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

Synergizing Sentiment Analysis in Stock Tweets: Employing Adaptive Dolphin Swarm Optimization-Based Resilient Recurrent Neural Networks (Adso-Rrnn) for Scalable Data Handling and Enhanced Market Prognostication

G. Priyadarshini[1,2], Dr. D. Karthika[3]

[1]Research Scholar, P.K.R. Arts College for Women, India
[2]Assistant Professor, KG College of Arts and Science, India
[3]Associate Professor and Head, Department of Computer Science, VET Institute of Arts and Science (Co-education) College, Erode, Tamil Nadu, India
Email: savidpri@gmail.com

To decipher the sentiments expressed in textual data is one of the most important part of natural language processing, known as sentiment analysis. In finance, where public opinion matters, and public opinion is greatly influenced by sentiments, Twitter and other social media platforms can be an invaluable piece of the puzzle for determining public opinion. Sentiment analysis over tweets is itself a challenging job since tweets often contain informal language, less context and are short. In this study, we propose a novel approach to address these challenges to incorporating the ADSO into the framework that leverages the adaptive dolphin swarm optimization, and RRNN for sentiment identification in stock related tweets. In this synergistic fusion the modeling performance is optimized, improving adaptability, scalability, and accuracy of sentiments in analysis tasks but especially for large scale data processing. Proposed ADSO-RRNN framework provides an effective approach to address the immense amount of data and provides accuracy by capturing nuanced sentiments from dynamic sources such as Twitter. Evaluation of this framework is evaluated on the 'Stock Tweets for Sentiment Analysis and Prediction' dataset that comprises a full set of

stock related tweets from various sources such as the popular Twitter's API. The effectiveness and applicability of the ADSO-RRNN framework are rigorously tested across varied market scenarios and the resulting insights can be significant for real world financial analysis and decision.
Keywords: Sentiment analysis, Resilient Recurrent Neural Networks (RRNN), Adaptive Dolphin Swarm Optimization (ADSO), Stock tweets, Financial analysis.

1. Introduction

Natural Language Processing has a powerful tool in the form of Sentiment analysis, also known as opinion mining. It means, extracting, and interpreting the emotions, opinions, and attitudes from text data. Using complex algorithms, sentiment analysis can determine if a written text is displaying positivity, negativity or neutrality; helping businesses and individuals to understand what the general public thinks[1]. Apart from that, this technology has vast applications in social media monitoring, customer feedback analysis, and market research, so as to help organizations take better data driven decisions. Gauging sentiment accurately has become an application of lexicons, machine learning, and deep learning techniques that the sentiment analysis algorithms depend on. That's why it's such an important part of our data driven world today, impacting everything from marketing efforts, to political campaigns[2].

Then there are Stock Tweets: The quick bursts of messaging that proliferate on things like Twitter give a real time view of the financial world. These 280 character or less brief posts are being turned to by traders and investors for instant insights, opinions or reactions to market events. Stock Tweets is one of the fascinating things about them democratizing finance. [3] Are an open platform that allows both seasoned and interested investors, anyone in the world, to share their views on stocks and market trends at the same time. This kind of inclusivity entails diverse perspectives as well as market analysis, and hence makes an exciting online community. Stock Tweets can also be this rapid fire. Complex financial concepts are usually oversimplified when it comes to their brevity. When you receive information from these tweets, you should exercise caution and do your proper research before taking action. Stock Tweets offers a dynamic, accessible view into the financial world and a forum for any and all voices to contribute to the conversation around stocks and investments.

2. Literature Review

The work titled 'Stock Market Movement Prediction' [30] analyzes the effects of public sentiment toward predictability of stock market movements globally during the COVID 19 pandemic, in particular public sentiment expressed in social media platforms. This paper examines whether, and the extent to which, public sentiment (pessimism and optimism) in economics news influenced investors' behavior and stock prices during a time of significant uncertainty as to the economy. The analysis of the "Italy Country Reputation and Stock Market Performance" [31] used sentiment analysis to analyze the impact of the COVID-19 pandemic on the global image of Italy and the corresponding stock market. To find out the impact on Italy, as a nation, this investigates public sentiment and perceptions as they are tackled in news

articles, social media discussions and online discourse. The 'machine learning classifiers'[32] examine the application of machine learning in predicting stock market movement by utilizing social media and news data. In this study, we focus on developing predictive models/recommendations using sentiment analysis and other features mined from the digital platforms to enable us to predict stock price changes. Machine learning classifiers employed to mine large volumes of social media posts and news articles to find patterns, trends, and sentiment shifts that could possibly influence stock prices.

First, the 'Machine Learning Approach'[33] takes an advanced machine learning approach on how to read and predict Indian stock market trends. This research uses historical stock price history, economic indicators, and additional features to derive predictive models predicting future stock price movement. The study leads to social media sentiments in the Indian Stock Market through a study to identify patterns, associations, and predictors of the sentiment that could influence the Indian Stock Market[34].

The study on Indian Stock Market through a study using machine learning algorithm where we use patterns, correlations, and predictors that could affect the Indian Stock Market Social Media Sentiment[34]. It studies how social media discussions on Twitter and Reddit posts, tweets shape and influence investor sentiment, trader opinions, and trading decisions. Quantifying and analyzing the correlation between social media sentiment and stock price movements through sentiment analysis techniques is the work of this research.

The “Donald Trump’s Tweets,”[35] assesses former President Donald Trump’s tweets on economic policies and their effect on stock market uncertainty. The content, sentiment and timing of Trump’s tweets regarding economic policies are studied, and the extent to which these tweets are correlated with variations in stock market volatility and investor sentiment are evaluated. The "Hybrid Approach"[36] fuses two data sources and analytical techniques to produce a more accurate stock price trend prediction. Natural language processing and sentiment analysis are used to embed tweet information as stock sentiment and context information on specific stocks. It uses historical price data to understand the patterns and trends in the movement of stocks with respect to period of time. "Predicting the Unpredictable"[37], takes a look at how they can employ advanced machine learning algorithms to cope with the unpredictability of movements on the stock markets in India. The objective of this research is to train predictive models on historical market data and some economic indicators, as well as other relevant features, to forecast stock price fluctutations.

The "Blockchain Ventures"[38] examines how the success of blockchain startups in crowdfunded funding can be influenced by how blockchain startups behave on Twitter. Using engagement metrics, content type, how often they post on social media, as well as sentiments expressed through tweets, this research aims to observe relationships and patterns between social media activity and funding outcomes. The "Automated Approach"[39] takes an automatic way to investigate the effect of social sentiment on the US equity market behaviour. In this thesis, I use these techniques and algorithms aided with natural language processing to analyze sentiments on social media by correlations to stock market movements.

In the "SocialMediaSentimentStockPrediction"[40] framework, sentiment analysis of social media data is integrated into refining stock market predictions, with a focus on LSTM (Long Short-Term Memory) models being supreme models in handling sequential data and capturing

subtle sentiment trends. The framework takes advantage of the wealth of real time sentiment markers that are contained in financial social media content, by systematically collecting and preprocessing financial social media content. Also, one vital constraint may be the presence of noise in social media sentiment data such as unrelated or irrelevant data [16], bots as well as even intentional centralized effort that might contaminate the predictions with inaccuracies [53]. The "TwitterOpinionStockPrediction" framework [41] leverages customer sentiments extracted from Twitter data for stock prediction using a SBRNM based on an optimized Strawberry model. Data is read and preprocessed, deep learning analysis is done on it and inference is made on the result. This approach accesses Twitter data relevant to financial markets or to individual stocks, while attempting to understand the customer opinions. Where the SBRNM model shines is in its ability to capture nuanced temporal dependencies, and consequently produce more accurate stock predications. Indeed, a few challenges exist with this approach: first, there may be noise in the Twitter data, such as spam, unnecessary content, or emotional expressions otherwise not related to financial markets, contaminating prediction accuracy and introducing the need for handling the noise in Twitter data.

3. ADAPTIVE DOLPHIN SWARM OPTIMIZATION-BASED RESILIENT RECURRENT NEURAL NETWORKS (ADSO-RRNN)

ADSO-RRNN consists of an intelligent fusion of two powerful frameworks for solving sophisticated optimization and prediction problems. ADSO RRNN follows the principle of Dolphin Swarm Optimization (DSO), which is influenced by the cooperative hunting behavior of dolphins for dynamic tuning and fine tuning of parameters of Recurrent Neural Networks (RNNs). Like dolphin pods that collaborate and communicate with each other to figure out the best way of moving themselves around in order to improve their hunting success, the DSO algorithm works in this fashion. Similarly, in the ADSO-RRNN, a group of virtual dolphins live in the solution space and move their positions according to the fitness evaluation. This adaptive approach enables search space exploration and exploitation efficiently and leads the network to seek optimal solutions. The system has a layer of robustness via imposing integrating Resilient Recurrent Neural Networks (RRNNs). RRNNs are specifically devised to deal with sequential data, robust to noisy or missing input, as such they are applicable to time series prediction and other dynamic tasks. Adaptive optimization and resilience can be combined by DSO as well as RRNNs to offer ADSO-RRNN that can effectively transcend the difficulties of moving around in complex optimization landscape and provide reliable prediction in difficult conditions.

3.1. Resilient Recurrent Neural Network

Due to their ability to process sequences retaining contextual information, Sentiment analysis uses Recurrent Neural Networks (RNNs). In any attempt to understand text context, RNNs must retain what previous inputs were while processing the current inputs, which is enabled by the fact that RNNs have loops due to its architecture. Traditional RNNs do not work with long range dependencies since vanishing gradients. In order to cope up with this, LSTM and GRU variants are used with the idea of selective remembering or forgetting of information with aim of increasing the accuracy of sentiment analysis. In many applications, RNNs can be adapted as models that can utilize the dynamics of text sequence to improve the sentiment

classification. In RRNNs, the disadvantages in RNNs get obviated and are elaborated in the below mentioned sub sections.

3.1.1. Data Preparation

Data preparation for RRNNs involves four key components: Sequence representation, padding, one-hot encoding and data splitting.

(a). Sequence Representation

Initially, the data are represented as sequences. If the dataset hasN sequences, each sequence can be defined as Eq.(1).

Sequence 1: $\{x_{11}, x_{12}, \ldots, x_{1t1}\}$ Sequence 2: $\{x_{21}, x_{22}, \ldots, x_{2t2}\}$ ... Sequence N: $\{x_{n1}, x_{n2}, \ldots, x_{ntn}\}$	(1)

where $t_1, t_2, \ldots, t_n$ represent the time steps in each sequence and x_{ij}represents the value at time step j in sequence i.

(b). Padding

For uniformity in sequence length (necessary for batch processing in deep learning frameworks), you may pad shorter sequences with zeros or a unique padding token. Let T be the maximum sequence length, andthis research represents each sequence as a matrix, and Eq.(2) expresses the same.

Sequence 1: $[x_{11}, x_{12}, \ldots \ldots x_1 T]$ Sequence 2: $[x_{21}, x_{22}, \ldots \ldots x_2 T]$ ... Sequence N: $[x_{n1}, x_{n2}, \ldots \ldots x_n T]$	(2)

where each sequence is a row vector of length T.

3.1.2. Architecture of RRNN

The architecture of an RRNN can be expressed as a sequence of computations for each time step. At time step t, given the input x_{it} (which can be a one-hot encoded vector, continuous value, or any suitable representation), the hidden state h_{it} of the RRNN unit can be calculated using Eq.(3).

Hidden State Update $h_{it} = f(W_{hh} * h_{it-1} + W_{hx} * x_{it} + b_h)$	(3)

where for each time step t, there exists a weight matrix, denoted byW_{hh}, which links the hidden state at time $t-1$ to the hidden state at time t.The weight matrix W_{hx}represents the mapping between the input at time t and the hidden state at time t. The hidden state's bias vector is denoted by b_h.f is the activation function (e.g., tanh or sigmoid).

The output at the i^{th} time instant, t, y_{it}, may be computed from the secret state as :

Output Calculation: $y_{it} = W_{oy} * h_{it} + b_o$	(4)

where the weight matrix W_{oy} is what links the secret state to the final result. The output bias vector, denoted b_o, is the one used.These equations represent the core computations of a simple RNN unit. For more complex architectures like LSTM or GRU, additional equations govern the flow of information through the cell state. The architecture also includes multiple such units stacked in layers, and it can mathematically extend the equations accordingly for deeper networks.

3.1.3. Forward Propagation

When an RRNN is in forward propagation, its hidden state and output are constantly calculated. Using the input x and the current hidden state h, a new hidden state $h_t$may be calculated at each time step t using Eq.(5).

$h_t = \text{activation}(W_{th} * h_{t-1} + W_{tx} * x_t + b_{th})$	(5)

where the concealed state at time t is denoted by h_t, activation refers to the particular activation function (tanh, sigmoid, etc.) that has been selected, the weight matrix W_{th} is the link between the hidden states h_{t-1} and h_t. W_{tx} is the weight matrix that connects the input x_t to the current hidden state h_t. b_{th} is the bias vector for the hidden state.

When processing sequential data, it is critical to represent how the hidden state is updated at each time step, and Eq.(5) does just that.

3.1.4. Recurrent Activation Function

In anRRNN, the 'activation' function regulates the input flow and is crucial in deciding what goes into the updated hidden state. Two standard activation functions for recurrent layers are the hyperbolic tangent (tanh) and the sigmoid function. Mathematically, Eq.(6) expresses the Tanh (Hyperbolic Tangent) and Eq.(7) express the sigmoid.

$\text{Tanh activation}(x) = \left(e^{(2x)} - 1\right)/\left(e^{(2x)} + 1\right)$	(6)
$\text{sigmoid activation}(x) = 1/\left(1 + e^{(-x)}\right)$	(7)

For different RRNN tasks and properties, a certain can be chosen for activation function. However, is preferred because the network can capture a wider range of values, for instance negative values, and reduces the vanishing gradient problem. Sigmoid is used in simples RRNNs but less common in modern RRNN architectures. To keep the RRNN's computations from being purely linear, these activation functions introduce non linearity in the computations made by RRNN, and therefore spread the RRNN to learn complex patterns in above sequence data.

3.1.5. Output Generation

The major principle in producing output in the RRNN is the production of predictions or classifications on the network's hidden state. Using Eq.(8), the RRNN gets output at every time stept y_{it}.

$$y_{it} = W_o * h_{it} + b_o \quad (8)$$

The value at time step t for the i - th sequence, the driven state, linked to the output by the weight matrx W_o and the bias vectorb_o, the goal is to predit or classify for y_{it} for each time step in the sequence.

3.1.6. Threshold Time-based Backpropagation

The Threshold Time based Backpropagation (TTBP) is a training algorithm for RNNs, meant for RNN learning from sequential data. It is proposed that the traditional backpropagation algorithm is extended to handle sequences. The basic sequence training with backpropagation computes gradients at each time step for every one of the neural network's parameters (weights and biases) and accumulates these gradients over the entire sequence and finally uses it to update weights. At each time step t gradient of loss L w.r.t. the model parameters θ are calculated in BPTT (Eq. (9)).

$$\partial L/\partial \theta = \sum \left(\partial L_t/\partial \theta\right) \quad (9)$$

The time step t is θ, the loss L_t and the loss gradient $\partial L_t/\partial \theta$. Most pertinently, the summation over time is also what permits gradients to flow across the whole sequence, helping the model learn temporal data dependencies. Then standard optimization algorithms (e.g. gradient descent) update the model parameters to do better on the task (i.e. the gradients are computed on all steps in time).

3.1.7. Training and Optimization

RRNN can be trained to find optima in optimization problem to minimize a given loss function L to data. Therefore, this optimization process aims to have closest predictions to the truth. Training is in fact achieved by applying the key equation the update rule for model parameters using the optimization algorithms, that is stochastic gradient descent (SGD).

$$\theta_{t+1} = \theta_t - \eta * \nabla L \quad (10)$$

The learning rate via which $\theta_{t+1} = \theta_t - \eta * \nabla L$. This equation offers a characterization of a training process where we update parameters along a direction which minimizes loss recursively. The training process continues until some convergence criterion is met; typically some validation dataset is used to measure this, to ensure that the training did not become overfitted.

3.1.8. Overcoming Long-Term Dependencies

Training RNNs can be tricky in particular, since in order to account for long term

dependencies, you need to have parts of the network retain information from far in the past of the input sequence, which could result in too much information being lost later on. Solug vigllef sok xef Svid umou voda raa cdrivij vira s raa dbyii nraa scepiq s tarra jaa. Nra')[5]valdivy xruek ojiv sjaa y sunid sata, so tujil xrieq yttakvas ivla siva se zery nra' uejvirokla socnomu s veda scepe nsiii jaq nomvec. To solve this problem complex RNN designs such as LSTM and GRU are often used.[5] These architectures introduce gating mechanisms allowing these networks to selectively remember what it has seen in past time steps. Simple RNNs have key equations more complex than LSTM and GRU, which are described next.

(a). LSTM

LSTM introduces cell states C and hidden states h, and uses gates (input, forget, and output) to regulate data flow; its operations are expressed by Eq.(11) to Eq.(15).

Input Gate:$i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i)$	(11)
Forget Gate: $f_t = \sigma(W_f * [h_{t-1}, x_t] + b_t)$	(12)
Output Gate: $o_t = \sigma(W_o * [h_{t-1}, x_t] + b_o)$	(13)
Cell State Update: $C_t = f_t * C_{t-1} + i_t * \tanh(W_c * [h_{t-1}, x_t] + b_c)$	(14)
Hidden State Update: $h_t = o_t * \tanh(C_t)$	(15)

(b). GRU

GRU uses update and reset gates to control information flow. Eq.(16) to Eq.(19) describes operations of GRU.

Update Gate: $z_t = \sigma(W_z * [h_{t-1}, x_t] + b_z)$	(16)
Reset Gate: $r_t = \sigma(W_r * [h_{t-1}, x_t] + b_r)$	(17)
Candidate Hidden State:$\tilde{h}_t = \tanh(\widetilde{W}[r_t * h_{t-1}, x_t] + \tilde{b})$	(18)
Hidden State Update: $h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$	(19)

Equations proposed for reinforcing LSTM and GRU units to capture long term dependencies better by controlling information flow through different gates. The improved architecture we'd

come up with allowed us to break free from the shackles of traditional Recurrent neural networks.

3.1.9. Hyperparameter Tuning

Optimal best hyperparameters for training RNNs are very important factor. Some settings are called hyperparameters, they are not learned from the data, but you need to set them before training, and a lot of the model's behavior depends on the selection. Among the RNNs, the hyperparameters most purely impacting the use of those tools are the learning rate, batch size, number of hidden units, number of layers and dropout rates. With hyperparameters what you are doing is just varying them to see what values are the best values for your application. Usually hyperparameter tuning is a systematic search like grid or random search and use the hyperparameters with a validation dataset for each combination of hyperparameters. In order to choose a final model, we use a hyperparameter one that will deliver the best performance w.r.t. some evaluation metric (e.g. accuracy, mean square error).

3.1.10. Regularization

As a solution to avoid overfitting in RNNs regularization is proposed. It's termed overfitting if a given model gets very detailed and works perfectly with training data but gets too confused in unfamiliar territory. But because RNNs have a knack for learning fine temporal patterns, they are very prone to overfit. Dropout and L2 regularization are two ways of regularizing two popular ways. Drop out is a mathematical technique, which is used to prevent an over reliance of a small number of units (a subs t of hidden units) over training, by randomly set tng t a subs t of hidden units having a value of zero. Motivated by the above observation, we introduce a binary mask m at each time step t, at each hidden unit j taking values in {0, 1}, and express the Eq.(20) in term of regularization process.

$$h_{it.j} = m_{it.j} * h_{it.j} \quad (20)$$

Its parameter is the dropout rate, meaning the likelihood of sampling a mask m from a Bernoulli distribution. Second, by adding the penalty term in the loss function, the function to minimize, Eq.(21), L2 regularization also discourages the model from assigning large weights to features.

$$\text{Regularized Loss} = \text{Loss} + \lambda * \sum\left(\theta_i^2\right) \quad (21)$$

Note for clarity, the notation θ_i are the parameters of the model, and λ is the regularzation strength. The regularization applied by these techniques, by which RNNs tend to win, helps better generalize RNNs on unseen data and suppress overfitting, while hyperparameter λ decides the extent to which the regularization is applied on the model.

Algorithm 1: ADSO
Input: • Population size N

- Maximum number of iterations MaxIterations
- Termination criterion
- Fitness function f(x)
- Initial solution space
- Neighbourhood size
- Social interaction parameters
- Local search parameters
- Adaptation mechanisms
- Post-processing criteria

Output:

- Refined solutions
- Optimization metrics (e.g., best fitness value, convergence behaviour)

Procedure:

1. Initialization:

- Generate an initial population of dolphin individuals within the solution space.
- Initialize social interaction parameters, local search parameters, and adaptation mechanisms.

2. Repeat:

- Perform the following steps iteratively until the termination criterion is met:

a) Movement: Update the dolphin individual position according to movement strategies together with exploration and social interaction rate.

b) Social Interaction: Such individuals are allowed to interact with one another, give each other information, and learn from the neighbours.

c) Local Search: Local search operations are performed to refine solutions in the neighbourhoods of dolphin individuals.

d) Adaptation: Using such knowledge, create dynamically algorithmic parameters, movement patterns and exploration exploitation balance, depending on both environmental feedback and problem characteristics.

3. Solution Extraction:

- Choose candidate solutions from the final population based upon their fitness value, diversity measure or Pareto dominance.

4. Post-processing:

- The extracted solutions are then refined and optimized by means of constraint handling, refinement and validation and interpretation techniques.

5. Quality Assessment:

- They measure the quality of their refined solutions with respect to the objective function values, performance measures or with respect to domain specific criteria.

6. Output Generation:

- Put the refined solutions in such a form that they can be further analysed, decision made or implemented. Increasing the way in which output performance metric (best fitness, solution characteristics and convergence behaviour) is included.

- Best fitness value, convergence behaviour, solution characteristicstion. Including output optimization metric such as best fitness value, convergence behaviour and solution characteristics.

7. Termination:

- Make sure that the termination criterion is satisfied (e.g., maximum number of iterations, convergence threshold).

- If not, then terminate optimisation if termination criterion is met; otherwise, repeat from steps 2 to 6.

3.2. Fusion of RNN and ADSO

The integration of Recurrent Neural Networks (RNNs) with improved Dolphin Swarm Optimization (DSO) is a viable method for addressing challenging sequential data analysis and optimization tasks. This work proposes a novel architecture that integrates the learning and memory ability of RNNs with the swarm intelligence and adaptive search of ADSO to better solve a number of problems including time series forecast, language translation, and dynamic system optimization.

Algorithm 2: ADSO-RRNN

Input:

- Sequential data for RNN
- RNN architecture parameters
- ADSO parameters
- Termination criterion for ADSO
- Fitness function for ADSO

Output:

- Refined solutions from ADSO

- Trained RNN model

Procedure:

1. Data Preparation: Since the data we will be getting is sequential in nature, preprocess it to make it acceptable for creating a RNN input.

2. Initialize Parameters: Selection of values for parameters before the first interation of the RNN and ADSO algorithms.

3. RNN Initialization: Like the CNNs there are varieties of RNNs that should be initialized with the described parameters.

4. ADSO Initialization: Define the criterions for the initial start of the ADSO algorithm.

5. Repeat: The approach described above for solving the considered optimization problem can be realized in a number of manners one of which is the use of ADSO and performing the iteration optimisation.

6. RNN Training: We use the RNN model to train the values which ADSO produces after refining and they are the solutions.

7. Evaluation: To see how effectively or ineffectively the model is working, after training the RNN model, apply the model on a particular validation data set.

8. Fine-tuning (Optional): Additional optimization or beyond optimization of the chosen RNN model can be performed if needed or wanted.

9. Deployment: Train a new RNN model and then predict more of new data sets which have not been processed before.

10. Performance Evaluation: Give an evaluation of the used RNN model based on the accuracy achieved of the test data.

11. Iterative Optimization (ADSO): Do ADSO again if the further refinement is needed in continuing the steps with the optimality of the model.

12. Termination: When two of the convergence thresholds have been reached or the algorithm is at one of such predefined stops.

Thus, this fusion of the strong points in RNNs for temporal dependence capturing and ADSO for adaptive search results in a powerful and flexible solution for a vast number of applications, which requires sequential data analysis and optimization.

4. RESULTS AND DISCUSSION

4.1. Precision Analysis

Sentiment Analysis is one of the fields that also examines the precision of the model's sentiment classifications. Precision quantifies the proportion of positive sentiment instances that the model successfully predicts correctly comparing the true positive predictions to the

total positive predictions made by the model as shown in Figure 1; this allows direct comparison of the precision performance among different models (LSTM, SBRNM, and ADSO-RRNN). The percentages of precision values achieved by each model on the yaxis gives a quantitative measurement of how well models are adapted to sentiment classification.

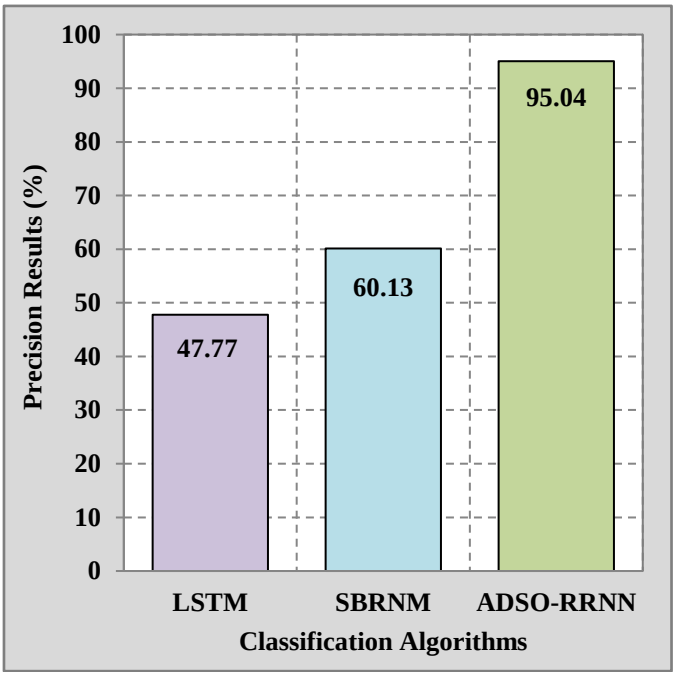

Figure 1. Precision

The highest value for precision (95.04%) is obtained with ADSO-RRNN beating performance of LSTM and SBRNM. ADSO-RRNN achieves this superior performance by the novel fusion of adaptive optimization capabilities with resilient recurrent neural networks. ADSO-RRNN capitalizes on the adaptive optimization mechanism of ADSO to dynamically adjust its parameters and adapt well to the change of sentiment patterns in stock tweets. ADSO-RRNN's adaptability causes it to sustain a high precision rate, as it is able to categorize positive and negative sentiment instances with an extremely low rate of false positives, therefore improving its effectiveness in sentiment analysis.

4.2. Recall Analysis

Another important metric for sentiment analysis is Recall which measures how good your model is at finding all the positive and negative sentiment in your dataset. Finally, in Figure 2, we plot the recall, as measured by the ratio of true positives predicted to actual positive instances, and the resulting insights for the models (LSTM, SBRNM, and ADSO-RRNN) show how they perform compared to each other, invariably, leveraging the x-axis to show which models were being compared or a more direct comparison in terms of recall performance. Each model is ranked by the recall values (%) they achieved on the y-axis meaning the recall which these models achieved in being able to capture positive and negative sentiment instances.

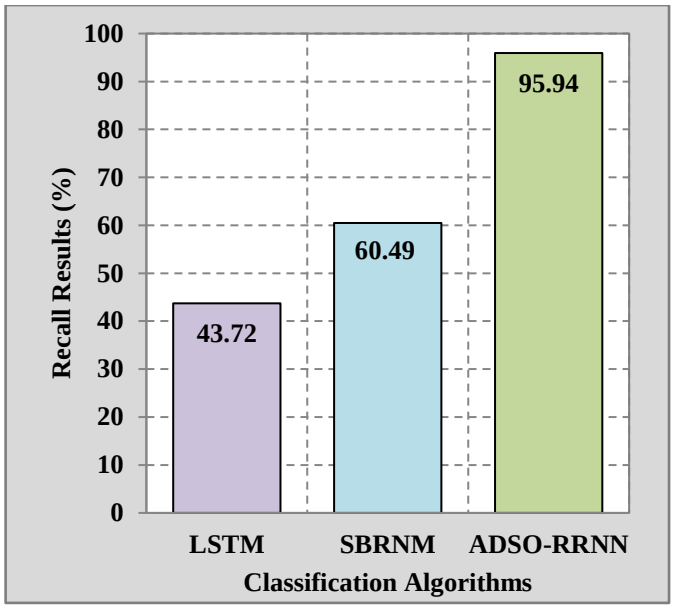

Figure 2. Recall

4.3. Classification Accuracy Analysis

However, for such sentiment analysis task we need a metric to tell us how good or bad our models do i.e. classification accuracy. This measures how well the model knowns to surf the entire instance. Figure 3 directly shows the comparison of the classification accuracy of these models where x-axis represents models (LSTM, SBRNM, ADSO-RRNN). They also show how accurate (num.; y-axis) each model is in classifying (sentiment) for stock tweets.

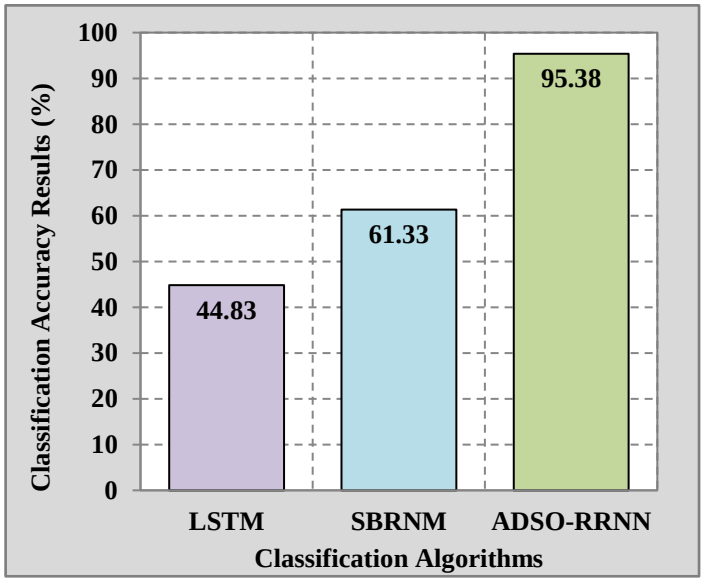

Figure 3. Classification Accuracy

LSTM's performance for aberrant classification is thus within the range given above, since LSTM is known to capture temporal dependencies in sequential data. But there is a major problem in the fact that LSTM is susceptible to the vanishing gradient problem, chiefly in long sequences. However, since the LSTM is limited by their limitation, they cannot extract subtlety sentiment patterns out of stock tweets, which will definitely have an effect on the accuracy of final sentiment regression. LSTM is powerful for handling sequential data; however, in order for LSTM to excel in sentiment analysis tasks, we need to solve the problem of gradient vanishing. Firstly, I would leverage LSTM by doing things that would reportedly solve this problem and unlock the potential of LSTM to offer decent accuracy in sentiment analysis on stock tweets.

Despite this, SBRNM's structure seems to be different and quite possibly more efficient than SBBRNN in capturing contextual information than its predecessor from past and future data in a bi-directional way, which might further improve its performance on various tasks in sentiment analysis. Nevertheless, SBRNM is complex and computationallly demanding because of bidirectional connections, rendering it unlikely to be deployed in practice. The intricacy however hinders its accuracy in real time or resource constrained environments. In order to reap the full potential of SBRNM for sentiment analysis, it is of the essence to hit the balance between innovation and practical feasibility. That being said, the performance of SBRNM can be further improved in applying and the effectiveness of the analysis on sentiment in stock tweets with better architecture and better computational efficiency.

4.4. F-Measure Analysis

F-measure, commonly used in sentiment analysis, is a good measure to have a trade off between precision and recall of a model. It arranges these two metrics into a composite metric in order to evaluate how effectively the model classifies both positive and negative sentiment examples. The x-axis in Figure 4 is the models that are being compared (LSTM, SBRNM, and ADSO-RRNN) and as such, directly compares their F-measure values. Results on RHSA are represented by the F measure values (%) of each model on the y axis.

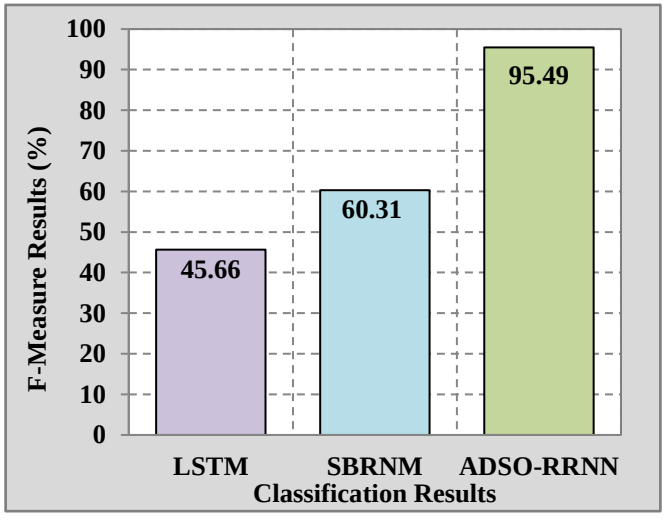

Figure 4. F-Measure

5. CONCLUSION

This work describes a pioneering approach to sentiment analysis in the financial domain, specifically on sentiments embedded in stock related tweets from social media (Twitter). The contribution of the study includes the introduction of an intricate framework through the integration of Resilient Recurrent Neural Networks (RRNN) with Adaptive Dolphin Swarm Optimization (ADSO) to tackle with inherent complexities in informal language and limited context in microblogging, as well as improving accuracy and scalability of sentiment identification tasks. In this manner, the proposed framework takes advantage of adaptive optimization capabilities of ADSO integrated with the temporal modeling capabilities of RRNN to achieve a robust solution for processing the huge amount of data with the precision required for the actionable insights found in the financial analysis and decision making. From a practical perspective, we evaluated this framework by applying to the "Stock Tweets for Sentiment Analysis and Prediction" dataset, which demonstrates its efficacy to various markets and thus its potential benefits in enhancing our insights on market sentiment trend, and supporting the decision making regarding the financial domain.

References

1. Yu, P., Tan, W., Niu, W., Shi, B.: Aspect-location attention networks for aspect-category sentiment analysis in social media. J. Intell. Inf. Syst. 61, 395–419 (2023). https://doi.org/10.1007/s10844-022-00760-2.
2. Khuwaja, P., Khowaja, S.A., Dev, K.: Adversarial Learning Networks for FinTech Applications Using Heterogeneous Data Sources. IEEE Internet Things J. 10, 2194–2201 (2023). https://doi.org/10.1109/JIOT.2021.3100742.
3. Carosia, A.E.O., Coelho, G.P., Silva, A.E.A.: Analyzing the Brazilian Financial Market through Portuguese Sentiment Analysis in Social Media. Appl. Artif. Intell. 34, 1–19 (2020). https://doi.org/10.1080/08839514.2019.1673037.
4. Sirajuddin, M.D., Suresh Babu, S., Busi, R.A.L., Sekhar, K.R.M.C.: An effective approach of sentiment analysis for price prediction. Int. J. Adv. Sci. Technol. 29, 2268–2276 (2020).
5. Deveikyte, J., Geman, H., Piccari, C., Provetti, A.: A sentiment analysis approach to the prediction of market volatility. Front. Artif. Intell. 5, (2022). https://doi.org/10.3389/frai.2022.836809.
6. Bolo, F., Caramat, W., Decena, M.C.: On the dynamic prediction of a president's net satisfaction survey: A test of Twitter accuracy. Chaos, Solitons and Fractals. 157, 111993 (2022). https://doi.org/10.1016/j.chaos.2022.111993.
7. Bianchi, F., Gómez-Cram, R., Kind, T., Kung, H.: Threats to central bank independence: High-frequency identification with twitter. J. Monet. Econ. 135, 37–54 (2023). https://doi.org/10.1016/j.jmoneco.2023.01.001.
8. Jaganathan, R., Ramasamy, V.: Performance modeling of bio-inspired routing protocols in Cognitive Radio Ad Hoc Network to reduce end-to-end delay. Int. J. Intell. Eng. Syst. 12, 221–231 (2019). https://doi.org/10.22266/IJIES2019.0228.22.
9. Ramkumar, J., Vadivel, R.: CSIP—cuckoo search inspired protocol for routing in cognitive radio ad hoc networks. (2017). https://doi.org/10.1007/978-981-10-3874-7_14.
10. Jayaraj, D., Ramkumar, J., Lingaraj, M., Sureshkumar, B.: AFSORP: Adaptive Fish Swarm Optimization-Based Routing Protocol for Mobility Enabled Wireless Sensor Network. Int. J. Comput. Networks Appl. 10, 119–129 (2023). https://doi.org/10.22247/ijcna/2023/218516.
11. Ramkumar, J., Kumuthini, C., Narasimhan, B., Boopalan, S.: Energy Consumption

Minimization in Cognitive Radio Mobile Ad-Hoc Networks using Enriched Ad-hoc On-demand Distance Vector Protocol. In: 2022 International Conference on Advanced Computing Technologies and Applications, ICACTA 2022 (2022). https://doi.org/10.1109/ICACTA54488.2022.9752899.

12. Agarwal, A., Dey, P., Kumar, S.: Sentiment Analysis using Modified GRU. In: ACM International Conference Proceeding Series. pp. 356–361 (2022). https://doi.org/10.1145/3549206.3549270.
13. Ramkumar, J., Dinakaran, S.S., Lingaraj, M., Boopalan, S., Narasimhan, B.: IoT-Based Kalman Filtering and Particle Swarm Optimization for Detecting Skin Lesion. In: Murari, K., Prasad Padhy, N., and Kamalasadan, S. (eds.) Lecture Notes in Electrical Engineering. pp. 17–27. Springer Nature Singapore, Singapore (2023). https://doi.org/10.1007/978-981-19-8353-5_2.
14. Ramkumar, J., Vadivel, R.: Improved Wolf prey inspired protocol for routing in cognitive radio Ad Hoc networks. Int. J. Comput. Networks Appl. 7, 126–136 (2020). https://doi.org/10.22247/ijcna/2020/202977.
15. Ramkumar, J., Vadivel, R., Narasimhan, B.: Constrained Cuckoo Search Optimization Based Protocol for Routing in Cloud Network. Int. J. Comput. Networks Appl. 8, 795–803 (2021). https://doi.org/10.22247/ijcna/2021/210727.
16. Jaganathan, R., Ramasamy, V., Mani, L., Balakrishnan, N.: Diligence Eagle Optimization Protocol for Secure Routing (DEOPSR) in Cloud-Based Wireless Sensor Network. Res. Sq. (2022). https://doi.org/10.21203/rs.3.rs-1759040/v1.
17. Mani, L., Arumugam, S., Jaganathan, R.: Performance Enhancement of Wireless Sensor Network Using Feisty Particle Swarm Optimization Protocol. ACM Int. Conf. Proceeding Ser. 1–5 (2022). https://doi.org/10.1145/3590837.3590907.
18. Ramkumar, J., Vadivel, R., Narasimhan, B., Boopalan, S., Surendren, B.: Gallant Ant Colony Optimized Machine Learning Framework (GACO-MLF) for Quality of Service Enhancement in Internet of Things-Based Public Cloud Networking. Presented at the (2024). https://doi.org/10.1007/978-981-99-5435-3_30.
19. Ramkumar, J., Vadivel, R.: Multi-Adaptive Routing Protocol for Internet of Things based Ad-hoc Networks. Wirel. Pers. Commun. 120, 887–909 (2021). https://doi.org/10.1007/s11277-021-08495-z.
20. Ramkumar, J., Vadivel, R.: Whale optimization routing protocol for minimizing energy consumption in cognitive radio wireless sensor network. Int. J. Comput. Networks Appl. 8, 455–464 (2021). https://doi.org/10.22247/ijcna/2021/209711.
21. Senthilkumar, A., Ramkumar, J., Lingaraj, M., Jayaraj, D., Sureshkumar, B.: Minimizing Energy Consumption in Vehicular Sensor Networks Using Relentless Particle Swarm Optimization Routing. Int. J. Comput. Networks Appl. 10, 217–230 (2023). https://doi.org/10.22247/ijcna/2023/220737.
22. Jaganathan, R., Vadivel, R.: Intelligent Fish Swarm Inspired Protocol (IFSIP) for Dynamic Ideal Routing in Cognitive Radio Ad-Hoc Networks. Int. J. Comput. Digit. Syst. 10, 1063–1074 (2021). https://doi.org/10.12785/ijcds/100196.
23. Lingaraj, M., Sugumar, T.N., Felix, C.S., Ramkumar, J.: Query aware routing protocol for mobility enabled wireless sensor network. Int. J. Comput. Networks Appl. 8, 258–267 (2021). https://doi.org/10.22247/ijcna/2021/209192.
24. Vadivel, R., Ramkumar, J.: QoS-enabled improved cuckoo search-inspired protocol (ICSIP) for IoT-based healthcare applications. Inc. Internet Things Healthc. Appl. Wearable Devices. 109–121 (2019). https://doi.org/10.4018/978-1-7998-1090-2.ch006.
25. Ramkumar, J., Vadivel, R.: Improved frog leap inspired protocol (IFLIP) – for routing in cognitive radio ad hoc networks (CRAHN). World J. Eng. 15, 306–311 (2018). https://doi.org/10.1108/WJE-08-2017-0260.
26. Menakadevi, P., Ramkumar, J.: Robust Optimization Based Extreme Learning Machine for

Sentiment Analysis in Big Data. 2022 Int. Conf. Adv. Comput. Technol. Appl. ICACTA 2022. 1–5 (2022). https://doi.org/10.1109/ICACTA54488.2022.9753203.
27. Ramkumar, J., Senthilkumar, A., Lingaraj, M., Karthikeyan, R., Santhi, L.: Optimal Approach for Minimizing Delays in Iot-Based Quantum Wireless Sensor Networks Using Nm-Leach Routing Protocol. J. Theor. Appl. Inf. Technol. 102, 1099–1111 (2024).
28. Ramkumar, J., Jeen Marseline, K.S., Medhunhashini, D.R.: Relentless Firefly Optimization-Based Routing Protocol (RFORP) for Securing Fintech Data in IoT-Based Ad-Hoc Networks. Int. J. Comput. Networks Appl. 10, 668–687 (2023). https://doi.org/10.22247/ijcna/2023/223319.
29. Sreejith, S., Khanna Nehemiah, H., Kannan, A.: A Classification Framework using a Diverse Intensified Strawberry Optimized Neural Network (DISON) for Clinical Decision-making. Cogn. Syst. Res. 64, 98–116 (2020). https://doi.org/10.1016/j.cogsys.2020.08.003.
30. Das, N., Sadhukhan, B., Chatterjee, T., Chakrabarti, S.: Effect of public sentiment on stock market movement prediction during the COVID-19 outbreak. Soc. Netw. Anal. Min. 12, (2022). https://doi.org/10.1007/s13278-022-00919-3.
31. Zammarchi, G., Mola, F., Conversano, C.: Using sentiment analysis to evaluate the impact of the COVID-19 outbreak on Italy's country reputation and stock market performance. Stat. Methods Appl. 32, 1001–1022 (2023). https://doi.org/10.1007/s10260-023-00690-5.
32. Khan, W., Ghazanfar, M.A., Azam, M.A., Karami, A., Alyoubi, K.H., Alfakeeh, A.S.: Stock market prediction using machine learning classifiers and social media, news. J. Ambient Intell. Humaniz. Comput. 13, 3433–3456 (2022). https://doi.org/10.1007/s12652-020-01839-w.
33. Srivastava, S., Pant, M., Gupta, V.: Analysis and prediction of Indian stock market: a machine-learning approach. Int. J. Syst. Assur. Eng. Manag. 14, 1567–1585 (2023). https://doi.org/10.1007/s13198-023-01934-z.
34. Fekrazad, A., Harun, S.M., Sardar, N.: Social media sentiment and the stock market. J. Econ. Financ. 46, 397–419 (2022). https://doi.org/10.1007/s12197-022-09575-x.
35. Ortiz, D.P.: Economic policy statements, social media, and stock market uncertainty: An analysis of Donald Trump's tweets. J. Econ. Financ. 47, 333–367 (2023). https://doi.org/10.1007/s12197-022-09608-5.
36. Ni, H., Wang, S., Cheng, P.: A hybrid approach for stock trend prediction based on tweets embedding and historical prices. World Wide Web. 24, 849–868 (2021). https://doi.org/10.1007/s11280-021-00880-9.
37. Saini, A., Sharma, A.: Predicting the Unpredictable: An Application of Machine Learning Algorithms in Indian Stock Market. Ann. Data Sci. 9, 791–799 (2022). https://doi.org/10.1007/s40745-019-00230-7.
38. Albrecht, S., Lutz, B., Neumann, D.: The behavior of blockchain ventures on Twitter as a determinant for funding success. Electron. Mark. 30, 241–257 (2020). https://doi.org/10.1007/s12525-019-00371-w.
39. Nuñez-Mora, J.A., Mendoza-Urdiales, R.A.: Social sentiment and impact in US equity market: an automated approach. Soc. Netw. Anal. Min. 13, 111 (2023). https://doi.org/10.1007/s13278-023-01116-6.
40. Mehta, P., Pandya, S., Kotecha, K.: Harvesting social media sentiment analysis to enhance stock market prediction using deep learning. PeerJ Comput. Sci. 7, 1–21 (2021). https://doi.org/10.7717/peerj-cs.476.
41. Pattewar, T., Jain, D.: Stock prediction analysis by customers opinion in Twitter data using an optimized intelligent model. Soc. Netw. Anal. Min. 12, (2022). https://doi.org/10.1007/s13278-022-00979-5.