from the memory.
- Output Gate: Determines the output based on the current memory state and input.

The activation of each LSTM unit at time l_t is calculated using Eq. (5):

$$l_t = \sigma(wm_{i,l} \cdot x_t + wm_{h,l} \cdot l_{t-1} + bi) \quad (5)$$

Where,l_t and l_{t-1} represent the activation at time respectively, σ is a non-linear activation function, $wm_{i,l}$ is the input-hidden weight matrix, $wm_{h,l}$ is the hidden-hidden weight matrix, bi is the hidden bias vector, and x_t is the input at time t. LSTM networks excel at capturing long-term dependencies in sequential data. They mitigate the problem of gradient vanishing,

allowing for more effective learning over longer sequences. EGWSCS algorithm is used to fine-tune hyperparameters in the hybrid RNN-LSTM model. Fig. 2 depicts the RNN-LSTM architecture.

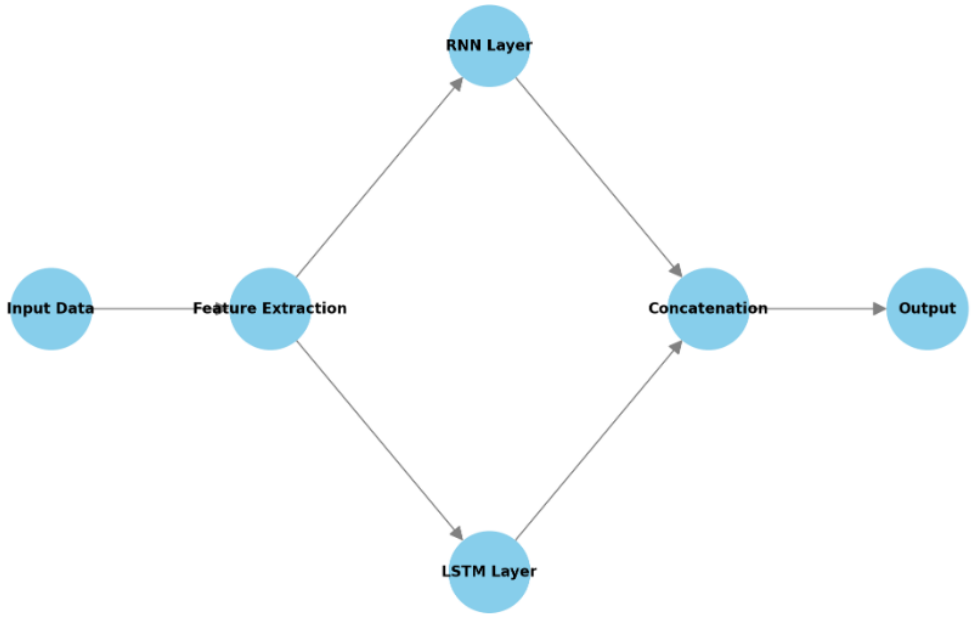


Figure 2: RNN-LSTM architecture

3.5. Optimization Using EGWSCS

EGWSCS algorithm is an optimization method that combines the principles of Grey Wolf Optimizer (GWO) and Sand Cat Search Optimization (SCSO). EGWSCS algorithm enhances the original SCSO and GWO by incorporating strategies to avoid local optima and improve convergence speed. By utilizing fitness-based updates and enhanced search strategies, EGWSCS achieves better global optima discovery. Incorporating GWO into SCSO enhances exploration by combining GWO's hierarchical social structure with SCSO's prey-search behavior, improving convergence speed, preventing local optima, and optimizing complex problem-solving tasks more effectively. SCSO algorithm, introduced in 2022, is a meta-heuristic optimization technique inspired by the hunting behavior of sand cats. Sand cats rely on sound frequency to locate prey and decide whether to attack or continue searching. In SCSO, each cat represents a candidate solution, and the search for prey mimics the exploration of the solution space. While SCSO demonstrates strong initial exploration capabilities, it often falls into local optima during later stages, limiting optimization performance. This challenge is addressed in improved variants of the algorithm, which enhance its convergence and ability to find global optima. It adjusts the search process to escape local traps, enhancing optimization accuracy and performance, in complex problem-solving and feature selection tasks. The algorithm initializes a population, employs Triangle Walk and Levy Flight strategies for exploration, and uses fitness-based updates for optimization.

3.3.1. Prey Searching

The location of each sand cat is represented by its position $posit_i$ throughout the SCSO algorithm's Search for Prey stage. Leveraging their acute hearing, sensitive to frequencies below 2 kHz, they efficiently explore prey locations. The sensitivity range s_g Eq. (6) and parameter S Eq. (7) optimize exploration-exploitation balance.

$$s_g = M_S - \left(\frac{M_S \times t}{T}\right) \cdot \vec{A}\,\vec{C} \quad (6)$$

$$S = 2s_g\,rand(0,1) - s_g \quad (7)$$

Incorporating $\vec{A}$ and $\vec{C}$ from the GWO into the SCSO enhances its capability to balance exploration and exploitation. The vectors A and C are computed as follows as per Eq. (8) and Eq. (9).

$$\vec{A} = 2\vec{a} \cdot \vec{r_1} - \vec{a} \quad (8)$$

$$\vec{C} = 2\vec{r_2} \quad (9)$$

By varying (s) for sand cat, as shown in Eq. (10) and Eq. (11), the algorithm promotes both exploration (finding diverse solutions) and exploitation (refining solutions). This adaptive sensitivity reduces the likelihood of the algorithm getting trapped in local optima, thereby improving the overall search effectiveness in the optimization process.

$$s = s_g\,rand(0,1) \quad (10)$$

$$posit(t+1) = s \times (posit_p(t) - rand(0,1) \times posit_c(t)) \quad (11)$$

3.3.2. Walk Strategy-Triangle

In the Triangle Walk Strategy, sand cats calculate distance D_1 Eq. (12), D_2 step size Eq. (13), and direction angle β Eq. (14). The updated position Eq. (15 and 16) improves exploration during prey pursuit.

$$D_1 = posit_p(t) - posit_c(t) \quad (12)$$

$$\overrightarrow{D_2} = rand() \times \overrightarrow{D_1} \quad (13)$$

$$\beta = 2\pi rand() \quad (14)$$

$$P = {D_1}^2 + {D_2}^2 - 2D_1D_2\cos\beta \quad (15)$$

$$posit_{new} = posit_p(t) + sP \quad (16)$$

Among them, $posit_{new}$ define position.

3.3.3. Prey Attacking

The distance that lies among the sand cat and the prey is determined during the Attack Prey stage Eq. (17). The Roulette Wheel selection algorithm determines a random direction angle α, enabling varied movement. The attack process is modelled using Eq. (18) for quick prey approach.

$$\text{posit}_{\text{scp}} = \left|\text{rand}(0,1) \times \text{posit}_{\text{p}}(t) - \text{posit}_{\text{c}}(t)\right| \quad (17)$$

$$\text{posit}(t+1) = \text{posit}_{\text{p}}(t) - s \times \text{posit}_{\text{scp}} \times \cos(\alpha) \quad (18)$$

3.3.4. Walk Strategy- Levy Flight

During the attack, the sand cat's movement is adjusted using a constant $A = 0.35$ in the Levy flight, refining its approach to prey. This modification enhances movement precision, as detailed in Eq. (19).

$$\text{posit}_{\text{new}} = \text{posit}_{\text{p}}(t) + (\text{posit}_{\text{p}}(t) - \text{posit}_{\text{c}}(t)) \times A \times \text{levy} \quad (19)$$

3.3.5. Population Position Update

Fitness metrics are compared to update the location. The original person will be replaced once the fitness value derived from the upgrade is higher. The original individual will be preserved since their fitness value is higher than that of the recently acquired person. Algorithm 1 demonstrates the step-by-step process of implementing the proposed methodology, highlighting key operations, optimizations, and workflows.

Algorithm 1: Optimized Stock Sentiment Prediction

1. Input
2. Preprocessing
   a. clean text data (remove special characters, stopwords)
   b. Tokenize and lemmatize text
   c. Extract sentiment features using STOCKBERT
3. Feature Dimensionality Reduction
   a. Apply Improved PCA (I-PCA) to reduce high-dimensional sentiment features
4. Model Initialization
   a. Define hybrid RNN-LSTM model architecture
      - RNN layers for capturing sequential dependencies
      - LSTM layers for handling long-term dependencies
   b. Initialize EGWSCS for hyperparameter optimization
5. Hyperparameter Optimization using EGWSCS
   a. Initialize wolf and sand cat populations
   b. Evaluate fitness (prediction accuracy) of each candidate solution
   c. Update positions of wolves and sand cats for better exploration and exploitation
   d. Select optimized hyperparameters (e.g., learning rate, layer sizes).
6. Training

a. Train the hybrid RNN-LSTM model on pre-processed sentiment and stock price data

b. Use optimized hyperparameters from EGWSCS

7. Prediction

a. Input new sentiment data to the trained model

b. Predict stock price movements

8. Output

Predicted stock price trends based on sentiment analysis

9. End

4. Result and Discussion

The results of the proposed hybrid model, which combines RNN and LSTM networks optimized by the EGWSCS algorithm, indicate significant advancements in stock price prediction based on sentiment analysis. The model achieved an accuracy of 0.94, a precision of 0.944, a recall of 0.938, and an F1-score of 0.941, demonstrating its ability to deliver consistent and reliable predictions across various datasets. This impressive performance can be attributed to several key factors. First, the use of EGWSCS optimization played a crucial role in fine-tuning the model's hyperparameters, ensuring optimal performance and overcoming the common challenge of local optima traps. By leveraging the strengths of both GWO and SCSO, the EGWSCS algorithm enhanced the model’s predictive capability.

Second, the hybrid RNN-LSTM architecture effectively captured the sequential patterns and temporal dynamics of the financial data. While RNN processed sequential information, LSTM addressed the long-term dependencies often inherent in financial trends, making the model robust to noisy and dynamic data. The combination of these elements enabled the model to balance precision and recall, as evidenced by its high F1-score. These findings not only demonstrate the model's capability to address the challenges of sentiment-based stock price prediction but also highlight its potential for adoption in real-world financial decision-making scenarios.

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.

4.2. Graphical Representation of Existing and Proposed Model

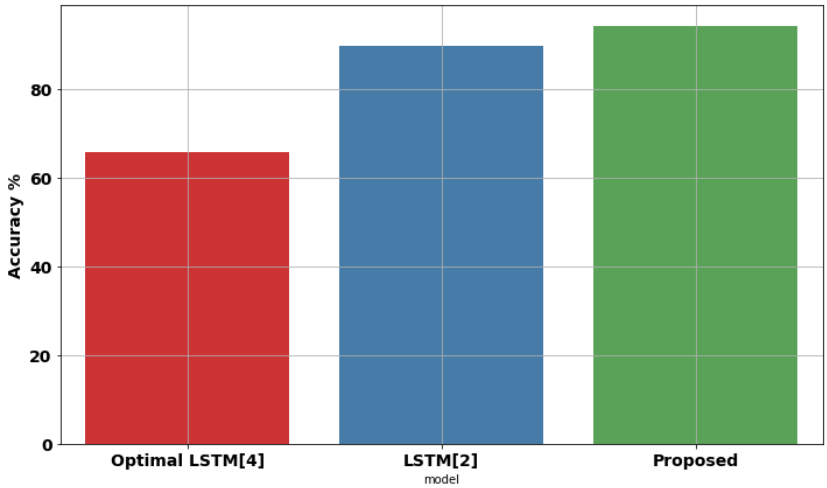


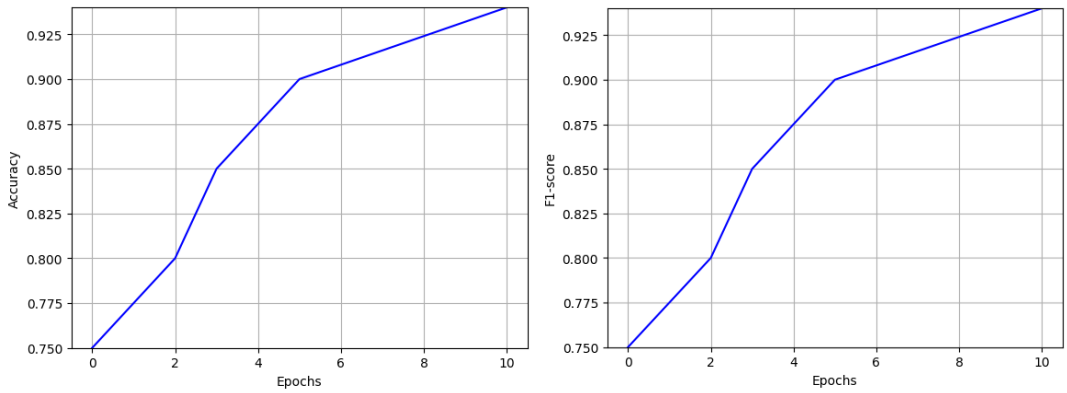
Figure 3: Existing vs Proposed Graph

Fig. 3 compares the existing and proposed models through performance metrics, highlighting significant improvements in accuracy achieved by the proposed method over the existing approach.

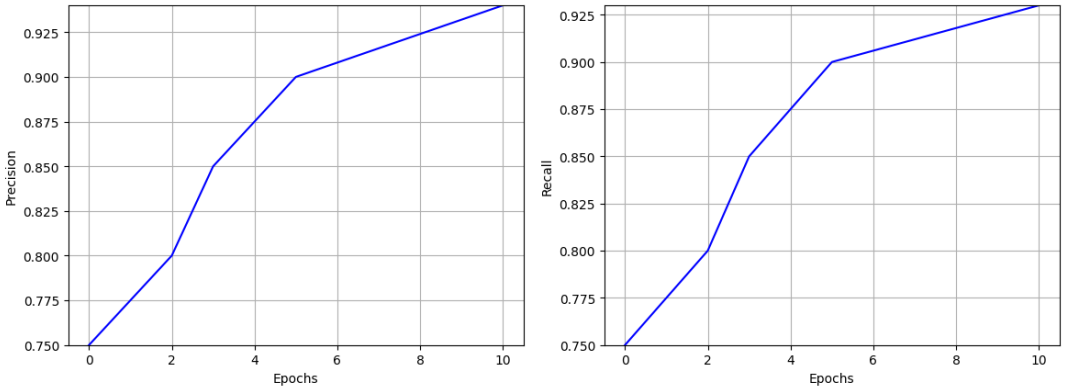
Table 1: Comparative Analysis of Existing and Proposed Model

Methods	Accuracy (%)
Optimal LSTM [4]	65.64
LSTM [2]	89.7
Proposed	94.0

Table 1 compares the accuracy of different models. The Optimal LSTM [4] achieves 65.64%, while the standard LSTM [2] shows a significant improvement at 89.7%. The proposed model outperforms both, achieving a higher accuracy of 94%. This demonstrates the effectiveness of the proposed model in improving performance.



(a) (b)



(c) (d)

Figure 4: Performance Metrics Varying Epochs

Figures 4, 5, and 6 collectively illustrate the performance of the proposed stock price prediction model. Fig. 4 shows the performance metrics varying epochs demonstrates how precision, recall, and F1-score evolve with training epochs, helping identify the optimal epoch count where the model achieves peak performance. Fig. 5 depicts the accuracy vs loss graph showcases the progression of training and validation accuracy alongside loss over epochs, indicating effective model learning as accuracy increases and loss decreases steadily. Finally, Fig. 6 illustrates the confusion matrix for the testing phase, summarizing the model's performance by displaying true positive, true negative, false positive, and false negative rates for sentiment classification.

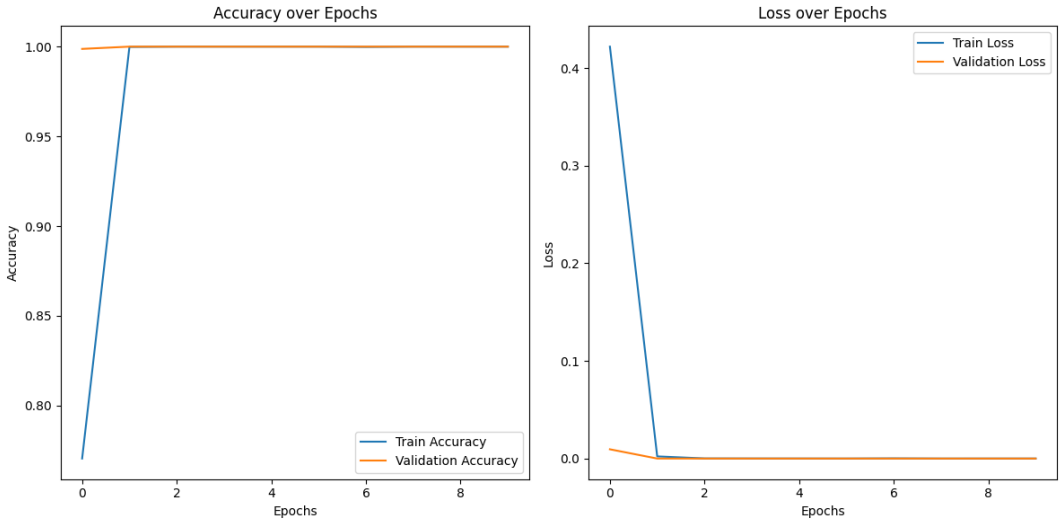


Figure 5: Accuracy vs Loss Graph

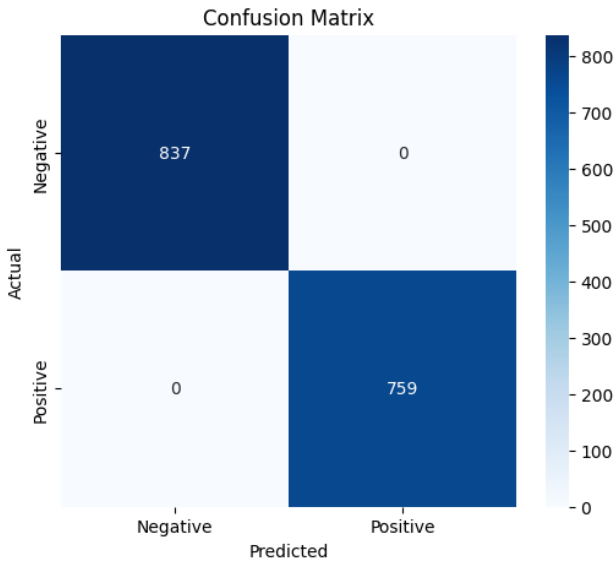


Figure 6: Testing Phase - Confusion Matrix

5. Conclusion

This study addressed these challenges by proposing a hybrid model that integrated RNN and LSTM networks, optimized using the EGWSCS algorithm, which combined GWO and SCSO. Sentiment features were extracted using STOCKBERT, a domain-specific language model designed to capture nuanced financial sentiment. The textual data underwent preprocessing steps such as text cleaning, tokenization, and lemmatization. I-PCA was employed to reduce feature dimensionality, ensuring efficient processing. The EGWSCS algorithm fine-tuned the model's hyperparameters, enhancing prediction accuracy and mitigating the risk of local optima. The model achieved a high accuracy of 94%, demonstrating its effectiveness in stock price prediction. Simulation results further confirmed that the proposed model provided superior performance in sentiment analysis-driven stock price prediction, offering a robust framework for decision-making in financial markets.

References

1. Lu, M. and Xu, X., 2024. TRNN: An efficient time-series recurrent neural network for stock price prediction. Information Sciences, 657, p.119951.
2. Agrawal, M., Khan, A.U. and Shukla, P.K., 2019. Stock price prediction using technical indicators: a predictive model using optimal deep learning. Learning, 6(2), p.7.
3. Zhang, J., Ye, L. and Lai, Y., 2023. Stock price prediction using CNN-BiLSTM-Attention model. Mathematics, 11(9), p.1985.
4. Billah, M.M., Sultana, A., Bhuiyan, F. and Kaosar, M.G., 2024. Stock price prediction: comparison of different moving average techniques using deep learning model. Neural Computing and Applications, 36(11), pp.5861-5871.
5. Chen, J., Wen, Y., Nanehkaran, Y.A., Suzauddola, M.D., Chen, W. and Zhang, D., 2023. Machine

learning techniques for stock price prediction and graphic signal recognition. Engineering Applications of Artificial Intelligence, 121, p.106038.
6. Wang, S., 2023. A stock price prediction method based on BiLSTM and improved transformer. IEEE Access.
7. Habib, H., Kashyap, G.S., Tabassum, N. and Tabrez, N., 2023. Stock price prediction using artificial intelligence based on LSTM–deep learning model. In Artificial Intelligence & Blockchain in Cyber Physical Systems (pp. 93-99). CRC Press.
8. Mu, G., Gao, N., Wang, Y. and Dai, L., 2023. A stock price prediction model based on investor sentiment and optimized deep learning. Ieee Access, 11, pp.51353-51367.
9. Qi, C., Ren, J. and Su, J., 2023. GRU neural network based on CEEMDAN–wavelet for stock price prediction. Applied Sciences, 13(12), p.7104.
10. Ma, D., Yuan, D., Huang, M. and Dong, L., 2024. VGC-GAN: A multi-graph convolution adversarial network for stock price prediction. Expert Systems with Applications, 236, p.121204.
11. Jiang, M., Chen, W., Xu, H. and Liu, Y., 2024. A novel interval dual convolutional neural network method for interval-valued stock price prediction. Pattern Recognition, 145, p.109920.
12. Zhao, C., Hu, P., Liu, X., Lan, X. and Zhang, H., 2023. Stock market analysis using time series relational models for stock price prediction. Mathematics, 11(5), p.1130.
13. 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.
14. Chen, J., 2023. Analysis of bitcoin price prediction using machine learning. Journal of Risk and Financial Management, 16(1), p.51.
15. Nasiri, H. and Ebadzadeh, M.M., 2023. Multi-step-ahead stock price prediction using recurrent fuzzy neural network and variational mode decomposition. Applied Soft Computing, 148, p.110867.
16. Zhang, J. and Chen, X., 2024. A two-stage model for stock price prediction based on variational mode decomposition and ensemble machine learning method. Soft Computing, 28(3), pp.2385-2408.
17. Li, M., Zhu, Y., Shen, Y. and Angelova, M., 2023. Clustering-enhanced stock price prediction using deep learning. World Wide Web, 26(1), pp.207-232.
18. Velu, S.R., Ravi, V. and Tabianan, K., 2023. Multi-lexicon classification and valence-based sentiment analysis as features for deep neural stock price prediction. Sci, 5(1), p.8.
19. Wang, J. and Zhu, S., 2023. A multi-factor two-stage deep integration model for stock price prediction based on intelligent optimization and feature clustering. Artificial Intelligence Review, 56(7), pp.7237-7262.
20. Yang, F., Chen, J. and Liu, Y., 2023. Improved and optimized recurrent neural network based on PSO and its application in stock price prediction. Soft computing, 27(6), pp.3461-3476.
21. Ji, Z., Wu, P., Ling, C. and Zhu, P., 2023. Exploring the impact of investor's sentiment tendency in varying input window length for stock price prediction. Multimedia Tools and Applications, 82(18), pp.27415-27449.
22. Yan, W.L., 2023. Stock index futures price prediction using feature selection and deep learning. The North American Journal of Economics and Finance, 64, p.101867.
23. Wang, T., Guo, J., Shan, Y., Zhang, Y., Peng, B. and Wu, Z., 2023. A knowledge graph–GCN–community detection integrated model for large-scale stock price prediction. Applied Soft Computing, 145, p.110595.
24. Srivinay, Manujakshi, B.C., Kabadi, M.G. and Naik, N., 2022. A hybrid stock price prediction model based on PRE and deep neural network. Data, 7(5), p.51.
25. Zhang, D. and Lou, S., 2021. The application research of neural network and BP algorithm in stock price pattern classification and prediction. Future Generation Computer Systems, 115, pp.872-879.
26. Dataset collected from: "https://www.kaggle.com/datasets/liqiang2022/2022-sse-50-dataset", dated 1/12/2024.