Stock Price Forecasting using Convolutional Neural Networks and Optimization Techniques

Nilesh B. Korade[1]
Research Scholar, Department of Computer Science and Engineering, Madhyanchal Professional University
Ratibad, Bhopal, Madhya Pradesh-462044, India

Dr. Mohd. Zuber[2]
Associate Professor, Department of Computer Science and Engineering, Madhyanchal Professional University
Ratibad, Bhopal, Madhya Pradesh-462044, India

***Abstract*—Forecasting the correct stock price is intriguing and difficult for investors due to its irregular, inherent dynamics, and tricky nature. Convolutional neural networks (CNN) have impressive performance in forecasting stock prices. One of the most crucial tasks when training a CNN on a stock dataset is identifying the optimal hyperparameter that increases accuracy. In this research, we propose the use of the Firefly algorithm to optimize CNN hyperparameters. The hyperparameters for CNN were tuned with the help of Random Search (RS), Particle Swarm Optimization (PSO), and Firefly (FF) algorithms on different epochs, and CNN is trained on selected hyperparameters. Different evaluation metrics are calculated for training and testing datasets. The experimental finding demonstrates that the FF method finds the ideal parameter with a minimal number of fireflies and epochs. The objective function of the optimization technique is to reduce MSE. The PSO method delivers good results with increasing particle counts, while the FF method gives good results with fewer fireflies. In comparison with PSO, the MSE of the FF approach converges with increasing epoch.**

***Keywords*—Convolutional neural networks; swarm intelligence; random search; particle swarm optimization; firefly**

I. Introduction

The non-linear characteristics of stock market data make it challenging to guess the next movement of stock value. Exact stock forecasting can boost consumer and seller confidence in the stock market, which will attract investors to buy shares and grow the nation's economy [1]. The accuracy of neural networks and their variations in predicting stock prices is rising day by day [2, 3]. Several studies using time-series data have demonstrated that CNN is useful for forecasting issues [4]. CNN can accurately and efficiently identify the changing trend in stock value, and it may be used in other financial transactions. Choosing the best CNN parameters is one of the difficulties we encounter while constructing CNN architectures. The result may vary if we apply different parameter values to CNN architectures while solving the same problem. An optimization process is defined as determining the ideal combination of inputs to reduce or enhance the cost of the objective function without impacting training performance. Optimization is a computational problem whose aim is to extract the best solution among all possible solutions. The hyperparameters that affect CNN's architecture include filters, kernel size, stride, padding, pool size, batch size, epoch, and others. In a reasonable amount of time, we would like to identify the ideal set of hyperparameter values for a particular dataset. To address this issue, several researchers have proposed various techniques based on evolutionary computation to automatically detect the best CNN structures and improve performance [5]. It can be difficult to determine the best parameter in a high-dimensional space.

The procedure used to set the hyper-parameter values is typically a random search, including running a number of tests or making manual adjustments. Random search is the process of selecting and analyzing inputs randomly for the objective function [6]. Swarm intelligence (SI) is inspired by social behavior found in nature, such as the movement of fish and birds. Based on a group's intelligence and behavior, the SI algorithms have a significant ability to determine the optimal solution [7]. The PSO algorithm is considered one of the meta-heuristic evolutionary algorithms. Each particle in PSO has its own position, velocity, and fitness and also keeps track of its best fitness value and best fitness position. The PSO maintains a record of the global best fitness position and the global best fitness value [8]. The FF algorithm is a metaheuristic algorithm based on the attraction of fireflies towards brighter fireflies. The FF algorithm is able to identify optimal parameters for CNN that minimize the error or fitness function with fewer iterations. Each firefly in the FF algorithm has a position in the search space that corresponds to a solution, and it progresses toward more brilliant solutions. For each iteration, the FF algorithm keeps track of the best position, which will reduce the objective function cost [9].

This study evaluates optimization techniques such as RS, PSO, and FF on CNN to forecast stock prices. The Tata Motors stock dataset was taken from Yahoo Finance between January 1, 2003, and September 30, 2022. The CNN is trained on hyperparameter return using RS, PSO, and FF algorithms, and the trained model is used to forecast stock prices for the next day as well as stock prices for the entire month of September. The results show that the FF method returns the best hyperparameters with fewer fireflies, reduces MSE, and requires the fewest training epochs. The remaining paper is structured as follows: In Section II, the stock price forecasting literature is discussed, and the dataset used, the CNN architecture, and different optimization techniques are explained in Section III. The evaluation metrics used and the accuracy of CNN trained on hyperparameter returns by different optimization techniques are discussed in Section IV. We conclude the work in Section V with a summary and recommendations for additional research.

II. Literature Survey

Traditional stock value analysis is based on economics and finance, and it primarily focuses on external factors influencing stock prices, such as international relations, exchange rates, interest rates, business policy, financial institutions, political factors, etc. It's challenging to convince people of the accuracy of the traditional fundamental analysis method. The technical analysis approach primarily concentrates on the movement of the stock price, psychological expectations of investors, trading volume, historical data, etc. [10]. In comparison to other deep learning models, CNN is more accurate and can detect both uptrend and downtrend stock movements [11]. Setting hyperparameters is necessary for CNN implementation, which influences accuracy, learning time, and CNN architecture. The effectiveness of machine learning will be significantly increased if an effective hyperparameter optimization algorithm can be designed to optimize any specific machine learning method. Bayesian optimization based on the Bayesian theorem can be used to solve the hyperparameter tuning problem, as it can be considered an optimization issue [12]. Manual search and RS are the most widely used strategies for hyper-parameter optimization. Randomly searching is more efficient for hyperparameter optimization, where random combinations of hyperparameters are chosen and used to train a model. The hyperparameter combinations with the best costs are selected [13]. Finding the ideal collection of hyperparameter values in a reasonable amount of time is difficult because the values of the hyperparameters change when the dataset changes. The weighted random search approach, which locates the global optimum more quickly than RS, combines RS with a probabilistic greedy heuristic method [14]. Reinforcement learning (RL)-based optimization algorithms give good performance and highly competitive results for CNN hyperparameter tuning with fewer iterations [15].

The hyperparameters include the number of layers, the size of the kernel in each layer, the loss function, the optimizer, and many others. Integers or real numbers may be used as hyperparameters, and there is an infinite range of possible values for each, so CNN hyperparameter tuning is a challenging problem. Swarm intelligence algorithms have been used for decades to solve challenging optimization issues and give promising results [16]. The fish swarm optimization method utilizes a fish's social behavior to carry out a variety of tasks, with each fish serving as a potential solution to the optimization problem. The aquarium is the design space where the fish are found, and the food density is related to an objective function to be optimized [17]. The bee algorithm emulates honey bee foraging activities to find the optimal solution to an optimization issue. The food source or flower is considered a candidate solution, and the population of n bees searches the solution space [18]. Two important parameters of a deep learning network are the number of hidden layers and the number of neurons in each layer. The PSO method has great potential for optimizing parameters and saving precious computational resources when deep learning models are being tuned. Each particle in a PSO has three properties: a position that corresponds to a solution's; a velocity that is a moving parameter; and a fitness value that is calculated by an objective function that takes the particle's position into consideration. The position of every particle in the swarm is updated such that it will migrate toward the one with the best position [19]. One of the most effective metaheuristic algorithms for solving optimization issues is the FF algorithm, which is based on the flashing characteristic of fireflies. The unisex nature of fireflies makes them attracted to brighter lights, and as distance increases, their attractiveness and brightness will decrease [20]. Metaheuristic algorithms, particularly evolutionary and SI algorithms, are strong methods for addressing a wide range of challenging engineering issues that arise in the real world. The echolocation abilities of microbats served as an inspiration for the bat algorithm. Each bat has some position, flying randomly with velocity, fixed frequency, varying wavelength, and loudness in order to search for prey [21]. While training CNN on a stock dataset, it is required to select optimal parameters that improve training performance and reduce error. It is essential to choose the best optimization technique to boost the CNN's performance and reduce errors.

III. Methodology

A. Data Description

From January 1, 2003, until September 30, 2022, historical stock information for Tata Motors was retrieved from Yahoo Finance [22]. Table I shows features and values for Tata Motors stock data. The dataset contains features such as date, open value, low value, close value, high value, volume, and adjacent close value.

It is typical to assess the calculation of profit or loss using the closing price of a stock on a given date; hence, we considered the closing price as the target variable. The plot of the target variable against time is shown in Fig. 1.

TABLE I. Tata Motors Stock Price Dataset

Date	High	Low	Open	Close	Volume	Adj. Close
2003-01-01	31.26	30.66	30.66	31.15	4460733.00	25.15
2003-01-02	31.41	30.76	31.29	30.84	4800428.00	24.91
2003-01-03	31.22	30.77	30.84	30.88	3939402.00	24.94
….	**….**	**….**	**….**	**….**	**….**	**….**
2022-09-28	406.60	392.85	394.90	399.10	18114880.00	399.10
2022-09-29	413.25	399.60	411.00	402.25	20725995.00	402.25
2022-09-30	408.25	392.50	398.00	404.60	20951277.00	404.60

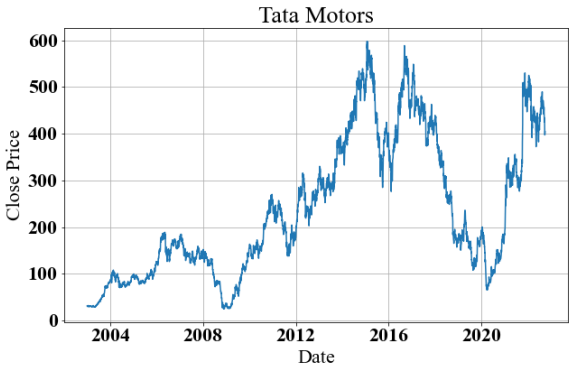

Fig. 1. Tata Motors Closing Price.

B. CNN

Deep learning is a subset of machine learning that contains algorithms that are designed to mimic how neural networks or the human brain function. CNN, or ConvNets, is one of the models that has significantly impacted image analysis and computer vision. Recently, there has been a rising interest among researchers in using CNNs for time-series forecasting challenges. We could use a CNN model effectively for the prediction issue if we could transform the 1D time-series sequence into an input image matrix structure. Fig. 2 shows the architecture for CNN.

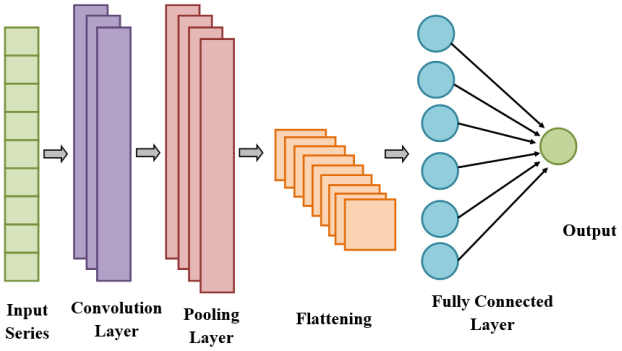

Fig. 2. CNN Architecture.

The convolution layer, the first layer of the CNN, extracts features from the input by sliding the filter over the input. In order to extract features, CNN uses filter matrices of various sizes that are multiplied by the input matrix. The stride and padding operations influence how the convolution process takes place, resulting in an increase or decrease in the output matrix's dimensions. The size of an output matrix is regulated by CNN using padding, which specifies the number of pixels that are added to an input matrix during the convolution process. The stride, or number of pixels shifted, regulates how the filter convolves across the input matrix. The following formula can be used to calculate the convolution process output matrix size:

$$d = \left(\left(\frac{I - F + 2P}{S}\right) + 1\right) \quad (1)$$

Where I and F stand for the input and filter matrix dimensions, respectively, S stands for stride, and P stands for padding. The pooling layer's aim is to gradually reduce the spatial dimension of the representation in order to decrease the number of parameters and calculations in the network by multiplying the resultant matrix from the convolution layer and pooling matrix. The input units are randomly set to zero at a random frequency at each step during training by the dropout layer, which helps in preventing overfitting. The dense layer is a combination of a fully connected layer and an output layer that implements the following operation:

$$O = \text{activation}\left((I \,.K) + \text{bias}\right) \quad (2)$$

where 'I' is the input vector, 'K' is a weight matrix generated by the layer, and "bias" is a bias vector generated by the layer. The most frequently used activation functions are tanh, sigmoid, relu, etc.

$$sigmoid\ function\ g(z) = \frac{1}{(1+e^{-z})} \quad (3)$$

$$tanh\ function\ g(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \quad (4)$$

$$relu\ function\ g(z) = max(0\,, z) \quad (5)$$

C. Random Search

Hyperparameter tuning refers to the building of the model architecture from the available space. Random search is a method for finding the optimal solution to build the model by selecting combinations of the hyperparameters at random or using probability.

$$x^{*} = \arg min_{x \in A} f(x) \quad (6)$$

where 'f' is the objective function such as MSE to be minimized on the validation set. 'x*' is the set of hyperparameters for which the objective function returns the lowest value, and any value from the domain 'A' can be assigned to 'x' [23]. The flowchart for RS-CNN is shown in Fig. 3.

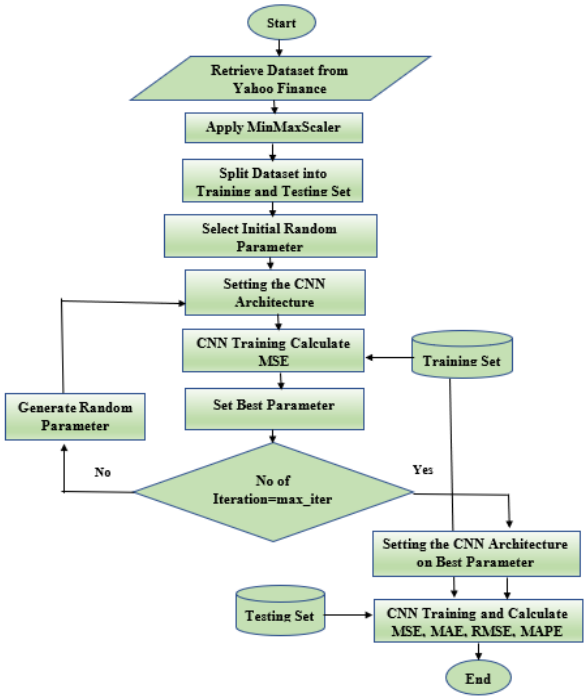

Fig. 3. Flowchart of the RS-CNN Algorithm.

D. PSO Algorithm

The particle swarm optimization technique is inspired by the social behavior of bird swarms and was designed by Kennedy and Eberhart in 1995. PSO simulates swarm behaviors to optimize the way of finding food, and each bird in the swarms constantly changes the search pattern based on its own and other members' learning experiences [24]. The objective function in data science entails feeding a candidate solution into a model and evaluating it against the training dataset. The cost may be an error score, also known as the loss of the model, which is to be decreased, or an accuracy score, which is to be increased. The optimization process aims to increase or decrease the cost of an objective function. There may be multiple local maximums and minimums for objective functions, but only one global maximum or minimum. Each particle in a PSO has a position, which corresponds to the attributes of a solution; a velocity, which is a moving parameter; and a fitness value, which is determined by an objective function taking into account the position of the particle [25]. The particle's quality is measured by its fitness value. Each swarm particle's position is updated such that it will move closer to the one with the best position. Each particle maintains pbest, the best solution each particle independently found, and gbest, the best solution found by all particles, to update its position and velocity in each iteration [26]. The processing steps of the PSO algorithm are mentioned below [27].

Step 1: Generate random position p and velocity v in all dimensions by using the following equation:

$$x_i = \mathrm{l} + \mathrm{r} * (\mathrm{u} - \mathrm{l}) \quad (7)$$

$$v_i = l + r * (u - l) \quad (8)$$

x_i represents the position of particle i, r is a random number, l represents the lower bound, and u represents the upper bound. For velocity calculation, the lower bound is 0 and the upper bound is 1.

Step 2: Calculate objective function value $f(x_i)$ for each particle where i=1..n. The initial position of a particle is pbest for that particle, i.e., pbest=x_i.

Step 3: Find gbest i.e., best position from all particle.

Step 4:

For t=1 to max_iterations

For i= 1 to no_of_particles

Update particle velocity and position

Calculate Objective function value and pbest

End for i

Find gbest position having less cost

End for t

Equation 9 describes how to calculate particle velocity, and Equation 10 describes how to calculate particle position.

$$V_i^{t+1} = WV_i^t + C_1 r_1^t (P_t^b - X_t) + C_2 r_2^t (P^q - X_t) \quad (9)$$

$$X_i^{t+1} = X_i^t + V_i^{t+1} \quad (10)$$

where V_i^{t+1} and X_i^{t+1} are the new calculated velocity and position, t represents the iteration number, V_i^t is the velocity of the i^{th} particle at iteration t, X_i^t is the position of a particle. W is the inertia weight that regulates the motion of the particles. The particle may continue travelling in the same direction if W = 1, because the particle's motion is entirely determined by the preceding motion. if 0≤W<1, this influence is diminished, causing a particle to travel to different areas within the search domain. The C1, C2 are the correlation factors and if C1=C2 =0, all particles continue to move at their current speed until they collide with the search space border. The r_1 and r_2 are the random number in between 0 and 1. The P_t^b is the particle's best position for iteration t and P^q is the global best position. The updated position is used to evaluate the value for the objection function in each iteration. The particle best position (pbest) and global best position (gbest) are updated based on the objective function return value. The best parameters returned by PSO were used to train CNN, and performance was calculated using evaluation metrics. The flowchart for PSO-CNN is shown in Fig. 4.

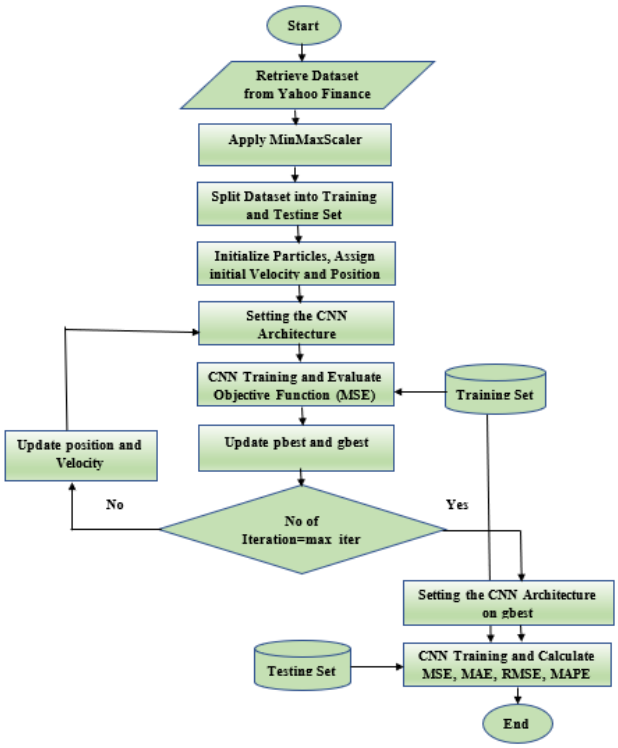

Fig. 4. Flowchart of the PSO-CNN Algorithm.

E. Firefly Algorithm

An optimization issue is one where an objective function is maximized or minimized by selecting appropriate values from a set of possible values for the variables. This research takes into consideration a minimization problem in which the objective function calculates and returns the difference between the actual and forecasted stock price. The Xin-Shi Yang-developed Firefly Algorithm is a stochastic and metaheuristic optimization algorithm inspired by nature. It was designed to mimic the social behavior of fireflies based on their flashing and attraction properties [28].

Below are the guidelines for the FF optimization algorithm:

- An objective function is used to determine the brilliance of a firefly.
- Any firefly is attracted towards more brilliant fireflies, as all fireflies are unisex, and they will move randomly if there is no brilliant firefly.
- Attractiveness is directly proportional to brilliance, and as distance grows, both attractiveness and brilliance will decrease [29].

In an optimization problem, the objective function accepts input variables and returns costs based on their calculation. By taking into account the optimal parameters, an optimization algorithm aims to reduce or enhance the cost return by the objective function [30]. The processing steps of the firefly algorithm are as below.

Step 1: Generate a random position X in all dimension using Equation 11.

$$X_i = \mathrm{l} + \mathrm{r} * (\mathrm{u} - \mathrm{l}) \quad (11)$$

X_i is position of firefly i, r is random number, l is lower bound and u is upper bound.

Step 2: Calculate cost, determined by objective function $f(X_i)$, for each firefly where i=1..n.

Step 3:

For t=1 to max_iterations
  For i= 1 to n
    For j= 1 to n
      If $f(X_i) > f(X_j)$
        Move firefly i towards j in all d dimensions
        Evaluate new position and update cost
      Else
        Move firefly i randomly
      End if
    End for j
  End for i
Find best position having less cost
End for T

The following equation describes how a firefly i will migrate when it encounters a firefly j, which is more illuminated [31].

$$X_i^{t+1} = X_i^t + \beta_0 e^{-\gamma r_{ij}^2} * \left(X_j^t - X_i^t\right) + \alpha_t \epsilon_i^t \quad (12)$$

where t is the iteration count, X_i^{t+1} is the new position of the i^{th} firefly, X_i^t is the current position of the i^{th} firefly. The term β_0 and α_t in the equation are the attraction and randomization parameters respectively. The recommended values used in most of the implementation are β_0=1 and $\alpha \in$ [0, 1] respectively. The term ϵ_i is a random number obtained from a Gaussian or uniform distribution. Theoretically, the light absorption coefficient (γ) varies from 0 to ∞, but in most applications, it frequently ranges from 0.01 to 100. The distance r between fireflies i and j can be calculated using the Euclidean distance formula [32].

$$r_{i,j} = \sqrt{\sum_{k=1}^{d} \left(X_{i,k} - X_{j,k}\right)^2} \quad (13)$$

The flowchart for FF-CNN is shown in Fig. 5. Initially, the FF algorithm initializes a population, assigns a random position, and calculates the objective function cost. The initial position of the firefly is the best position. The cost of each firefly is compared with every other firefly in each iteration, and if the cost of firefly i is greater than the cost of firefly j, then firefly i will move towards firefly j. After n iterations, the best positions were used to train CNN, and accuracy on the training and testing datasets was calculated.

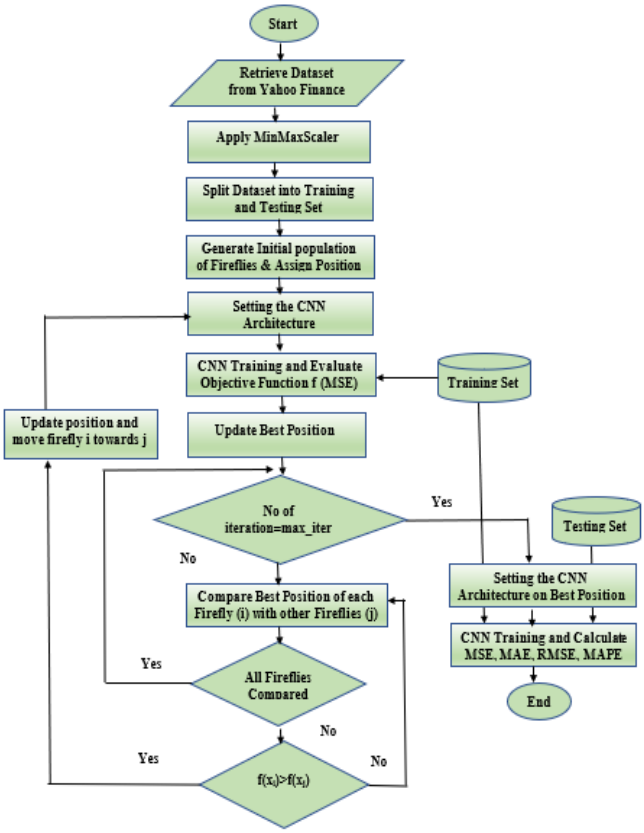

Fig. 5. Flowchart of the FF-CNN Algorithm.

IV. Results

The dataset contains 4902 records split into two sets: the training set has 4882 records covering the period from 01/01/2003 to 31/08/2022; and the testing set has 22 records covering the period from 01/09/2022 to 30/09/2022. The CNN models are evaluated for their ability to predict the stock price for the following day and for the entire month of September. Based on the past 1000 samples, the stock price for the following day is computed, and the stock price for the entire month of September is predicted using the entire training set. The assessment metrics should show the difference between the actual and anticipated value in the stock forecasting problem, which is a regression problem. The performance of

the model is evaluated against MSE, MAE, RMSE, and MAPE [33]. The formula for evaluation metrics is as below.

$$Mean\ Square\ Error = \frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \hat{Y}_i\right)^2 \quad (14)$$

$$Mean\ Absolute\ Error\ = \frac{1}{N}\sum_{i=1}^{N}\left|Y_i - \hat{Y}_i\right| \quad (15)$$

$$Root\ Mean\ Square\ Error = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \hat{Y}_i\right)^2} \quad (16)$$

$$\underset{Percentage\ Error}{Mean\ Absolute} = \frac{1}{N}\sum_{i=1}^{N}\frac{|Y_i - \hat{Y}_i|}{Y_i} * 100\% \quad (17)$$

where N is the total number of samples in the stock dataset, Y_i is the actual stock price, and $\hat{Y}_i$ is the predicted stock price by model. The evaluation metrics shows how forecasted value closer to actual and variation in between them.

The goal of an optimization problem is to identify the optimal solution from a collection of possible options. The parameters for CNN are convolution layer filters, kernel size, pool size in max pooling operation, batch size, etc. If we apply different parameter values to CNN architectures while solving the same problem, the result may vary. Selecting the right hyperparameter gives better accuracy with a minimum epoch. The hyperparameter values returned by the RS, PSO, and FF algorithms were used to train the CNN model. Table II displays the upper and lower bounds for the hyperparameters, and step size is used to increase or decrease their value. Table III provides a summary of the control parameters for the FF and PSO algorithms.

Tables IV and V compare the training and testing performances of CNN, RS-CNN, FF-CNN, and PSO-CNN. The manual hyperparameter was considered while training CNN on different epochs, and resultant metrics were calculated. The number of iterations in the RS-CNN is set to 10, and the current iteration's result is compared to the best one. The FF and PSO algorithms are executed using 5 and 10 particles for different epochs, and CNN is trained using the hyperparameter return from the FF and PSO algorithms. The evaluation metrics for the training set and the testing set are calculated. The objective function for FF and PSO is to reduce MSE. The results reveal that PSO delivers good results with increasing particle counts, while the firefly method gives good results with fewer fireflies. With a large number of particles, the PSO can find the ideal hyperparameter, and the MSE starts decreasing with increasing epoch. In comparison with PSO, the MSE of the FF approach converges with increasing epoch.

The FF-CNN method identifies the ideal parameter with fewer iterations and converges the result with subsequent iterations. This indicates that the FF-CNN strategy has the potential to be more beneficial than CNN, RS-CNN, or PSO-CNN in resolving the stock forecasting problem.

The manual hyperparameter selection method's cost decreases with increasing epoch, while the RS-CNN method's cost varies as parameters are randomly chosen. PSO-CNN requires more particles and epochs, while FF-CNN achieves better outcomes with fewer fireflies and epochs. If we increase the firefly or epoch, the results converge in the early stages. The evaluation metrics value of a CNN trained on a hyperparameter return by various optimization approaches is shown in Fig. 6 and 7.

TABLE II. UPPER AND LOWER BOUNDS FOR THE HYPERPARAMETERS

Parameter	Lower Bound	Upper Bound	Step Size
Convolution layer filters	16	128	16
kernel size	1	5	1
pool size	1	5	1
batch size	32	256	32

TABLE III. SUMMARY OF THE CONTROL PARAMETERS

PSO-CNN		FF-CNN	
Inertia weight (W)	0.5	Attraction parameter (β_0)	0.97
Correlation factors (C_1, C_2)	0.5, 0.5	Randomization parameter (α_t)	1
Random number (r_1, r_2)	0≤1	Absorption coefficient (γ)	0.01
Maximum iteration number	10	Maximum iteration number	10

TABLE IV. OPTIMIZATION METHODS AND EVALUATION METRICS FOR TRAINING DATASET

Epoch	Metrics	CNN	RS-CNN	PSO-CNN Particle=5	FF-CNN Firefly=5	PSO-CNN Particle=10	FF-CNN Firefly=10
1	MSE	2807.70	1394.60	2028.91	1521.22	1812.57	836.48
	MAE	40.61	28.16	35.05	28.80	31.90	22.19
	RMSE	52.98	37.34	45.04	39.00	42.57	28.92
	MAPE	20.27	13.22	16.14	13.07	15.88	12.95
3	MSE	1466.55	593.12	938.39	569.33	718.46	552.14
	MAE	28.86	18.28	22.99	18.18	19.93	17.73
	RMSE	38.29	24.35	30.63	23.86	26.80	23.49
	MAPE	14.86	8.57	11.06	8.24	9.30	8.22
5	MSE	863.72	1174.88	786.61	330.05	355.49	421.93
	MAE	21.68	26.18	21.20	13.70	14.02	15.58
	RMSE	29.38	34.27	28.04	18.16	18.85	20.54
	MAPE	10.05	13.40	10.27	7.80	7.78	7.09
7	MSE	573.12	953.33	444.52	300.75	335.13	328.43
	MAE	17.89	27.05	15.79	13.10	13.73	13.36
	RMSE	23.93	30.87	21.08	17.34	18.30	18.12
	MAPE	8.21	6.83	8.67	6.61	6.68	5.59
9	MSE	550.88	435.38	454.32	303.94	227.92	198.12
	MAE	17.64	15.85	16.19	12.98	11.37	10.38
	RMSE	23.47	20.86	21.31	17.43	15.09	12.56
	MAPE	8.22	6.63	8.24	6.39	6.58	5.56

TABLE V. OPTIMIZATION METHODS AND EVALUATION METRICS FOR TESTING DATASET

Epoch	Evaluation Metrics	CNN	RS-CNN	PSO-CNN Particle=5	FF-CNN Firefly=5	PSO-CNN Particle=10	FF-CNN Firefly=10
1	MSE	3360.60	2108.53	1895.34	1457.71	2300.72	1895.70
	MAE	52.54	38.79	38.26	32.72	42.60	38.75
	RMSE	57.97	45.91	43.53	38.18	47.96	43.53
	MAPE	12.41	9.24	9.07	7.78	10.09	9.17
3	MSE	1743.05	681.92	1897.19	325.52	1458.45	381.67
	MAE	36.46	19.93	36.96	13.91	34.31	14.67
	RMSE	41.74	26.11	43.55	18.04	38.18	19.53
	MAPE	8.65	4.79	8.80	3.33	8.11	3.53
5	MSE	1687.88	1185.88	1401.91	188.75	523.64	213.27
	MAE	36.65	28.02	30.78	11.74	18.46	10.84
	RMSE	41.08	34.43	37.44	13.73	22.88	14.60
	MAPE	8.67	6.70	7.35	2.66	4.41	2.60
7	MSE	850.56	953.33	695.28	122.97	191.09	121.54
	MAE	23.45	27.05	22.52	9.55	10.19	8.92
	RMSE	29.16	30.87	26.36	11.08	13.82	11.02
	MAPE	5.61	6.41	5.35	2.17	2.43	2.09
9	MSE	509.63	717.68	336.91	104.75	171.86	102.89
	MAE	17.71	25.39	13.60	8.44	11.18	6.36
	RMSE	22.57	26.78	18.35	10.23	13.10	8.69
	MAPE	4.24	5.78	3.27	1.96	2.55	1.94

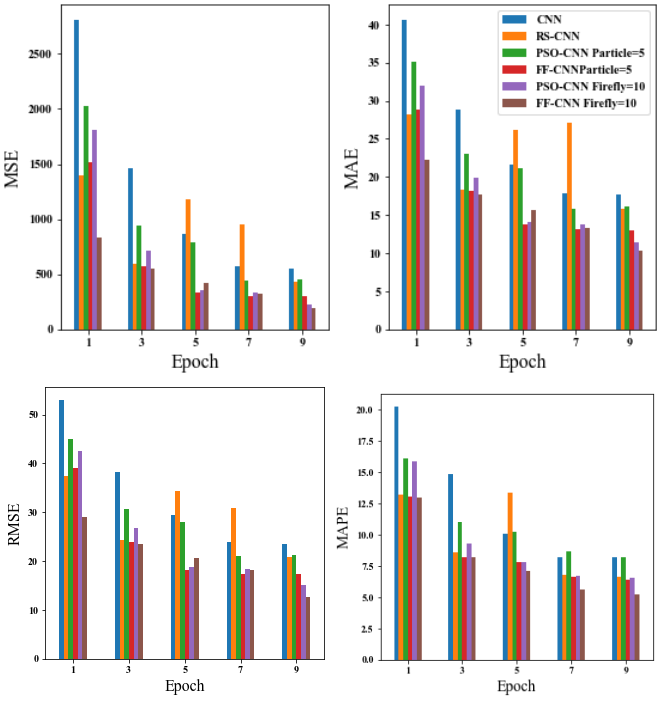

Fig. 6. Evaluation Metrics for Training Dataset.

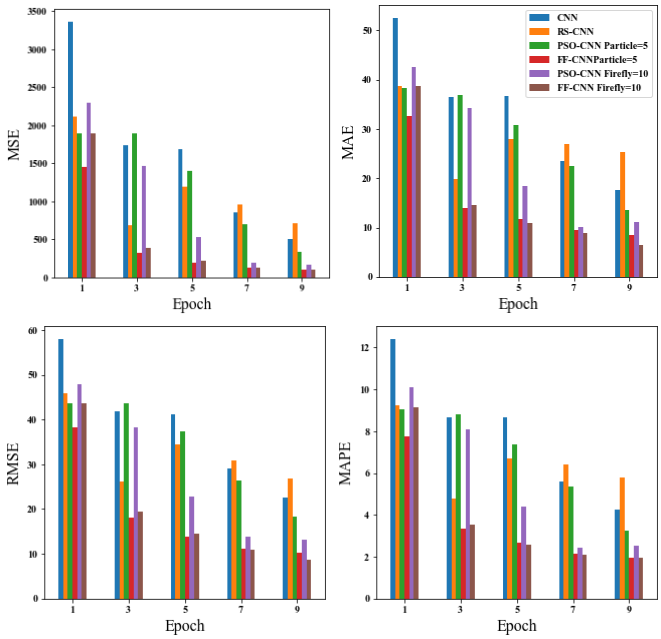

Fig. 7. Evaluation Metrics for Testing Dataset.

The result shows that the FF-CNN technique returns the best hyperparameter for CNN that gives the best accuracy and a low value for evaluation metrics.

V. CONCLUSION

Multiple parameters, including convolution layer filters, kernel size, pool size of the max pooling operation, batch size, etc., must be determined while training a stock dataset using a CNN. The selection of hyperparameters in CNN has an impact on forecasting accuracy. When selecting a hyperparameter, it is essential to apply optimization techniques that reduce the objective function's cost. This research compares manual selection, random search, PSO, and FF optimization algorithms in order to choose the appropriate hyperparameter for training CNN on a stock dataset. The outcome demonstrates that PSO reduces costs as particle size and epoch increase, whereas FF has the ability to accomplish this with lower firefly counts and epoch. Less MSE is given for both the training and testing datasets by the CNN that has been trained on hyperparameter returns by the FF method. The experimental instance reveals that the FF-CNN framework outperformed other state-of-the-art methods in terms of both the quality of the best solutions and the efficient use of computational resources. Future improvements will focus on modifying the FF algorithm to identify the ideal parameter with fewer iterations, and multichannel CNN can boost the accuracy of the algorithm.

REFERENCES

[1] Milka Grbić, "Stock market development and economic growth: the case of the Republic of Serbia," Post-Communist Economies, 33(4), pp. 484-499, 2021, doi: 10.1080/14631377.2020.1745566.

[2] L. Sayavong, Z. Wu, and S. Chalita, "Research on Stock Price Prediction Method Based on Convolutional Neural Network," 2019 International Conference on Virtual Reality and Intelligent Systems (ICVRIS),), pp. 173-176, 2019, doi: 10.1109/ICVRIS.2019.00050.

[3] W. Budiharto, "Data science approach to stock prices forecasting in Indonesia during Covid-19 using Long Short-Term Memory (LSTM)," Journal of Big Data, 8(47), 2021, doi: 10.1186/s40537-021-00430-0.

[4] R. Chauhan, H. Kaur, and B. Alankar, "Air Quality Forecast using Convolutional Neural Network for Sustainable Development in Urban Environments," Sustainable Cities and Society, 75, 2021, doi: 10.1016/j.scs.2021.103239.

[5] G. Habib, and S. Qureshi, "Optimization and acceleration of convolutional neural networks: A survey," Journal of King Saud University –Computer and Information Sciences, 34(7), pp. 4244-4268, 2022, doi: 10.1016/j.jksuci.2020.10.004.

[6] D.G. Misganu, and K. G. Arnt, "Mapping Seasonal Agricultural Land Use Types Using Deep Learning on Sentinel-2 Image Time Series," remote sensing, 13(2), pp. 289-305, 2021, doi: 10.3390/rs13020289.

[7] J. Xue, and B. Shen, "A novel swarm intelligence optimization approach: sparrow search algorithm," Systems Science & Control Engineering, 8(1), pp. 22–34, 2020, doi: 10.1080/21642583.2019.1708830.

[8] K. Shahana, S. Ghosh, and C. Jeganathan, "A Survey of Particle Swarm Optimization and Random Forest based Land Cover Classification," International Conference on Computing, Communication and Automation (ICCCA2016), pp. 241-2452016, doi: 10.1109/CCAA.2016.7813756.

[9] S. Arora, and S. Singh, "The Firefly Optimization Algorithm: Convergence Analysis and Parameter Selection," International Journal of Computer Applications, 69(3), pp. 48-52, 2013.

[10] L. Wenjie, L. Jiazheng, L. Yifan, S. Aijun, and W. Jingyang, " A CNN-LSTM-Based Model to Forecast Stock Prices," Complexity, 2020, pp. 1076-2787, 2020, doi: 10.1155/2020/6622927.

[11] S. Chen, and H. He, "Stock Prediction Using Convolutional Neural Network," IOP Conference Series: Materials Science and Engineering, 435, 2018, doi:10.1088/1757-899X/435/1/012026.

[12] J. Wu, X. Chen, H. Zhang, L. Xiong, and H. Lei, S. Deng, "Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization," Journal of Electronic Science and Technology, 7(1), pp. 26-40, 2019, doi:10.11989/JEST.1674-862X.80904120.

[13] J. Bergstra, and Y. Bengio, "Random Search for Hyper-Parameter Optimization," Journal of Machine Learning Research, 13, pp. 281-305, 2012.

[14] R. Andonie, and A.C. Florea, "Weighted Random Search for CNN Hyperparameter Optimization," International Journal of Computers Communications & Control, 5(2), 2020, doi: 10.15837/ijccc.2020.2.3868.

[15] F. M. Talaat, and S. A. Gamel, "RL based hyper-parameters optimization algorithm (ROA) for convolutional neural network," Journal of Ambient Intelligence and Humanized Computing, 2022, doi: 10.1007/s12652-022-03788-y.

[16] Y. Cai, and A. Sharma, "Swarm Intelligence Optimization: An Exploration and Application of Machine Learning Technology," Journal of Intelligent Systems, 30(1), pp. 460-469, 2021, doi: 10.1515/jisys-2020-0084.

[17] F. Lobato, and J. V. Steffen, "Fish Swarm Optimization Algorithm Applied to Engineering System Design," Latin American Journal of Solids and Structures, 11, pp. 143-156, 2014, doi: 10.1590/S1679-78252014000100009.

[18] M. Ahmad, A. A. Ikram, R. Lela, I. Wahid, and R. Ulla, "Honey bee algorithm–based efficient cluster formation and optimization scheme in mobile ad hoc networks," International Journal of Distributed Sensor Networks, 13(6), 2017, doi:10.1177/1550147717716815.

[19] Z. Fouad, M. Alfonse, M. Roushdy, A. M. Salem, "Hyper-parameter optimization of convolutional neural network based on particle swarm optimization algorithm", Bulletin of Electrical Engineering and Informatics, 10(6), 2021, pp. 3377-3384, doi: 10.11591/eei.v10i6.3257.

[20] S. Dhawan, "Optimization Performance Analysis of Firefly Algorithm using Standard Benchmark Functions," International Journal of Engineering Research & Technology, 11(2), pp.385-392, 2022, doi: 10.17577/IJERTV11IS020165.

[21] X. Yang and A. H. Gandomi, "Bat Algorithm: A Novel Approach for Global Engineering Optimization," Engineering Computations, 29(5), pp. 464-483, 2012.

[22] Yahoo Finance - Business Finance Stock Market News, <https://in.finance.yahoo.com/> [Accessed on October 01,2022].

[23] E. Elgeldawi, A. Sayed, A. R. Galal , and A. M. Zaki, "Hyperparameter Tuning for Machine Learning Algorithms Used for Arabic Sentiment Analysis," Informatics, 2021, 8(4), pp. 79-99, 2021, doi:10.3390/informatics8040079.

[24] B. Qolomany, M. Maabreh, A. Al-Fuqaha, A. Gupta, and D. Benhaddou, "Parameters optimization of deep learning models using Particle swarm optimization," 2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC), pp. 1285-1290, 2017, doi: 10.1109/IWCMC.2017.7986470.

[25] D. Wang, D. Tan, and L. Liu, "Particle swarm optimization algorithm: an overview," Soft Comput, 22, pp.387-408, 2018, doi: 10.1007/s00500-016-2474-6.

[26] A. G. Gad, "Particle Swarm Optimization Algorithm and Its Applications: A Systematic Review," Archives of Computational Methods in Engineering, 29, pp. 2531–2561, 2022, doi: 10.1007/s11831-021-09694-4.

[27] J. Kennedy and R. Eberhart, "Particle swarm optimization," Proceedings of ICNN'95 - International Conference on Neural Networks, 4, pp. 1942-1948, 1995, doi: 10.1109/ICNN.1995.488968.

[28] A. Abdulhadi, "Application of the Firefly Algorithm for Optimal Production and Demand Forecasting at Selected Industrial Plant," Open Journal of Business and Management, 08, pp. 2451-2459, 2022, doi:10.4236/ojbm.2020.86151.

[29] A. Sharma, A. Zaidi, R. Singh, S. Jain, and A. Sahoo, "Optimization of SVM classifier using Firefly algorithm," 2013 IEEE Second International Conference on Image Information Processing (ICIIP-2013), pp. 198-202, 2013, doi: 10.1109/ICIIP.2013.6707582.

[30] M. Farrell, K. N. Ramadhani, and S. Suyanto, "Combined Firefly Algorithm-Random Forest to Classify Autistic Spectrum Disorders," 2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), pp. 505-508, 2020, doi: 10.1109/ISRITI51436.2020.9315396.

[31] [31] XS. Yang, "Firefly Algorithms for Multimodal Optimization," O. Watanabe and T. Zeugmann (Eds.): SAGA 2009, Lecture Notes in Computer Science, 5792, pp. 169–178 doi: 10.1007/978-3-642-04944-6_14.

[32] L. Liberti, C. Lavor, N. Maculan, and A. Mucherino, "Euclidean Distance Geometry and Applications" SIAM Review. 56, 2012 doi: 10.1137/120875909.

[33] A. Botchkarev, "Evaluating performance of regression machine learning models using multiple error metrics in Azure Machine Learning Studio," SSRN Electronic Journal, 2018, doi: 10.2139/ssrn.3177507.