IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 14, No. 4, August 2025, pp. 2753~2764
ISSN: 2252-8938, DOI: 10.11591/ijai.v14.i4.pp2753-2764

Optimizing long short-term memory hyperparameter for cryptocurrency sentiment analysis with swarm intelligence algorithms

Kristian Ekachandra, Dinar Ajeng Kristiyanti
Information Systems Study Program, Faculty of Engineering and Informatics, Universitas Multimedia Nusantara, Tangerang, Indonesia

Article Info

Article history:

Received Jun 18, 2024
Revised Feb 26, 2025
Accepted Mar 15, 2025

Keywords:

Cryptocurrency
Feature selection
Long short-term memory networks
Sentiment analysis
Swarm intelligence

ABSTRACT

This study investigates the application of swarm intelligence algorithms, specifically particle swarm optimization (PSO), ant colony optimization (ACO), and cat swarm optimization (CSO), to optimize long short-term memory (LSTM) networks for sentiment analysis in the context of cryptocurrency. By leveraging these optimization techniques, we aimed to enhance both the accuracy and computational efficiency of LSTM models by fine-tuning critical hyperparameters, notably the number of LSTM units. The study involved a comparative analysis of LSTM models optimized with each algorithm, evaluating performance metrics such as accuracy, loss, and execution time. Results indicate that the PSO-LSTM model achieved the highest accuracy at 86.08% and the lowest loss at 0.57, with a reduced execution time of 58.43 seconds, outperforming both ACO-LSTM and CSO-LSTM configurations. These findings underscore the effectiveness of PSO in tuning LSTM parameters and emphasize the potential of swarm intelligence for enhancing neural network performance in real-time sentiment analysis applications. This research contributes to advancing optimized deep learning techniques in high dimensional data environments, with implications for improving cryptocurrency sentiment predictions.

Corresponding Author:

Dinar Ajeng Kristiyanti
Information Systems Study Program, Faculty of Engineering and Informatics
Universitas Multimedia Nusantara
Tangerang, Indonesia
Email: dinar.kristiyanti@umn.ac.id

1. INTRODUCTION

The global finance sector has increasingly turned its attention to cryptocurrencies, a class of digital assets known for rapid value fluctuations and decentralized control mechanisms. Cryptocurrencies have gained widespread popularity as highly volatile digital assets that offer both high risk and potential reward for investors. The rapid and often unpredictable fluctuations in cryptocurrency prices are influenced by various factors, including market sentiment, technological advancements, regulatory changes, and influential social media activities. Bitcoin, the first cryptocurrency, saw an extraordinary rise in value in 2017, increasing by over 2,000% to reach $20,000 [1]. Cryptocurrencies like bitcoin offer secure, direct transactions without intermediaries, facilitated by blockchain technology [2]. However, they present unique challenges, such as high energy consumption due to mining activities, and have been associated with illicit activities, which have led to regulatory responses from various governments [3], [4]. These challenges contribute to cryptocurrency market volatility, as regulatory news and advancements in cryptocurrency infrastructure (e.g., proof of stake implementations and exchange-traded fund (ETF) approvals) continue to influence price movements [5]–[8]. With platforms like Twitter/X playing a significant role in shaping public opinion, sentiment analysis has emerged as a valuable tool for assessing investor sentiment and predicting cryptocurrency trends [9]. However, conducting accurate sentiment analysis in this field poses challenges due to the high dimensionality and noisy nature of social media data, which necessitates advanced machine learning models capable of handling such complexity [10], [11].

Sentiment analysis on social media data is typically conducted using three approaches: lexicon-based, machine learning-based, and hybrid methods. Lexicon-based methods are suitable for unsupervised data, while machine learning approaches require labeled datasets [12]. Hybrid approaches that integrate lexicons with machine learning have demonstrated improved accuracy, with studies reporting up to 10% gains over conventional methods [13]. Recent research highlights the effectiveness of deep learning algorithms, particularly long short-term memory (LSTM) networks, which can capture temporal dependencies and complex patterns in sequential data like social media posts [14], [15]. Studies have shown that LSTM outperforms traditional machine learning models in sentiment analysis, making it a suitable choice for analyzing high dimensional and noisy data [11], [16], [17].

Despite LSTM's advantages, improving its performance for sentiment analysis on large social media datasets requires feature selection techniques to manage data dimensionality and reduce noise. However, the effectiveness of LSTM models is highly dependent on their hyperparameters, such as the number of LSTM units. Traditional methods of hyperparameter tuning can be time-consuming and may not always yield optimal configurations, especially in high dimensional sentiment analysis tasks. In recent years, swarm intelligence algorithms, such as particle swarm optimization (PSO), ant colony optimization (ACO), and cat swarm optimization (CSO), have been utilized to improve optimization processes across various domains due to their ability to efficiently explore large search spaces [18], [19]. Prior studies have used swarm intelligence algorithms to improve accuracy in machine learning applications, with PSO increasing accuracy for SVM models from 78.70% to 86.20% [20], and adaptive particle swarm optimization (APSO) improving LSTM accuracy from 95.1% to 97.8% in sentiment classification [21]. Studies comparing PSO and CSO have shown that CSO can deliver even better accuracy and faster processing times in sentiment analysis [11]. However, research integrating these algorithms specifically with LSTM for cryptocurrency sentiment analysis remains limited, presenting a gap that this study aims to fill. This study aims to address the limitations of traditional LSTM tuning methods by employing PSO, ACO, and CSO algorithms to optimize LSTM networks specifically for cryptocurrency sentiment analysis. By fine-tuning the LSTM units and other key hyperparameters, we aim to enhance the model's accuracy, reduce processing time, and improve overall performance.

The remainder of this paper is organized as follows: section 2 presents the methodology, detailing the data pre-processing steps, the swarm intelligence-based optimization techniques applied, and optimizing LSTM using hyperparameter tuning. Section 3 discusses the experimental results, including a best model performance analysis, confusion matrix and classification metrics, discussion, and limitations and implications for future research. Finally, section 4 concludes with a summary of the findings and potential applications in cryptocurrency market analysis.

2. METHOD

The methodology of this study, as illustrated in Figure 1, consists of several key stages to prepare and optimize an LSTM model for cryptocurrency sentiment analysis. Each stage, from data preprocessing to model evaluation, is outlined to facilitate understanding and reproducibility. This structured approach ensures that each component of the model development process is systematically addressed, leading to more reliable and insightful sentiment predictions.

2.1. Data collection

Data for this study was collected from Twitter/X using the tweet-harvest tool, configured with parameters such as twitter_auth_token, search_keyword, and limit to streamline data extraction. The keywords included terms like "cryptocurrency," "crypto," and "bitcoin," as well as related terms that capture public sentiment on cryptocurrencies. The data collection was limited to 1,000 tweets per day to ensure a manageable dataset with a broad range of opinions. Over the data collection period, from December 31, 2023, to January 31, 2024, a total of 9,884 tweets were successfully gathered, providing a robust dataset for sentiment analysis. This period was specifically chosen to capture discussions around the U.S. SEC’s BTC ETF approval on January 10, 2024, an event anticipated to significantly influence cryptocurrency sentiment and public discussion [22].

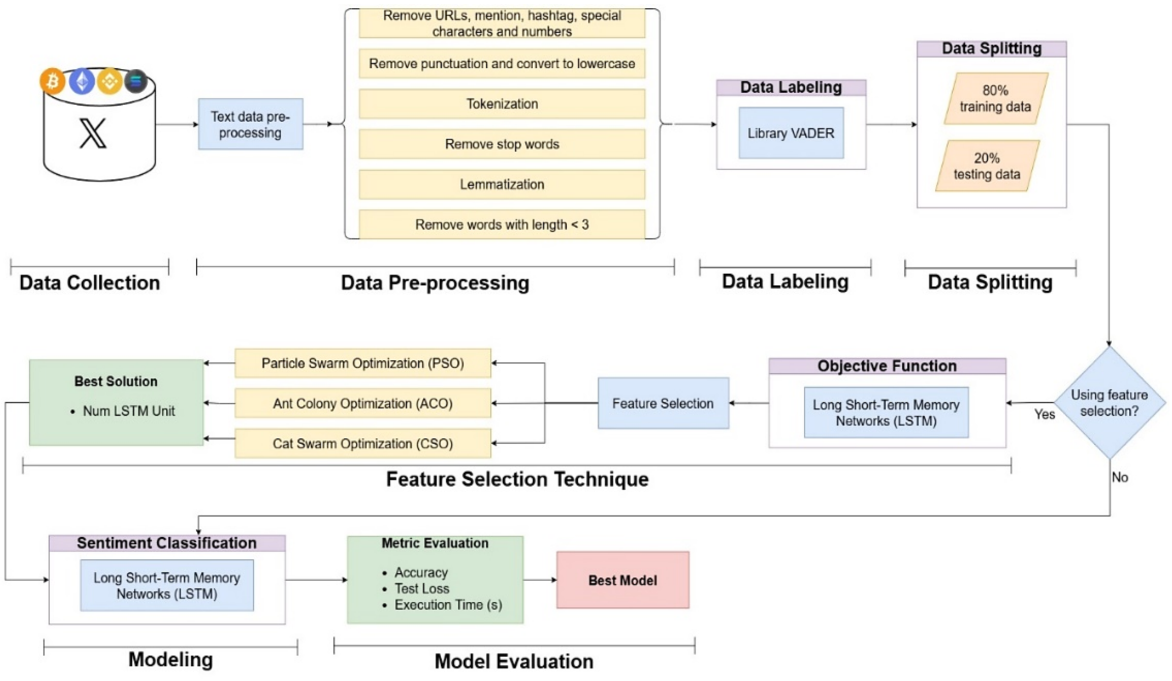

Figure 1. Proposed architecture

2.2. Data pre-processing

The collected Twitter/X data required extensive pre-processing to ensure consistency and reduce noise, thereby improving the accuracy of the sentiment analysis model. Each stage in this process is outlined in following subsection. This process includes data cleaning, text normalization, and removal of irrelevant elements to produce higher quality input for the model.

2.2.1. Remove URLs, mentions, hastags, special characters, and number

The first step involved removing URLs, mentions, hashtags, special characters, and numbers from the tweets. URLs and mentions (e.g., https://example.com, @username) often contain non-sentiment-bearing information, while hashtags and special characters add noise to the data. Numbers were also removed to focus the analysis on textual content relevant to sentiment. For example, the tweet “Check this out! https://crypto.com @crypto #bitcoin123” would be reduced to “Check this out bitcoin” [23].

2.2.2. Remove punctuation and convert to lowercase

All punctuation was removed, and the text was converted to lowercase. This step ensures that similar words with different cases (e.g., “Bitcoin” and “bitcoin”) are treated uniformly, reducing variability and vocabulary size. For instance, the text “Bitcoin, the future!” becomes “bitcoin the future,” ensuring consistency across the dataset [23], [24].

2.2.3. Tokenization

Tokenization was applied to split each tweet into individual words or tokens, making it easier for the model to analyze textual data more effectively. This process involves breaking down sentences into smaller components, such as words or symbols, which allows machine learning algorithms to handle and interpret the text systematically. For instance, the sentence “Bitcoin is the future” would be tokenized into [“bitcoin”, “is”, “the”, “future”], enabling the model to evaluate each token separately and identify patterns or sentiments associated with individual words [25].

2.2.4. Remove stop words

Commonly used words that do not contribute to sentiment, known as stop words (e.g., "is," "and," "the"), were removed. This step reduces data complexity by allowing the model to focus on sentiment-relevant words. After stop word removal, the phrase “Bitcoin is the future” would retain only ["bitcoin", "future"], highlighting the core sentiment-bearing words [23].

2.2.5. Lemmatization

Lemmatization was applied to convert words into their base forms, ensuring that variations of the same word are treated as one. For instance, "rising" and "rose" were both converted to "rise." This step improves the model's ability to recognize different forms of the same word, thereby enhancing consistency and reducing dimensionality in the data [23].

2.2.6. Remove words with length < 3

Lastly, words with fewer than three characters were removed, as they typically provide limited sentiment information and can add noise. Short words such as "it" and "an" were excluded to streamline the dataset further and focus the analysis on meaningful terms. This filtering improves the model's efficiency by reducing unnecessary data elements [26].

2.3. Data labeling

Data was labeled using the valence aware dictionary and sentiment reasoner (VADER), a sentiment analysis tool that performs well on social media data by accounting for emoticons, abbreviations, and other informal language features commonly used on Twitter/X. VADER assigns sentiment scores that categorize each tweet as positive, negative, or neutral [26]. For this study, only positive and negative sentiments were retained, as these are most indicative of the buy-or-sell decisions in cryptocurrency trading, while neutral sentiments were excluded to maintain a focus on directional sentiment that impacts trading behavior [27], [28]. Table 1 represents a sample of the labeled data using the VADER on top 5 data that have undergone data pre-processing.

Table 1. Labeled data using VADER

Processed text	VADER sentiment
BTC honestly dont think matter target get past ath drop back still bullish close	Positive
official happy new year crypto community praying may year bring green candle amp every coin	Positive
financial market analyst cryptocurrency blockchain amp web researcher	Negative
money broken low interest rate fake money lead people treat real estate investment entered chat	Negative
absolutely outrageous bankmanfrieds pac essentially paid francis conoles democratic primary campaign stolen crypto fund conole eked narrow win fix one held accountable	Negative

2.4. Data splitting

The labeled dataset was split into training and testing sets, with 80% of the data allocated for training and 20% for testing. This split enables the evaluation of model generalizability. Referring to study that compared the ratios of training and testing data in sentiment analysis for cryptocurrencies using tweet data, the ratio of 80:20 yielded the best performance compared to ratios of 90:10 and 70:30 [29]. The training data is used to train the classification model for both the objective function and sentiment classification, while the testing data is used to validate the trained model.

2.5. Feature selection using swarm intelligence algorithms

The feature selection stage in sentiment analysis is used to choose relevant features because data extracted from social media generally has high dimensional characteristics. Swarm intelligence is one of the components of feature selection that utilizes the hybrid method. Due to its suitability for the research objective, this study employs the hybrid method of feature selection based on swarm intelligence to optimize the LSTM model. Swarm intelligence algorithms PSO, ACO, and CSO were employed to optimize the number of LSTM units. Each algorithm aimed to identify an ideal configuration that maximizes accuracy while reducing computational time. The hyperparameter tuning process focused on the LSTM units, as this parameter critically impacts the model's ability to capture sequential patterns in sentiment data. However, to get the LSTM units we must use the objective function. The objective function is designed to train the LSTM model intelligently, and it has been proven to effectively adjust the weights in the LSTM model, minimizing loss and facilitating efficient model learning [30].

2.5.1. Particle swarm optimization

Optimization technique inspired by the social behavior of birds or fish when searching for food. Birds do not know the exact location of food, so they generally follow other birds considered close to the food source [31]. In PSO, each bird is referred to as a particle, and each particle has a fitness function (square of error). A group of particles is known as a swarm. PSO algorithm is shown in Pseudocode 1. The working principle of PSO is as follows: in each iteration, the algorithm first finds the best solution found within the swarm, which is stored as personal best (pbest). Then, global best (gbest) is updated to be the best solution found across all iterations. The discovery of pbest and gbest is determined by (1) [31].

$$\begin{cases} \vec{v}_i \Leftarrow \vec{v}_i + \vec{U}(0,\phi_1) \otimes (\vec{p}_i - \vec{x}_i) + \vec{U}(0,\phi_2) \otimes \left(\vec{p}_g - \vec{x}_i\right) \\ \vec{x}_i \Leftarrow \vec{x}i + \vec{v}_i \end{cases} \quad (1)$$

Where $\vec{v}_i$ represents the particle's velocity, $\vec{x}_i$ represents the particle's position, $\vec{U}(0,\phi_\mathrm{i})$ signifies a sequence of uniformly distributed random numbers between 0 and ϕ, in freshly generated for each particle at every iteration, $\otimes$ denotes the operation of multiplying elements correspondingly [32]. To ensure balance, each velocity vector component $\vec{v}_i$ is confined within the minimum and maximum velocity thresholds, donoted as $[-V_{max}, +V_{max}]$ [33].

Pseudocode 1. Particle swarm optimization algorithm [33]

```
1) Set up an array of particles with random coordinates and speeds across 'D' dimensions.
2) Begin iteration.
3) For each particle within the iteration, determine the value of the targeted optimization function in 'D' variables.
4) Assess the fitness of the particle and compare it with its best recorded position (pbest). If the new evaluation is superior, update pbest to this newer measurement and record the particle's current coordinates as its best spot within the 'D' dimensional grid.
5) Recognize the most successful particle in the vicinity and assign the index of this particle to a variable $g$.
6) Modify each particle's motion and location using (1), which incorporates the best positions identified by the individual particle and its neighbors.
7) Persist with the iteration until a certain-requirements is fulfilled, which could be an acceptable level of fitness or a ceiling on iteration counts.
8) Terminate the iteration loop.
```

In the context of feature selection, the PSO algorithm is designed to find an optimal subset of features that improves the model's performance by reducing the data's dimensionality while maintaining or enhancing classification accuracy [34].

2.5.2. Ant colony optimization

Optimization technique inspired by the foraging behavior of ants. As ants search for food, they leave pheromone trails that serve as a route to guide them back to the nest [31]. The number of ants travelling through that path influences the density of pheromone deposition and evaporation. The quality and quantity of food brought by the ants also affect pheromone deposition. Therefore, the ants can identify the optimal path by following the trail with the maximum pheromone density. The discovery of the optimal path (2) and the update of the pheromone (3) by the ants are determined by the following equations [31].

$$P\left(\frac{c_i}{s}\right) = \frac{[\tau_i]^{\alpha} \cdot [n(c_i)]^{\beta}}{\Sigma cj \in N(s) [\tau_j]^{\alpha} [n(c_j)]^{\beta}} \qquad \forall c_i \in N(s) \quad (2)$$

$$\tau_i \leftarrow (1-\rho)\tau_i + \rho \cdot \Sigma \left\{ s \leftarrow \frac{s_{upd}}{C_i E s} \right\} w_s \cdot F(s) \quad (3)$$

Here is the explanation of pheromone deposition τ_i in which it represents the pheromone deposition at the ith node. n is an optional weighing function, c_j represents each feasible solution, α and β are positive parameter. On the other hand, pheromone updation s_{upd} is the solution used for pheromone update. w_s is the weight of solution s, ρ is the evaporation constant, $F(s)$ is the quality function. Based on these equations, Pseudocode 2 is the ACO algorithm.

Pseudocode 2. Ant colony optimization algorithm [35]

```
1) Initialize pheromone trails
2) While (termination criteria not met) do
3) For each ant
4) Build a solution path based on pheromone trails and heuristic information (2)
5) Calculate the fitness of the solution
6) Update the local pheromone trail (3)
7) End iteration
```

8) Update the global pheromone trail based on the best solution found
9) End

In feature selection, ACO is used to find a subset of features that yields the best performance for the model by mimicking how ants find the shortest path from the nest to a food source [19]. Virtual ants iterate through the features, constructing solutions by selecting features based on probabilities influenced by pheromone levels. This increases the likelihood of selecting features that contribute positively to improved classification accuracy [36].

2.5.3. Cat swarm optimization

Optimization technique inspired by the behavior of cats, utilizing two specific behaviors known as the seeking mode and tracing mode [11]. Each cat has a position consisting of M dimensions in the search space, where each dimension has its velocity. The fitness value represents how well a set of solutions (cats) performs. Additionally, there is a flag used to classify cats into seeking mode or tracing mode. The working principle of CSO involves determining the number of cats involved in each iteration and running them through the algorithm. The best cat in each iteration is stored in memory, and the cat in the final iteration represents the final solution. The CSO algorithm aims to find the optimal solution in the search space by utilizing the seeking and tracing behaviors inspired by cats. In the seeking mode, cats randomly explore or observe their surroundings to find better positions. On the other hand, in the tracing mode, cats move towards the target with a mathematically calculated velocity. The tracing mode CSO algorithm is expressed in (4) and (5) [37].

$$xjd_{new} = (1 + rand \cdot SRD) \cdot xjd_{old} \tag{4}$$

$$Pi = \frac{|FS_i - FS_b|}{FS_{max} - FS_{min}}, where\ 0 < i < j \tag{5}$$

The seeking mode of CSO mimics the resting behavior of cats. There are four important parameters in this mode: seeking memory pool (SMP), seeking range of the selected dimension (SRD), count of dimensions to change (CDC), and self-position considering (SPC), which are manually set values. In each iteration of CSO, randomly select CDC dimensions to be mutated. Add or subtract a random value within SRD from the current value, replacing the old position xjd_{old} with the new position xjd_{new}, as shown in (4). Here, xjd_{new} represents the next position, j denotes the cat index, d means the dimension, and $rand$ is a random number in the interval between 0 and 1. Based on probabilities, select one of the candidate points to be the following position for the cat. The candidate points with a higher fitness value are more likely to be chosen, as shown in (5). However, if all fitness values are equal, set the probability of selecting each candidate point to 1. If the goal is minimization, set $FS_b = FS_{max}$ otherwise, if the goal is maximization, specify FS_b=FS_{min}. Pseudocode 3 is the CSO algorithm in seeking mode. The seeking mode CSO algorithm is expressed in (6) and (7) [37]:

$$V_{k,d} = V_{k,d} + r_1 c_1 (x_{best,d} - x_{k,d}), where\ d = 1,2,\dots,M \tag{6}$$

$$V_{k,d} = V_{k,d} + V_{k,d} \tag{7}$$

Pseudocode 3. Cat swarm optimization algorithm in seeking mode [37]

```
1) Create $j$ instances of the current position of the cat, where $j = SMP$. If SPC is a true condition, let
   $j = (SMP - 1)$, maintain the current position as an option among the possible candidates.
2) For each instance, according to CDC, randomly increase or decrease SRD percents of the existing values
   and replace the former ones.
3) Calculate the fitness values of all candidate points.
4) If the case where not all fitness value are identical, calculate the selecting probability of each candidate
   point by (5) otherwise set all the selecting probability of each candidate point be 1.
5) Randomly pick the point to move to from the candidate points, and replace the position of the cat $k$.
```

In the first iteration of the tracing mode, the velocity values are randomly assigned for all dimensions of the cat's position. However, the velocity values need to be updated for each dimension according to (6) for the subsequent steps. If the velocity exceeds the maximum allowed value, it is set to the maximum velocity. Update the cat's position based on (7). Pseudocode 4 is the CSO algorithm in tracing mode referring to (6) and (7).

Pseudocode 4. Cat swarm optimization algorithm in tracing mode [38]

```
1) Update the velocities for every dimension $V_{k,d}$ as shown in (6).
2) Verify that the velocities are within the maximum allowed velocities range. Adjust any velocity exceeding this range back to the maximum limit.
3) Adjust the location of the cat $k$, according to (7).
```

Based on two modes in the CSO algorithm, namely, seeking mode and tracing mode. Pseudocode 5 is the combination of the two modes.

Pseudocode 5. Cat swarm optimization algorithm [38]

```
1) Initialize by creating ‘N’ cats in the algorithm
2) Place the cats randomly within an ‘M’-dimensional search area, assigning velocities that are within the predefined maximum bounds. Randomly determine a number of cats to engage in tracing mode based on the Mixing Ratio (MR), positioning the rest in seeking mode.
3) Compute the fitness for each cat using the fitness function which measures their proximity to the objective, and memorize the location of the most optimal cat $xbest$.
4) Relocate the cats based on their assigned modes: those in seeking mode undergo a different process, while those in tracing mode adjust their velocity and position according to specific formulas.
5) Selectively switch a number of the cats back to tracing mode as per the MR, and the remainder continue in seeking mode.
6) Check if the end conditions of the algorithm have been met; if so, stop the algorithm, otherwise cycle through steps 3 to 5 again.
```

In the context of feature selection, the CSO algorithm operated by exploring the feature space to discover the optimal feature combinations. Through iterations between the seeking and tracing modes, CSO adaptively explored and utilized information from the feature space to identify feature subsets that provided the best performance for the used model. In this process, the algorithm aimed to balance exploration (searching for different feature combinations) and exploitation (following promising positions) to obtain an optimal solution.

2.6. Optimizing long short-term memory using hyperparameter tuning

The hyperparameter tuning process was specifically directed towards optimizing the LSTM model to improve its performance in sentiment analysis. The primary hyperparameter adjusted in this study was the number of LSTM units, which determines the dimensionality of the cell state within the model and directly influences its ability to capture sequential dependencies in the data. An LSTM model was configured to classify the sentiment of the preprocessed cryptocurrency-related data. The optimized model configurations determined by PSO, ACO, and CSO were compared against a baseline LSTM without optimization. Evaluation metrics included accuracy, loss, and execution time, which provide insights into the model’s effectiveness and efficiency. The LSTM model’s performance was evaluated using accuracy, loss, and execution time. Accuracy measures the percentage of correct predictions; loss indicates model convergence and execution time assesses computational efficiency. Each metric was compared across the PSO-LSTM, ACO-LSTM, CSO-LSTM, and baseline LSTM models to determine the most effective optimization technique. The optimal configuration of the model was determined through hyperparameter tuning, as summarized in Table 2.

Table 2. Hyperparameter tuning

Algorithm	Parameters	Values
LSTM	Embedding input dimension	7818
	Embedding output dimension	300
	Embedding input length	25
	LSTM unit	256 (optimized by each swarm intelligence algorithms)
	LSTM dropout	0.2
	LSTM recurrent dropout	0.2
	Dense classes	2
	Dense activation	sigmoid
	Optimizer	Adam
PSO, ACO, CSO	n_particle, n_ants, n_cats	15
	num_iterations	50
	lb; ub	16; 256

3. RESULTS AND DISCUSSION

3.1. Experimental results

Table 3 presents a comparison of each LSTM model configuration, including a standard LSTM model without optimization and LSTM models optimized using PSO, ACO, and CSO algorithms. Each configuration baseline, PSO-LSTM, ACO-LSTM, and CSO-LSTM was evaluated based on accuracy, loss, and execution time. The results of this tuning process, guided by each swarm intelligence algorithm, demonstrated that adjusting the LSTM units using swarm-based optimization significantly improved model performance. Each algorithm provided a unique perspective on optimal LSTM unit selection, reflecting the strengths of swarm intelligence in hyperparameter optimization.

Table 3. Comparison of sentiment classification models

LSTM optimizer	Num LSTM unit	Loss	Accuracy	Execution time (s)
-	256	0.930019	0.853225	139.465759
PSO	16	0.570487	0.860843	58.430918
ACO	16	0.602706	0.853225	56.374625
CSO	29	0.662189	0.859319	65.443925

As shown in Table 3, the PSO-optimized LSTM achieved the highest accuracy at 86.08% with the lowest loss value of 0.570487 and a relatively low execution time of 58.43 seconds. Compared to the baseline LSTM model without optimization, which has a loss value of 0.930019 and an accuracy of 85.32%, the PSO-LSTM demonstrates improved performance, especially in terms of accuracy and efficiency. The PSO optimizer effectively identified a configuration with only 16 LSTM units, thereby balancing model performance and execution time. The ACO and CSO algorithms also achieved comparable results, though their accuracy and loss values were slightly lower than those of PSO. Nevertheless, both ACO and CSO significantly reduced execution time relative to the non-optimized model, demonstrating the effectiveness of swarm intelligence algorithms in accelerating the training process.

3.2. Model performance analysis

The performance of the PSO-LSTM model during training and validation is shown in Figure 2. The left plot illustrates the model accuracy across epochs, with the training accuracy increasing to nearly 100% by the end of the training epochs. The validation accuracy stabilizes around 85%, indicating a potential issue with overfitting. This result suggests that the model might be learning features specific to the training data that do not generalize well to unseen data, which could impact its effectiveness in real-world applications.

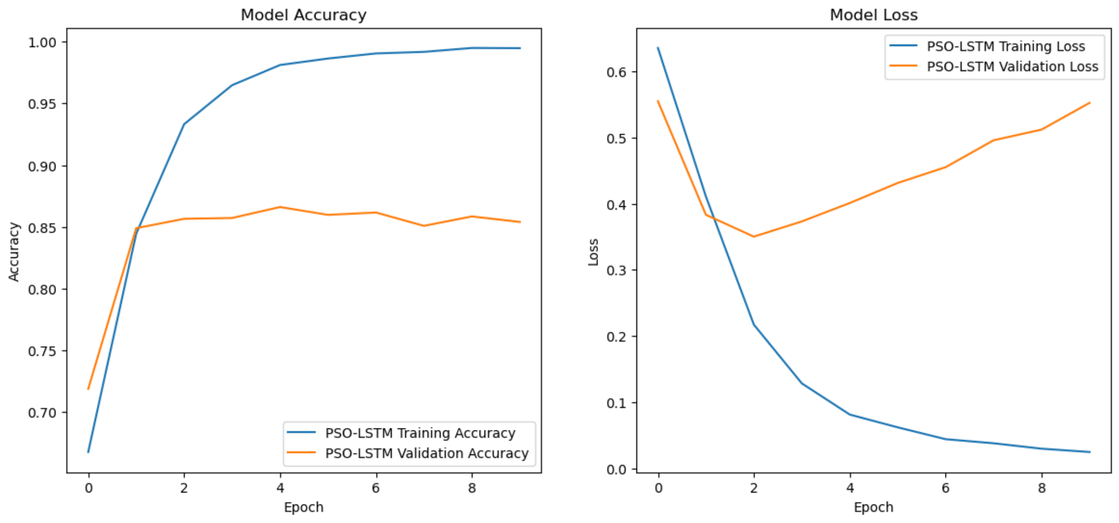

Figure 2. Model accuracy and model loss of LSTM using the PSO algorithm

The right plot shows the training and validation loss over the epochs, where the training loss decreases rapidly, nearing zero by the final epoch. In contrast, the validation loss initially decreases but then begins to rise slightly after the third epoch. This pattern of increasing validation loss alongside decreasing training loss indicates that the model may be memorizing the training data rather than learning generalizable patterns. To address this issue in future experiments, regularization techniques such as dropout could be employed to prevent overfitting and improve the model's robustness.

3.3. Confusion matrix and classification metrics

Figure 3 presents the confusion matrix for the PSO-LSTM model, which reveals that the model correctly classified 831 instances as negative and 864 instances as positive. However, it also produced 147 false positives and 127 false negatives. These results allow us to calculate important classification metrics that evaluate the model's performance. The overall accuracy of the model is 85.5%, reflecting its ability to correctly predict both positive and negative classes in most cases. The precision for the positive class, which measures the proportion of correct positive predictions out of all predicted positives, stands at approximately 85.5%. This high precision indicates that the model is generally reliable in identifying true positives, minimizing the occurrence of false alarms.

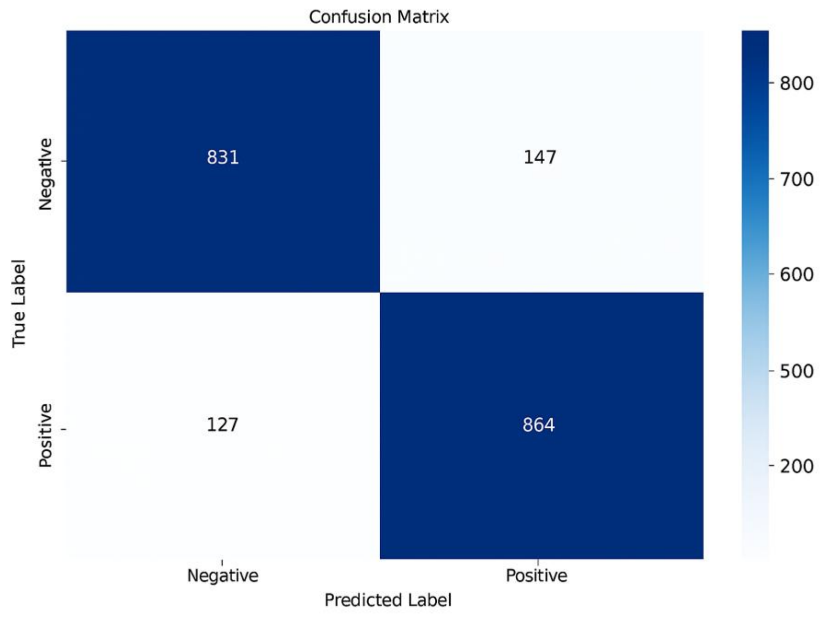

Figure 3. Confusion matrix of LSTM using the PSO algorithm

In terms of recall, which assesses the model's sensitivity to correctly identify actual positive cases, the PSO-LSTM model demonstrates a strong capability in detecting positive instances. However, there are still cases where the model fails to capture some positive examples, as evidenced by the false negatives in the confusion matrix. Finally, the F1-score, which balances precision and recall into a single metric, provides a comprehensive view of the model's classification performance. The F1-score is especially useful in cases where there is a trade-off between precision and recall, as it reflects the model's effectiveness in maintaining both a high precision and a strong recall. Overall, these metrics confirm that the PSO-LSTM model performs well, though it exhibits a minor tendency to misclassify certain instances, particularly when distinguishing between similar negative and positive cases.

3.4. Discussion

Swarm intelligence algorithms effectively optimized the LSTM model by fine-tuning the number of units in the LSTM layer. This optimization, particularly through PSO, allowed for a significant reduction in execution time without compromising model accuracy. Such improvements underscore the potential of swarm intelligence in deep learning applications, especially for tasks involving high dimensional data like cryptocurrency sentiment analysis.

3.5. Limitations and implications for future research

While the optimized models show promising results, the study is limited to cryptocurrency sentiment data and a fixed set of swarm intelligence algorithms. Future research could expand by integrating hybrid optimization techniques and testing across diverse sentiment analysis tasks. The current findings lay the groundwork for further exploration of swarm intelligence in neural network optimization.

4. CONCLUSION

This study explores the effectiveness of integrating swarm intelligence algorithms namely PSO, ACO, and CSO with LSTM networks for sentiment analysis tasks. Each of these optimization techniques was employed to fine-tune the LSTM model, specifically adjusting the number of LSTM units to enhance performance metrics. Comparative analysis reveals that the PSO-LSTM model outperformed both the ACO-LSTM and CSO-LSTM models, achieving the highest accuracy of 86.08% and the lowest loss of 0.57, alongside the shortest execution time of 58.43 seconds. These results suggest that PSO optimization effectively enhances LSTM model performance, delivering superior accuracy and faster processing times compared to the other swarm intelligence algorithms used in this study. The analysis also underscores the value of using swarm intelligence algorithms in deep learning contexts. By applying these optimizations, it was possible to refine the LSTM architecture, leading to significant improvements in sentiment classification accuracy and computational efficiency. Additionally, the use of the PSO algorithm demonstrated robust parameter tuning capabilities, providing a balance between model complexity and accuracy that is suitable for sentiment analysis applications. Given the volatile and sentiment-driven nature of cryptocurrency markets, accurate and efficient sentiment analysis models are valuable for understanding public sentiment and making data-driven predictions. The integration of PSO with LSTM networks offers promising potential for real-time cryptocurrency sentiment analysis, which could support better decision-making in trading and investment. Future research could explore the application of other optimization techniques to further improve model performance in this domain.

ACKNOWLEDGMENTS

We acknowledge the support received from Universitas Multimedia Nusantara from carrying out research work of this paper.

FUNDING INFORMATION

The authors would like to thank the Universitas Multimedia Nusantara for supporting and funding this research funded by the Internal Research Grant Universitas Multimedia Nusantara 2024 with the contract number 0015-RD-LPPM-UMN/P-INT/VI/2024.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Kristian Ekachandra	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓		✓	
Dinar Ajeng Kristiyanti		✓		✓	✓	✓	✓	✓		✓		✓	✓	✓

C :	**C**onceptualization	I :	**I**nvestigation	Vi :	**Vi**sualization
M :	**M**ethodology	R :	**R**esources	Su :	**Su**pervision
So :	**So**ftware	D :	**D**ata Curation	P :	**P**roject administration
Va :	**Va**lidation	O :	Writing - **O**riginal Draft	Fu :	**Fu**nding acquisition
Fo :	**Fo**rmal analysis	E :	Writing - Review & **E**diting		

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

INFORMED CONSENT

We have obtained informed consent from all individuals included in this study.

DATA AVAILABILITY

The data that support the findings of this study will be available in Thesis-Cryptocurrency-SentimentAnalysis https://github.com/KristianEka/Thesis-Cryptocurrency-SentimentAnalysis following a 6 months embargo from the date of publication to allow for the commercialization of research findings.

REFERENCES

[1] K. D. Shilov and A. V. Zubarev, “Evolution of bitcoin as a financial asset,” *Finance: Theory and Practice*, vol. 25, no. 5, pp. 150–171, 2021, doi: 10.26794/2587-5671-2021-25-5-150-171.

[2] Z. Zheng *et al.*, “An overview on smart contracts: challenges, advances and platforms,” *Future Generation Computer Systems*, vol. 105, pp. 475–491, 2020, doi: 10.1016/j.future.2019.12.019.

[3] J. Sedlmeir, H. U. Buhl, G. Fridgen, and R. Keller, “The energy consumption of blockchain technology: beyond myth,” *Business and Information Systems Engineering*, vol. 62, no. 6, pp. 599–608, 2020, doi: 10.1007/s12599-020-00656-x.

[4] S. Kethineni and Y. Cao, “The rise in popularity of cryptocurrency and associated criminal activity,” *International Criminal Justice Review*, vol. 30, no. 3, pp. 325–344, 2020, doi: 10.1177/1057567719827051.

[5] G. A. Tauda, A. Omara, and G. Arnone, “Cryptocurrency: highlighting the approach, regulations, and protection in Indonesia and European Union,” *Bestuur*, vol. 11, no. 1, pp. 1–25, 2023, doi: 10.20961/bestuur.v11i1.67125.

[6] J. Choi, T. Lee, K. Kim, M. Seo, J. Cui, and S. Shin, “Discovering message templates on large scale bitcoin abuse reports using a two-fold NLP-based clustering method,” *IEICE Transactions on Information and Systems*, vol. 105, no. 4, pp. 824–827, 2022, doi: 10.1587/transinf.2021EDL8092.

[7] S. N. G. Gourisetti, M. Mylrea, and H. Patangia, “Evaluation and demonstration of blockchain applicability framework,” *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1142–1156, 2020, doi: 10.1109/TEM.2019.2928280.

[8] A. Mehrotra and P. Singh, “Bitcoin ETF’s and blockchain - regulatory challenges and the way forward,” in *2021 8th International Conference on Signal Processing and Integrated Networks (SPIN)*, IEEE, 2021, pp. 916–921, doi: 10.1109/SPIN52536.2021.9566017.

[9] C. Tandon, S. Revankar, H. Palivela, and S. S. Parihar, “How can we predict the impact of the social media messages on the value of cryptocurrency? Insights from big data analytics,” *International Journal of Information Management Data Insights*, vol. 1, no. 2, 2021, doi: 10.1016/j.jjimei.2021.100035.

[10] A. Erfina and D. T. Mahardika, “Indonesian analysis sentiment on non fungible token (NFT),” *IJNMT (International Journal of New Media Technology)*, vol. 9, no. 2, pp. 69–77, 2023, doi: 10.31937/ijnmt.v9i2.2760.

[11] A. Alarifi, A. Tolba, Z. Al-Makhadmeh, and W. Said, “A big data approach to sentiment analysis using greedy feature selection with cat swarm optimization-based long short-term memory neural networks,” *Journal of Supercomputing*, vol. 76, no. 6, pp. 4414–4429, 2020, doi: 10.1007/s11227-018-2398-2.

[12] M. Y. Khan and K. Nazir, “Exerting 2D-space of sentiment lexicons with machine learning techniques: A hybrid approach for sentiment analysis,” *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 6, pp. 599–608, 2020, doi: 10.14569/IJACSA.2020.0110672.

[13] A. M. Rajeswari, M. Mahalakshmi, R. Nithyashree, and G. Nalini, “Sentiment analysis for predicting customer reviews using a hybrid approach,” in *2020 Advanced Computing and Communication Technologies for High Performance Applications (ACCTHPA)*, IEEE, 2020, pp. 200–205, doi: 10.1109/ACCTHPA49271.2020.9213236.

[14] A. Pumsirirat and L. Yan, “Credit card fraud detection using deep learning based on auto-encoder and restricted Boltzmann machine,” *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 1, pp. 18–25, 2018, doi: 10.14569/IJACSA.2018.090103.

[15] A. E. I. Brownlee, J. Adair, S. O. Haraldsson, and J. Jabbo, “Exploring the accuracy - energy trade-off in machine learning,” in *2021 IEEE/ACM International Workshop on Genetic Improvement (GI)*, IEEE, 2021, pp. 11–18, doi: 10.1109/GI52543.2021.00011.

[16] X. Huang *et al.*, “LSTM based sentiment analysis for cryptocurrency prediction,” in *Database Systems for Advanced Applications*, Cham, Switzerland: Springer, 2021, pp. 617–621, doi: 10.1007/978-3-030-73200-4_47.

[17] S. Bouktif, A. Fiaz, A. Ouni, and M. A. Serhani, “Optimal deep learning LSTM model for electric load forecasting using feature selection and genetic algorithm: Comparison with machine learning approaches,” *Energies*, vol. 11, no. 7, 2018, doi: 10.3390/en11071636.

[18] Z. Wang and Z. Lin, “Optimal feature selection for learning-based algorithms for sentiment classification,” *Cognitive Computation*, vol. 12, no. 1, pp. 238–248, 2020, doi: 10.1007/s12559-019-09669-5.

[19] J. Tang, G. Liu, and Q. Pan, “A review on representative swarm intelligence algorithms for solving optimization problems: applications and trends,” *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 10, pp. 1627–1643, 2021, doi: 10.1109/JAS.2021.1004129.

[20] D. A. Kristiyanti, Normah, and A. H. Umam, “Prediction of Indonesia presidential election results for the 2019-2024 period using Twitter sentiment analysis,” in *2019 5th International Conference on New Media Studies (CONMEDIA)*, IEEE, 2019, pp. 36–42, doi: 10.1109/CONMEDIA46929.2019.8981823.

[21] J. Shobana and M. Murali, “An efficient sentiment analysis methodology based on long short-term memory networks,” *Complex and Intelligent Systems*, vol. 7, no. 5, pp. 2485–2501, 2021, doi: 10.1007/s40747-021-00436-4.

[22] D. Barro, A. Basso, S. Funari, and G. A. Visentin, “The effects of the introduction of volume-based liquidity constraints in portfolio optimization with alternative investments,” *Mathematics*, vol. 12, no. 15, 2024, doi: 10.3390/math12152424.

[23] U. Naseem, I. Razzak, and P. W. Eklund, “A survey of pre-processing techniques to improve short-text quality: a case study on hate speech detection on Twitter,” *Multimedia Tools and Applications*, vol. 80, no. 28–29, pp. 35239–35266, 2021, doi: 10.1007/s11042-020-10082-6.

[24] C. Steven and W. Wella, “The right sentiment analysis method of indonesian tourism in social media Twitter,” *IJNMT (International Journal of New Media Technology)*, vol. 7, no. 2, pp. 102–110, 2020, doi: 10.31937/ijnmt.v7i2.1732.

[25] D. A. Kristiyanti, I. S. Sitanggang, Annisa, and S. Nurdiati, “Feature selection using new version of V-shaped transfer function for salp swarm algorithm in sentiment analysis,” *Computation*, vol. 11, no. 3, 2023, doi: 10.3390/computation11030056.

[26] K. M. Ridhwan and C. A. Hargreaves, “Leveraging Twitter data to understand public sentiment for the COVID-19 outbreak in Singapore,” *International Journal of Information Management Data Insights*, vol. 1, no. 2, 2021, doi: 10.1016/j.jjimei.2021.100021.

[27] A. Kalinin, A. Kolmogorova, G. Nikolaeva, and A. Malikova, "Mapping texts to multidimensional emotional space: Challenges for dataset acquisition in sentiment analysis," *Communications in Computer and Information Science*, vol. 859, pp. 361–367, 2018, doi: 10.1007/978-3-030-02846-6_29.
[28] T. Pano and R. Kashef, "A complete vader-based sentiment analysis of bitcoin (BTC) tweets during the ERA of COVID-19," *Big Data and Cognitive Computing*, vol. 4, no. 4, pp. 1–17, 2020, doi: 10.3390/bdcc4040033.
[29] R. N. Satrya, O. N. Pratiwi, R. Y. Farifah, and J. Abawajy, "Cryptocurrency sentiment analysis on the Twitter platform using support vector machine (SVM) algorithm," in *2022 International Conference Advancement in Data Science, E-learning and Information Systems (ICADEIS)*, IEEE, 2022, pp. 1–5, doi: 10.1109/ICADEIS56544.2022.10037413.
[30] A. Banik, C. Behera, T. V. Sarathkumar, and A. K. Goswami, "Uncertain wind power forecasting using LSTM-based prediction interval," *IET Renewable Power Generation*, vol. 14, no. 14, pp. 2657–2667, 2020, doi: 10.1049/iet-rpg.2019.1238.
[31] A. Gupta and S. Srivastava, "Comparative analysis of ant colony and particle swarm optimization algorithms for distance optimization," *Procedia Computer Science*, vol. 173, pp. 245–253, 2020, doi: 10.1016/j.procs.2020.06.029.
[32] D. A. Kristiyanti, I. S. Sitanggang, Annisa, and S. Nurdiati, "Feature selection technique model for forest and land fire data sentiment analysis: comparison of SSA, PSO, and ALO," in *2023 7th International Conference on New Media Studies (CONMEDIA)*, IEEE, 2023, pp. 236–242, doi: 10.1109/CONMEDIA60526.2023.10428170.
[33] A. Kaveh, "Particle swarm optimization," in *Advances in Metaheuristic Algorithms for Optimal Design of Structures*, Cham: Springer International Publishing, 2017, pp. 11–43, doi: 10.1007/978-3-319-46173-1_2.
[34] F. Moslehi and A. Haeri, "A novel hybrid wrapper–filter approach based on genetic algorithm, particle swarm optimization for feature subset selection," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 3, pp. 1105–1127, 2020, doi: 10.1007/s12652-019-01364-5.
[35] C. Blum, "Ant colony optimization: Introduction and recent trends," *Physics of Life Reviews*, vol. 2, no. 4, pp. 353–373, 2005, doi: 10.1016/j.plrev.2005.10.001.
[36] S. R. Ahmad, A. A. Bakar, and M. R. Yaakub, "Ant colony optimization for text feature selection in sentiment analysis," *Intelligent Data Analysis*, vol. 23, no. 1, pp. 133–158, 2019, doi: 10.3233/IDA-173740.
[37] A. M. Ahmed, T. A. Rashid, and S. A. M. Saeed, "Cat swarm optimization algorithm: a survey and performance evaluation," *Computational Intelligence and Neuroscience*, vol. 2020, 2020, doi: 10.1155/2020/4854895.
[38] S.-C. Chu, P. Tsai, and J.-S. Pan, "Cat swarm optimization," in *PRICAI 2006: Trends in Artificial Intelligence*, Berlin, Heidelberg: Springer, 2006, pp. 854–858, doi: 10.1007/11801603_94.

BIOGRAPHIES OF AUTHORS

Kristian Ekachandra holds a bachelor's degree in information systems from Universitas Multimedia Nusantara, Indonesia. Currently, he serves as a mentor for Mobile Development at Bangkit Academy 2024, a program supported by Google, GoTo, and Traveloka. He is an alumnus of Bangkit Academy 2023, where he specialized in mobile development. Additionally, he gained valuable experience as an intern at Kompas Gramedia, working as a Mobile Apps Programmer. His research interests encompass sentiment analysis, feature selection, and the application of sentiment analysis to cryptocurrency. He can be contacted at email: kristian.ekachandra@student.umn.ac.id.

Dinar Ajeng Kristiyanti received her master's degree in computer science, and bachelor degree in information system from Universitas Nusa Mandiri, Indonesia. Currently she is a Ph.D. candidate in computer science at IPB University, Indonesia. She is a lecturer at Department of Information System in Universitas Multimedia Nusantara, Indonesia. Her research interests include sentiment analysis, text mining, feature selection, optimization, data science, and machine learning. She has published over 19 papers in international journals and conferences. She can be contacted at email: dinar.kristiyanti@umn.ac.id.