Bol. Soc. Paran. Mat.
©SPM – E-ISSN-2175-1188
SPM: www.spm.uem.br/bspm

(3s.) **v. 2026 (44) 17** : 1–26.
ISSN-0037-8712
doi:10.5269/bspm.82355

Adaptive Architecture and Advanced Optimization in Artificial Neural Networks for Financial Time Series Forecasting

Jaykumar Ishvarbhai Prajapati and Raja Das

ABSTRACT: It is challenging to predict what will happen in the stock market because financial data isn't always the same and changes quickly over time. However, accurate predictions are important for making smart investment decisions. This study presents a novel dynamic artificial neural network that adaptively adjusts the number of hidden-layer neurons based on the recent market conditions to enhance the prediction of stock opening and closing prices. The model uses the Fletcher-Reeves method for conjugate gradient optimization, the modified stochastic MCCV fold for validation, and the dynamic K-day sliding window for updating weights and biases over and over again. The proposed model outperformed the conventional artificial neural network along with different pairs of methods, achieving a higher correlation coefficient of 0.97292. The results highlight that the effectiveness of dynamic architectures and advanced training strategies can improve the accuracy of stock market forecasts.

Keywords: Stock price forecasting, artificial neural network, stock market, gradient descent method, optimization.

Contents

2020 *Mathematics Subject Classification:* 62M45, 91G15, 68T07.
Submitted March 03, 2026. Published June 19, 2026.

1. Introduction

The stock market is where investors trade in and out of company shares. It is an organization of exchanges where firms trade shares and other securities. The stock market provides tremendous prospects for investors to profit from long-term benefits or trade on the stock by purchasing a stock and becoming a stockholder. The emergence of the company, as well as other unforeseen national, global, and social changes, has an immediate detrimental or positive impact on stock prices. Many factors can influence stock market performance, whether beneficial or detrimental, including political events, general economic conditions, global crises, and so on. The prediction is an assertion about the future, and investors can decide whether or not to invest in the stock market based on it.

In contemporary times, a diverse range of machine learning (ML) models have been employed for the purpose of predicting short-term fluctuations in financial markets [25] and have achieved satisfactory outcomes. Machine learning challenges can be viewed as one approach to determining how to improve the performance of computer algorithms based on previous information. The researchers commonly used Artificial Neural Networks (ANNs) [18], genetic algorithms [6], and hidden markov models [3]. These techniques are promising and are becoming increasingly popular among researchers because they can effectively map highly non-linear input-output data sets, surpassing the capabilities of traditional statistical regression models. There are numerous varieties of ANN algorithms based on the prediction algorithm utilized, one of which is back-propagation. In general, the Back Propagation Neural Network (BPNN) is the most commonly employed. However, BPNN is concerned with over-fitting, a high number of parameters, and challenges achieving a stable solution [17]. To address the issues raised above, extensive research into improving ANNs is provided, including ANNs with genetic algorithms [22] and ANNs with metaheuristics [10,5]. The Fletcher-Reeves conjugate gradient method (FRM) utilizes information from previous iterations to compute a new conjugate direction instead of relying solely on the steepest descent direction used by standard BPNN to determine optimal weight and bias values. Additionally, it does not require storing a large Hessian matrix, which enhances efficiency during batch training, and it is also faster than the standard steepest descent method for non-linear convex optimization problems [7].

In recent years, machine learning (ML) approaches, especially artificial neural networks (ANNs) have shown considerable promise in capturing non-linear and hidden relationships within financial datasets. ANNs have been widely used to forecast short-term stock price fluctuations, benefiting from their ability to learn complex mappings between inputs and outputs [24,21,26]. Nevertheless, a major limitation in the current literature is the reliance on static ANN architectures [32,29], where the number of neurons in hidden layers remains fixed throughout training, regardless of the underlying data characteristics. Such rigid architectures may fail to adapt effectively to changes in market dynamics, which can lead to reduced forecasting accuracy and generalization. To address this gap, we propose a dynamic artificial neural network (DANN) framework for stock market prediction that adaptively adjusts the number of neurons in the hidden layer in response to changes in input data patterns. The training procedure integrates the FRM, modified stochastic Monte Carlo Cross-Validation (MCCV) fold, and a K-day sliding window strategy for dynamic updates of weights and biases. This combination allows the model to maintain flexibility and responsiveness over time, thereby improving its performance in non-stationary environments.

The primary objectives of this study are as follows:

- To determine the optimal number of nodes in the hidden layer of the ANN required to achieve maximum predictive accuracy.
- To develop a dynamic neural network model by integrating a dynamic sliding window technique with modified stochastic Monte Carlo Cross-Validation (MCCV) folds.
- To compare the proposed dynamic model with a conventional neural network, assessing its accuracy and reliability when applied to the characteristically non-linear nature of stock market data.
- To evaluate whether the dynamic model exhibits superior predictive capabilities over its conventional counterpart.

Here is the breakdown of the remaining sections of the paper: In Section 2, we essentially provide literature survey and the significant ANNs used in this investigation. Section 3 presents the experimental configuration and methodology. Section 4 presenting the results of every possible pairs of ANNs and statistical evaluation of the analyzed datasets derived for each of the examined models. Section 5 discusses the outcomes and implications of the experiment. The last section 6 concludes the future research scope.

2. Related Works

The body of work on multivariate time-series prediction can be classified into two main categories: (i) Traditional statistical approaches and (ii) Deep learning methods [27]. In Deep learning, specifically ANNs have been used for classification and regression problems due to their capacity to act as universal approximators to any linear as well as non-linear functions. There are numerous varieties of ANN algorithms based on the prediction algorithm utilised, one of which is back-propagation [13]. The back-propagation algorithm is an effective tool for solving prediction problems. Conjugate Gradient is an ANN-based Deep learning approach that can be used to deal with computational difficulties associated with forecasting data. In general, the gradient descent approach is used for optimization. Its effectiveness is influenced by the optimization method employed during training [1]. The disadvantage of this approach is its slow convergence rate. This is due to its faster completion time and fewer iterations. The oldest and most well-known gradient conjugation approach for solving non-linear infinite optimization problems is the Fletcher-Reeves method [8]. Fletcher and Reeves conjugate gradient algorithm [9] can usually forecast more efficiently than standard gradient descent-based algorithms [20], whereas Polak-Ribiére [12] is type of a gradient conjugation algorithm for solving infinite monotone non-linear equations.

Table 1: Validation of the proposed method by comparison with existing methods.

Study	**Method Type**	**Forecasting Parameter**	**Accuracy**
Patel et al. [25]	ANN, SVR, Random Forest and their combinations	CNX Nifty index and S&P BSE Sensex index	SVR–ANN, SVR–RF model give high accuracy than standard ANN, SVR models.
Sahoo and Mohanty [28]	ANN, ANN and GWO	Prediction of stock & market pricing 1 day ahead,& 15 days ahead, and 30 days ahead.	Best RMSE value 0.5133
Gülhan Toga et al. [30]	ARIMA, ANN	COVID-19 prevalence forecasting in Turkey.	ARIMA has higher accuracy than ANN, but cannot estimate multiple outputs simultaneously.
Binita Kumari and Tripti Swarnkar [16]	ANN, SVM, and KNN	Forecasting daily stock price movement of HANG SENG, NIFTY50, NASDAQ, etc.	ANN-generated models outperformed SVM and KNN models.
Fathe Jeribi et al. [15]	CNN, LSTM, SMV, hybrid DRL-ANN	Forecasting S&P500-S, S&P500-L, and DAX indices prices	99.562%, 98.235%, and 98.825% respectively.
B. Manohar et al. [19]	ANN-LM, ANN-GDX	Forecasting daily prices of Reliance, TCS.	R^2 values: 0.9533, 0.9889 (ANN-LM); 0.9261, 0.9601 (ANN-GDX).
Proposed work	DANN model (ANN-FRM, stochastic MCCV fold & K-sliding window)	Forecasting opening and closing prices of Reliance, Maruti, Tata Motors, and HCL for next 10 days.	r values : 0.9713 (Reliance), 0.9603 (Tata), 0.9726 (Maruti), 0.9729 (HCL Technologies)

The following characteristics of financial forecasting can be applied: it is noisy, unstructured, non-stationary, and involves hidden linkages [23,4,14]. ANN is an algorithm created to comprehend complex problems that are beyond the scope of accessible neural networks or simple machine learning algorithms [2]. To provide reliable forecasting data, it is crucial to understand an algorithm's performance, particularly the conjugate gradient method. Numerous researches have been done, including Wanto et al. [31], who used FRM and the back-propagation method to solve the challenge of predicting the consumer price index. In this investigation, the back-propagation algorithm had a prediction accuracy advantage of 75% over 67%, whereas the FRM approach performed significantly better in terms of performance, MSE, and time. More literature review related with ANNs and other deep learning methods mentioned in Table 1.

3. Methodology

3.1. Fletcher Reeves Method (FRM)

Let us define the unconstrained convex problem $min\{L(x)|x \in R^n\}$, where $L : R^n \to R^1$ and $L \in C^1$. The conventional approach of the conjugate gradient method involves the minimization of the differentiable loss function L by the iterative formation of a succession of approximations x_{i+1} as,

$$x_{i+1} = x_i + \alpha_i d_i, \tag{3.1}$$

where α_i is the step length or learning rate in neural network terms. By using line search methods, it is possible to find the step length α_i such that $L(w_i + \alpha_i d_i)$ that is minimized along the direction d_i , given x_i and d_i fixed.

$$d_1 = - \bigtriangledown L(x_1) = -g_1, \tag{3.2}$$

$$d_i = -g_i + \beta_{i-1} d_{i-1} \qquad (i = 2, 3, 4, \dots), \tag{3.3}$$

where g denotes the gradient $\bigtriangledown L$ of L. The scalar β_i is to be found based on the criterion that d_i and d_{i-1} must satisfy the conjugate property. Powell demonstrated the global convergence of the Fletcher-Reeves algorithm (FRM) with exact linesearch in 1983. In 1985, Al-Baali [1] demonstrated the global convergence of FRM under comparable conditions using a modification of the wolfe line search that incorporates inexact line search as a special case. The conventional conjugate gradient technique commences the process of minimizing by using an initial estimate x_0 and an initial search direction d_0. There exist numerous formulae that can be utilized to calculate the parameters β_i, where β_i is given by,

$$\beta_{i-1} = \frac{g_i^T g_i}{g_{i-1}^T g_{i-1}} \quad \text{(Fletcher-Reeves formula)}. \tag{3.4}$$

The global convergence of the FRM with a low-accuracy, inexact linesearch is proven by L Guanghui et al [11]. The interested readers can refer the appendix A.1 for the global convergence of the FRM.

3.2. Data Description

In this research, to ensure DANN is reliable and valid, we are collecting daily datasets from 11-1-2022 to 11-1-2023 with six parameters (opening price, closing price, volume, adjusted closing price, day high, and day low) from the Yahoo Finance India website for four distinct companies across diverse industries, including automobile, refinery, and IT sectors: Tata Motors, Maruti Suzuki, Reliance, and HCL Technologies, all of which are listed on the National Stock Exchange (NSE) in India.

3.3. Normalization

Normalization refers to a collection of strategies that are employed to process data, particularly when the data originates from varying scales. Otherwise, it may result in errors in evaluation or conclusions that are not suitable. So, for that we are using Min-Max normalization. This technique involves the process of normalizing data by fitting it within a predetermined boundary, specifically within a predefined period. Which involves rescaling the values of numeric variables within the range of 0 to 1. The formula of Min-Max normalization is given by

$$x_{norm} = \frac{(1-(0))\,(x - x_{min})}{x_{max} - x_{min}} + (0) \text{ and so, } x_{norm} = \frac{(x - x_{min})}{x_{max} - x_{min}},$$

where $x_{max} =$ Maximum of x among dataset, x_{min} = Minimum of x among dataset. All elements of each row are considered to be non-equal, and x has only finite real values. (If $x_{max} = x_{min}$ or if x_{max} or x_{min} are not finite, then $x_{norm} = x$.)

3.4. Artificial Neural Networks (ANNs)

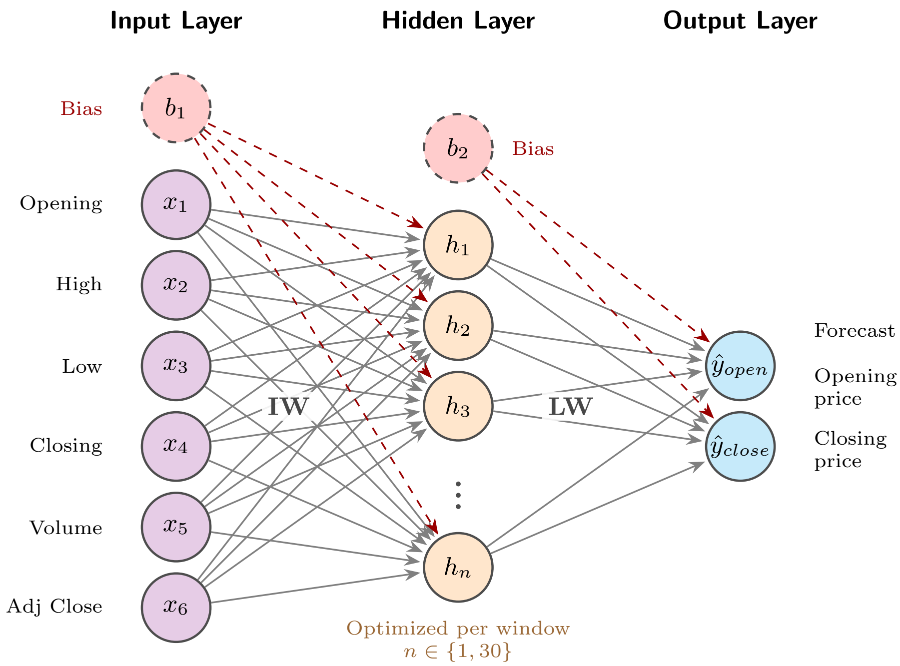


Figure 1: Architecture of the Proposed dynamic hidden layer of ANN. The network accepts 6 input features.

ANNs have many benefits, such as not needing as much formal statistical training, being able to find all possible interactions between predictor variables, being able to find complex non-linear relationships between dependent and independent variables, and having access to a number of different training algorithms. ANNs take the form of interconnected layers of artificial neurons. The connection between the input layer of neurons and the output layers of neurons is facilitated by the presence of one or more hidden layers of neurons. This study uses the least mean squared error (MSE) value to identify the hidden layer neurons of the standard neural network. In the hidden layer, 1 to 30 neurons are analyzed; the optimal number of neurons that yields the highest accuracy is selected for the final ANN model. We employ the error backpropagation algorithm to train an ANN. For model learning and find the optimal weight and bias values, the FRM algorithm serves as the optimization algorithm. We evaluate the ANN model's performance using a validation dataset to ensure its excellent generalization to unseen data. By systematically adjusting the number of hidden layer neurons and monitoring the MSE, we aim to find the most efficient architecture for accurate predictions. We first train and evaluate an ANN using experimental data to determine the ideal topology and weights by using the inputs and the activation and connectivity weights of different nodes to calculate the output of the ANN. We determine the MSE by comparing the predicted output with actual one. The sigmoid function is used as an activation function for the hidden layer, whereas linear activation function is used for the output layer. Figure 1 representing schematic ANN figure. The weight and bias connections between the input layer and hidden layer are shown by vectors $[IW]_{n\times 6}$ and $[b_1]_{n\times 1}$, respectively. The weight and bias connections between the hidden layer and output layer are shown by the vectors $[LW]_{2\times n}$ and $[b_2]_{2\times 1}$, respectively. The steps involved in ANN's training after normalization of the data and how the weight values are ultimately determined as follows:

$$
\begin{aligned}
\hat{Y}_{2\times 1} &= LW_{2\times n}\cdot\sigma(IW_{n\times 6}X_{6\times 1}+b_{1,n\times 1})+b_{2,2\times 1},\\
\text{where } \sigma(z) &= \frac{1}{1+e^{-z}},\quad z\in\mathbb{R}\to(0,1).\\
X &= [x_1,x_2,x_3,x_4,x_5,x_6]^T \text{ (where X=[Open, High, Low, Close, Adj. Close, Volume]}^T\text{)}\\
IW &= \begin{bmatrix} iw_{11} & \cdots & iw_{16}\\ \vdots & \ddots & \vdots\\ iw_{n1} & \cdots & iw_{n6}\end{bmatrix}_{n\times 6},\quad LW=\begin{bmatrix} lw_{11} & \cdots & lw_{1n}\\ lw_{21} & \cdots & lw_{2n}\end{bmatrix}_{2\times n},\\
b_1 &= [b_{11},\ldots,b_{1n}]^T,\quad b_2=[b_{21},b_{22}]^T.
\end{aligned}
$$

3.5. Stopping Criteria

A stopping criterion should aim to halt an ANN's learning process before it begins to overfit the exemplars. One criterion that may be monitored during ANN training is the error sum of squares, or, in other words, a loss function. Using this criterion, the error is computed between the desired output and the ANN output. Which can be defined as

$$\text{Loss function } (L) = \frac{1}{N}\sum_{i=1}^{N}|y(i)-\hat{y}(i)|^2, \tag{*}$$

where $y(i)$ is the actual value, $\hat{y}(i)$ is the predicted value and N is the number of observations. We keep an eye on validation performance, and if it stagnates or gets worse over the sequence of six iterations, we halt the training. This prevents over-fitting and ensures that the model maintains its generalization ability. By monitoring these metrics closely, we can make informed adjustments to our training strategy, ultimately leading to better performance on unseen data.

3.6. Multivariate Sliding Window Framework with ANN

To accurately capture the temporal interdependence and non-stationary characteristics of financial time series, we propose a dynamic rolling window methodology. In each iteration, the ANN is trained on the 6-dimensional dataset within this window to capture the complex inter-feature correlations. The model then forecasts the opening and closing prices for the subsequent 10-day horizon. Following each forecast, the window is shift by 's' days, and the model undergoes a complete re-training phase on the new data of K-days window. This approach ensures that the model dynamically adapts to non-stationary market conditions using the full spectrum of available information. Figure 2 representing the sliding window mechanism of size K. Let the multivariate time series represented by $\mathbf{X}=\{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_T\}$, where each $\mathbf{x}_t\in\mathbb{R}^6$ denotes the feature vector at time t, comprising the daily opening, high, low, closing, volume, and adjusted closing prices. For each sliding window with index i, the local dataset is defined as a matrix $\mathcal{D}^{(i)}$ of dimension $K\times 6$, and defined as:

$$\mathcal{D}^{(i)}=\{\mathbf{x}_{1+(i-1)s},\ldots,\mathbf{x}_{K+(i-1)s}\}, \tag{3.5}$$

where K represents the window size of K-days and $s=10$ denotes the rolling step size.

An ANN model is trained on the dataset $\mathcal{D}^{(i)}$ to learn the latent correlations among the six input features. After training, the model forecasts the target variables: opening and closing prices, for the subsequent 10-day horizon. Specifically, the predicted output vector $\hat{\mathbf{Y}}^{(i)}$ corresponds to the interval:

$$\hat{\mathbf{Y}}^{(i)}\approx\{\mathbf{y}_{K+(i-1)s+1},\ldots,\mathbf{y}_{K+(i-1)s+h}\}, \tag{3.6}$$

where $\mathbf{y}_t$ contains the target feature vector and $h=10$ is the forecast horizon. Post-training, we assess the model's performance over a forecast horizon of $h=10$ days using a static inference strategy. Unlike

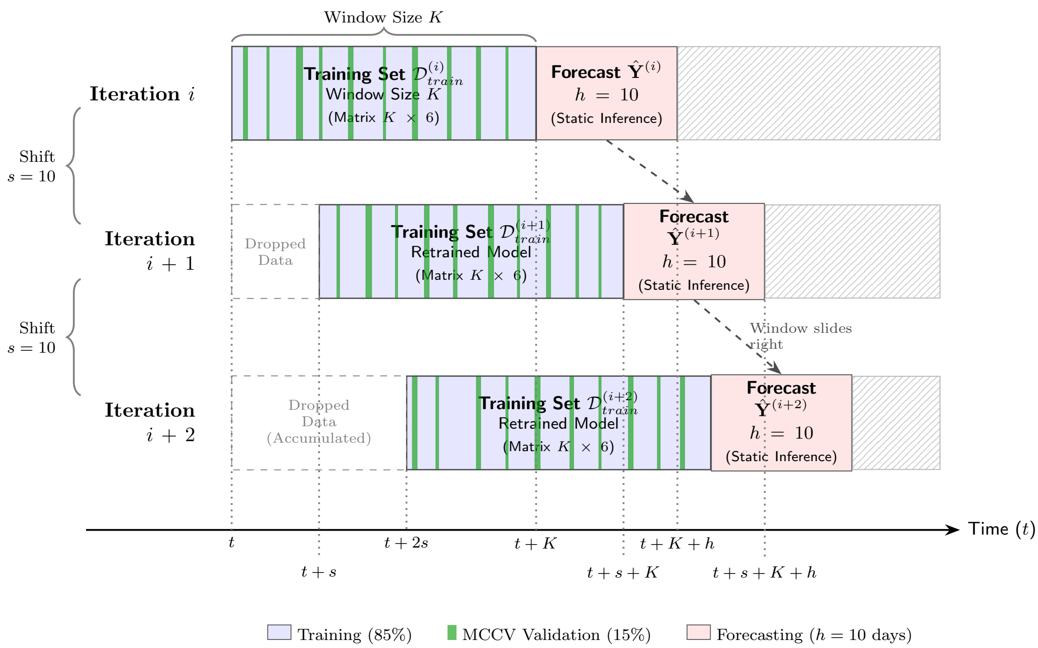


Figure 2: Architecture of the Proposed dynamic hidden layer of ANN. The network accepts 6 input features.

multi-step forecasting, we perform iterative one-step-ahead predictions using optimized weights and bias values obtained from training the ANN with the corresponding window shift. For each time step j within the horizon ($1 \leq j \leq h$), the model predicts the opening and closing prices for day $t + j$ using the actual observed feature vector from the previous day $t + j - 1$:

$$\hat{\mathbf{y}}_{t+j} = \mathcal{M}(\mathbf{x}_{t+j-1}; \theta^{(i)}), \tag{3.7}$$

where $\theta^{(i)}$ represents the fixed model parameters for the current window i, and $\mathbf{x}_{t+j-1}$ is the actual market data. This process is repeated 10 times. Once the 10-day's opening and closing price forecast is completed, the window slides forward by $s = 10$ days, and the model is fully retrained. After the 10-day forecast of the opening and closing prices is done, the window moves forward by $s = 10$ days, and the model is completely retrained. After that, the window moves by $s = 10$ days, and the model is reset and retrained on the new feature matrix $\mathcal{D}_{train}^{(i+1)}$ in order to continue up with the market's shifts.

3.7. Optimized Architecture Search via Stochastic Monte Carlo Cross-Validation (MCCV)

When using neural networks with sliding windows, there is a common problem: the model is likely to match local noisy data too well instead of capturing the underlying pattern. Standard validation techniques, which usually keep the last chronological segment for testing, are often insufficient in such an environment as they assess performance on a singular, fixed interval of time. To address this, we implemented a rigorous internal validation strategy based on Monte Carlo Cross-Validation (MCCV). For each sliding window with index i, we treat the local dataset $\mathcal{D}^{(i)}$ as a distinct regime. Instead of using fixed partitions, we created $M = 10$ independent stochastic folds. For each fold, the data within the i-th index window is randomly subsampled into two distinct sets: a training set $\mathcal{T}^{(i)}$ (85%) and a validation set $\mathcal{V}^{(i)}$ (15%) as shown in Figure 2 with green color partitions. For each sliding window index i,

$$\mathcal{D}^{(i)} = \mathcal{T}^{(i)} \cup \mathcal{V}^{(i)}, \quad \text{such that} \quad \mathcal{T}^{(i)} \cap \mathcal{V}^{(i)} = \emptyset. \tag{3.8}$$

This randomized folding technique makes sures that, the model is tested against several different patterns of the local regime. In this framework, we systematically identify the optimal network topology through analyzing hidden layer sizes ranging from 1 to 30 neurons. Rather than average the validation error across all folds, which tends to smooth out performance variations, we want to identify the architecture that can provide the highest accuracy. As a result, we are looking for the best fold with the minimum Mean Squared Error (MSE) across the M stochastic folds. So, the optimal number of neurons in the hidden layer, n^*, is selected by minimizing this best-case validation loss as:

$$n^* = \arg\min_{n\in\Omega_n} \left(\min_{k\in\{1,\dots,M\}} \mathcal{L}_{val}\left(\mathcal{M}_n(\mathcal{T}_k^{(i)}), \mathcal{V}_k^{(i)}\right)\right). \tag{3.9}$$

To further control the complexity of the model, we included an early stopping criterion. However, if the validation mean squared error does not reduce after $\delta = 6$ over subsequent iterations, we terminate the validation. This assures that the chosen architecture has true forecasting potential rather than being overfit.

3.8. Statistical Metrics for Assessing Performance

To assess the model's goodness-of-fit, the following metrics were used: Mean Square Error (MSE), Mean Percentage Error (MPE), Mean Absolute Error (MAE) and the Pearson's correlation coefficient (r) defined as follows:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N} \|y(i) - \hat{y}(i)\|^2 \tag{3.10}$$

$$\text{MPE} = \frac{100\%}{N}\sum_{i=1}^{N} \frac{y(i) - \hat{y}(i)}{y(i)} \tag{3.11}$$

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N} |y(i) - \hat{y}(i)| \tag{3.12}$$

$$r = \frac{\sum_{i=1}^{N}(y(i) - \bar{y})(\hat{y}(i) - \bar{\hat{y}})}{\sqrt{\sum_{i=1}^{N}(y(i) - \bar{y})^2}\sqrt{\sum_{i=1}^{N}(\hat{y}(i) - \bar{\hat{y}})^2}}, \tag{3.13}$$

where, $y(i)$ = Actual values, $\hat{y}(i)$ = Predicted values, $\bar{y}$ = Mean of actual values, $\bar{\hat{y}}$ = Mean of predicted values, and N = Number of observations.

The models' goodness-of-fit, which quantifies the disparity between observed and predicted values, is commonly assessed using these three metrics. The inaccuracy in the model will be dropped when the values of these three indicators drop. MSE measure quantifies the average differences between the calculated and true values. One advantage of this metric is that its magnitude is comparable to that of the original data. MAE, quantifies the average difference between the predicted and actual values in a model's fitting. MPE is adjusted to be expressed as a percentage rather than using the same units as the variable being measured, employing this comparative measure. The main advantage of utilizing MPE is its capacity to evaluate discrepancies between data sets across many scales. To assess the accuracy of the model, it takes into consideration not only the gap between the predicted and actual values but also the proportion of this mismatch to the actual values.

4. Results

4.1. Comparison of (i) ANN without Both Stochastic MCCV Fold and K-Slide Window,(ii) ANN with Stochastic MCCV Fold and without K-Sliding Window, (iii) ANN with K-Sliding Window and without Stochastic MCCV Fold, and (iv) ANN with Both K-Sliding Window and Stochastic MCCV Fold

We present every possible pair of ANN models' statistical metrics values in context of the stock HCL Technologies' opening and closing price forecasting. These include ANN without both stochastic MCCV fold and K-window size in Table 2, ANN with stochastic MCCV fold and no K-window size in Table 3, ANN with K-window size and no stochastic MCCV fold in Table 4, and lastly for proposed DANN model

(ANN with both K-window size and stochastic MCCV fold using FRM optimizer) in Table 5. When the testing data for HCL Technologies' opening price is within the validation data's range or follows a similar pattern, we discovered that, using only an ANN model (Figure 3(a)), the predicted price comes within that range. ANN using the stochastic MCCV fold technique in Figure 3(b) reduces the gaps between real and forecasted data compared to an ANN prediction alone. Figure 3(c) shows that ANN with K-window size gives predictions that are nearly equal to the real data, with just minor variations. Finally, when compared to all of the other combinations illustrated in Figure 3, ANN paired with the stochastic MCCV fold and K-window size (Figure 3(d)) yields the optimum level of prediction. Similarly, we are obtaining results for the remaining stocks.

Table 2: Statistical metrics for Baseline ANN (without stochastic MCCV fold and sliding window)

Window size (K)	**stochastic MCCV fold**	**Open**				**Close**			
		MSE	**MAE**	**MPE**	**r**	**MSE**	**MAE**	**MPE**	**r**
No	No	13071.73	91.22	0.0957	-0.0790	3876.45	52.90	0.0548	0.8676

Table 3: Statistical metrics for ANN with stochastic MCCV fold (without sliding window)

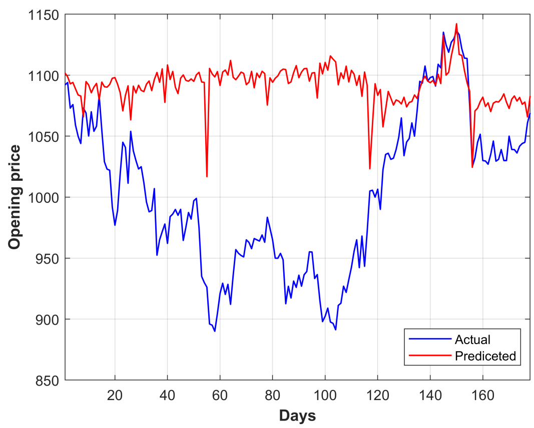
Window size (K)	**stochastic MCCV fold**	**Open**				**Close**			
		MSE	**MAE**	**MPE**	**r**	**MSE**	**MAE**	**MPE**	**r**
No	Yes	2850.81	36.97	0.0376	0.6605	8099.56	70.20	0.0734	0.0613

Table 4: Statistical metrics for ANN with sliding window (without stochastic stochastic MCCV fold)

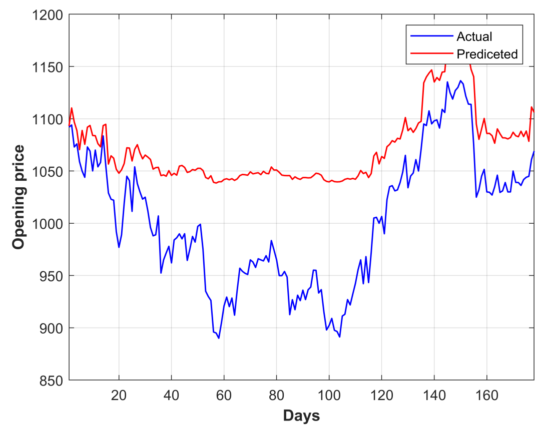
Window size (K)	**stochastic MCCV fold**	**Open**				**Close**			
		MSE	**MAE**	**MPE**	**r**	**MSE**	**MAE**	**MPE**	**r**
30	No	714.05	17.55	0.0172	0.9489	1211.84	23.72	0.0233	0.9155
40	No	671.01	17.86	0.0176	0.9503	789.66	20.42	0.0202	0.9408
50	No	873.02	19.24	0.0190	0.9187	1156.55	23.44	0.0232	0.8972
60	No	581.72	16.18	0.0159	0.9329	829.55	21.68	0.0217	0.9080
70	No	412.77	14.57	0.0144	0.9492	478.09	15.73	0.0160	0.9437
80	No	730.22	18.03	0.0180	0.9055	757.25	18.93	0.0192	0.9096

Table 5: Statistical metrics for proposed DANN model(ANN with both sliding window and stochastic MCCV fold)

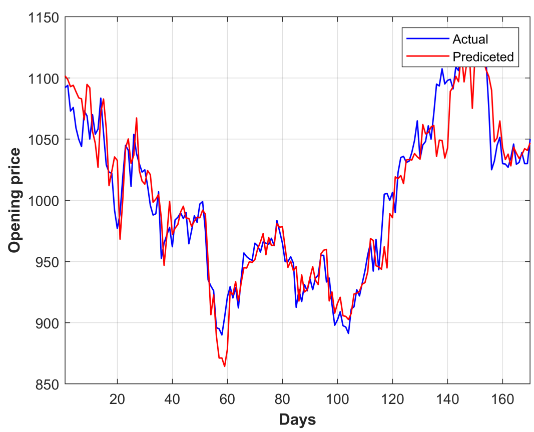
Window size (K)	**stochastic MCCV fold**	**Open**				**Close**			
		MSE	**MAE**	**MPE**	**r**	**MSE**	**MAE**	**MPE**	**r**
30	Yes	865.72	19.85	0.0193	0.9395	1195.71	23.70	0.0233	0.9160
40	Yes	474.87	15.68	0.0154	0.9662	717.16	19.49	0.0193	0.9460
50	Yes	435.12	14.76	0.0147	0.9640	722.66	20.06	0.0198	0.9390
60	Yes	329.11	13.49	0.0134	0.9643	649.10	18.80	0.0187	0.9254
70	Yes	239.97	11.89	0.0119	0.9729	574.59	17.45	0.0175	0.9315
80	Yes	420.43	14.83	0.0150	0.9499	859.26	19.67	0.0200	0.8905



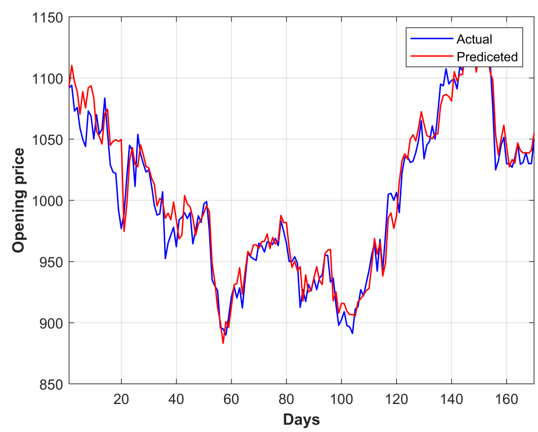
(a) Only ANN



(b) ANN with stochastic MCCV fold



(c) ANN with sliding window K=70 size

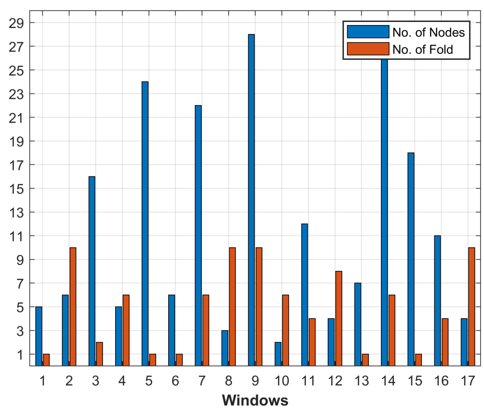


(d) ANN with stochastic MCCV fold and sliding window K=70 size (Proposed DANN model)

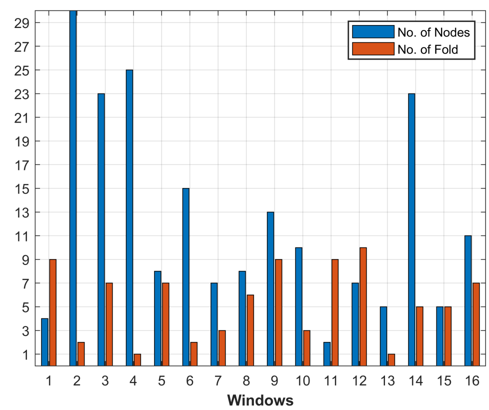
Figure 3: (a-d) Forecasting HCL Technologies stock's opening price for the next 10 days using different combinations with ANN.

4.2. Results of Experiments Employing DANN (ANN with the FRM, Stochastic MCCV Fold, and Optimal K-size Sliding Window)

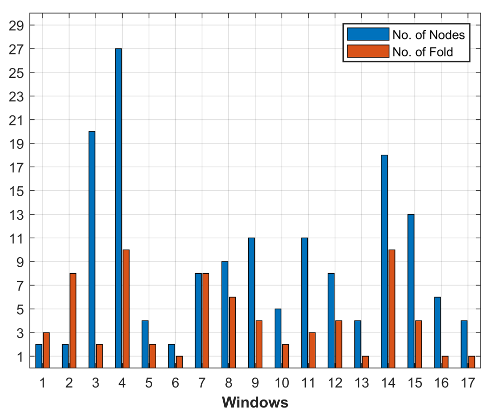
To forecast the opening and closing prices using the ideal K-sliding window size for the four stocks previously described, we first examine each sliding window of size K=30, 40, 50, 60, 70, and 80 days. Table 6 shows the MSE, MAE, MPE, and r values for opening and closing price predictions for the four stocks described earlier, with K-window sizes of 30, 40, 50, 60, 70, and 80 days. Figures 4 and 5 show how the number of neurons in the hidden layer changes dynamically when the window size for predicting opening and closing prices changes. Figures 6 and 7 show a graphical representation of the disparities between the actual and predicted values of the forecasted opening and closing prices for Reliance, Tata Motors, Maruti, and HCL Technologies, using DANN with optimal K-sliding window sizes for respective stocks on the MATLAB platform.



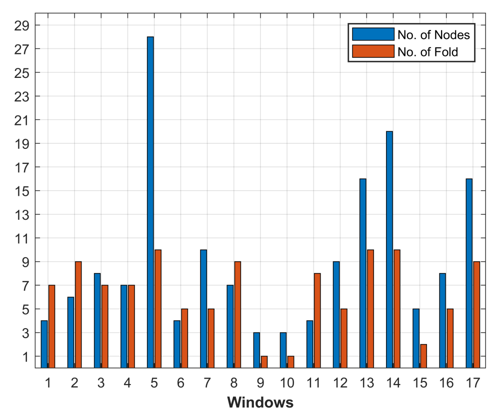
(a) Reliance ($K = 70$)



(b) Tata Motors ($K = 80$)



(c) Maruti ($K = 70$)

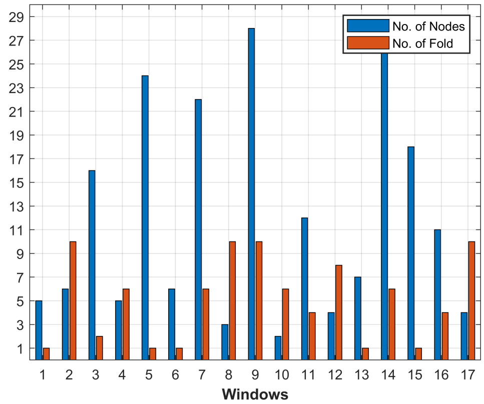


(d) HCL Technologies ($K = 70$)

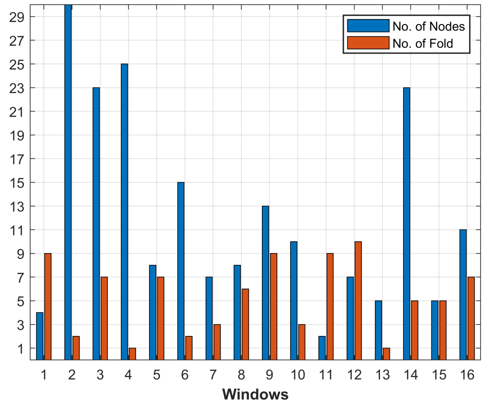
Figure 4: (a-d) Dynamic change in the number of nodes in hidden layer for each slide window shift to forecast opening price for stocks using DANN and K = (a) 70, (b) 80, (c) 70, and (d) 70 days slide window size respectively.

Table 6: Statistical metrics for DANN (ANN with K-window size and stochastic MCCV fold) assessing performance for each stock for each window size

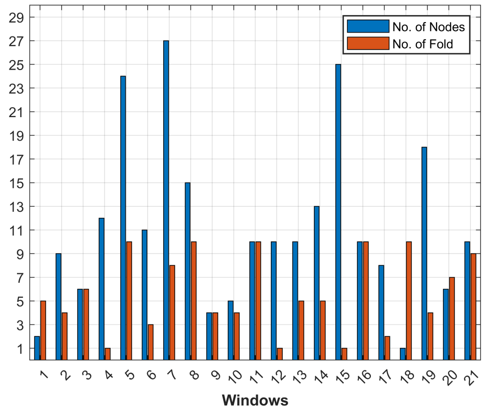
		Open				Close			
Window Size	**Stock**	**MSE**	**MAE**	**MPE**	**r**	**MSE**	**MAE**	**MPE**	**r**
K=30	Reliance	2732.63019	34.190716	0.013830	0.891402	2949.72837	40.650125	0.016625	0.897346
	Tata Motors	234.022735	10.248157	0.024362	0.776916	420.510044	13.760893	0.032791	0.643310
	Maruti	19867.50240	97.844687	0.011287	0.961906	25256.88830	118.227160	0.013715	0.91636
	HCL Tech.	865.718654	19.846115	0.019335	0.939493	1195.710661	23.697280	0.023249	0.916035
K=40	Reliance	2608.02273	33.804131	0.013823	0.894920	3508.93212	44.384904	0.018251	0.866151
	Tata Motors	116.234883	7.442751	0.017579	0.889021	153.980307	9.563185	0.022571	0.848137
	Maruti	23953.48870	101.303018	0.012133	0.969036	43360.69800	149.715841	0.017716	0.942819
	HCL Tech.	474.869438	15.676381	0.015446	0.966223	717.162721	19.494314	0.019284	0.946047
K=50	Reliance	1596.91397	23.902368	0.009581	0.936891	3410.76621	42.259487	0.017235	0.868491
	Tata Motors	82.622839	6.156303	0.014657	0.922139	176.144743	9.479008	0.022433	0.835568
	Maruti	28401.04410	102.255360	0.012112	0.961746	48417.38830	150.173883	0.017747	0.930141
	HCL Tech.	435.120315	14.756358	0.014675	0.963965	722.657967	20.057218	0.019835	0.939018
K=60	Reliance	1471.89751	25.264507	0.010477	0.944713	2645.55821	36.587501	0.014968	0.910733
	Tata Motors	54.654184	5.698255	0.013273	0.949479	126.468426	8.789296	0.020670	0.885921
	Maruti	41238.95430	138.727695	0.016454	0.936127	49513.38030	166.619733	0.019843	0.927973
	HCL Tech.	329.105628	13.489826	0.013422	0.964251	649.101456	18.795343	0.018739	0.925421
K=70	Reliance	846.141368	20.118857	0.008205	0.971379	1846.76556	32.753197	0.013339	0.934005
	Tata Motors	50.653277	4.904788	0.011535	0.956796	137.614347	8.608965	0.020352	0.878553
	Maruti	16814.63750	93.323394	0.010933	0.972696	35598.36540	137.558689	0.016185	0.940549
	HCL Tech.	239.965521	11.887597	0.011890	0.972921	574.587406	17.453346	0.017465	0.931509
K=80	Reliance	2214.46188	28.270894	0.011500	0.933483	4694.84362	41.881494	0.016939	0.834560
	Tata Motors	48.232117	5.105400	0.011974	0.960315	88.785585	7.160328	0.016890	0.925765
	Maruti	18952.50430	95.568823	0.011139	0.962655	36482.51380	136.189720	0.015896	0.930747
	HCL Tech.	420.433237	14.828837	0.014991	0.949865	859.261094	19.667739	0.019956	0.890490



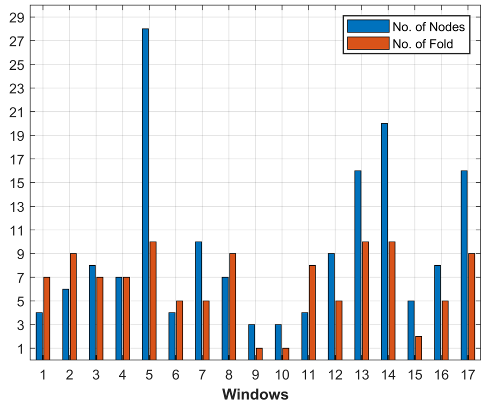
(a) Reliance (K=70)



(b) Tata Motors (K=80)

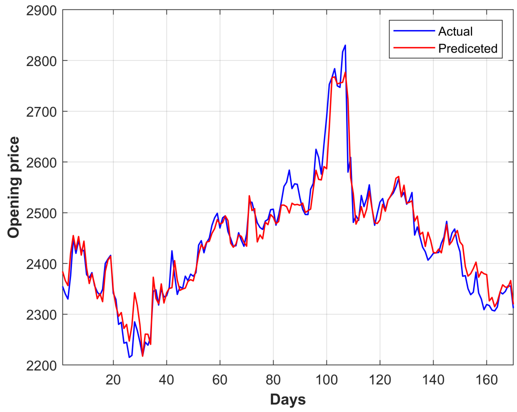


(c) Maruti (K=30)

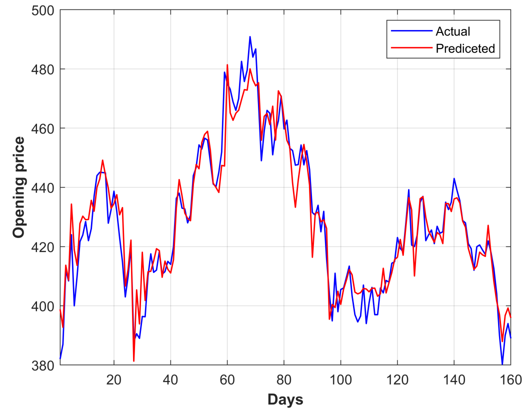


(d) HCL Technologies (K=70)

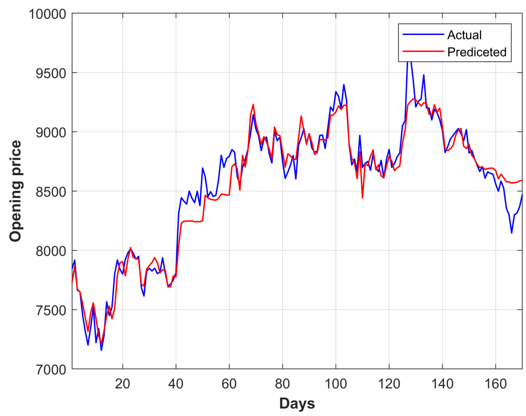
Figure 5: (a-d) Dynamic change in the number of nodes in hidden layer for each slide window shift to forecast closing price for stocks using DANN and K = (a) 70, (b) 80, (c) 30, and (d) 70 days slide window size respectively.



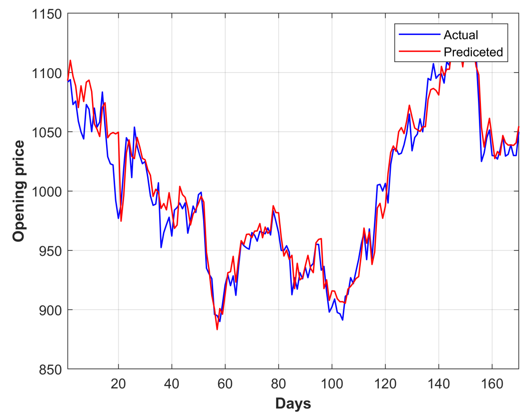
(a) Reliance



(b) Tata Motors

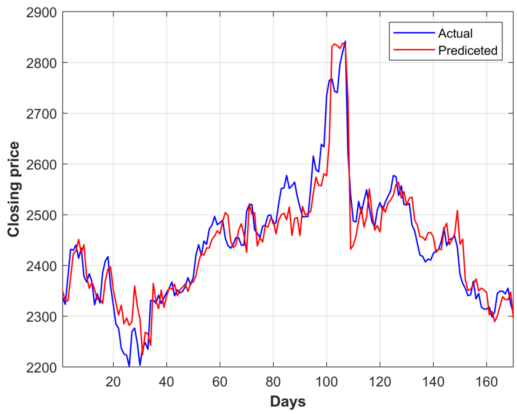


(c) Maruti

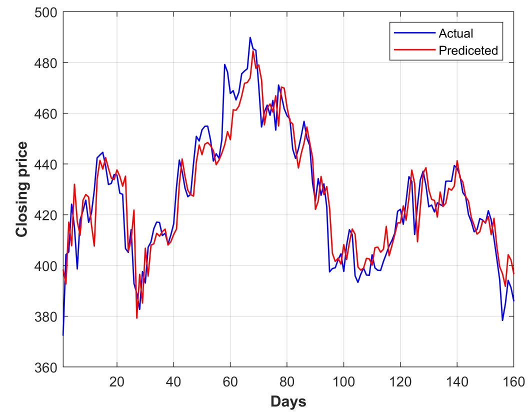


(d) HCL Technologies

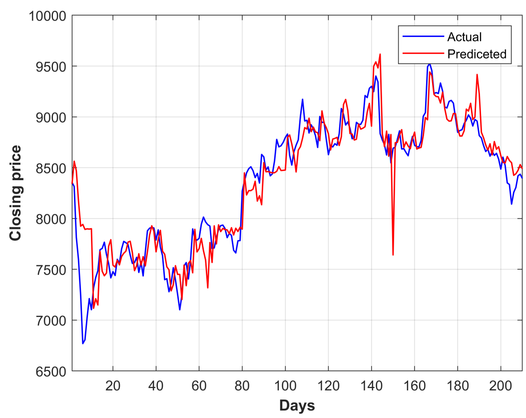
Figure 6: (a-d) Actual versus predicted forecasting opening price for stocks using DANN and K = (a) 70, (b) 80, (c) 70, and (d) 70 days slide window size.



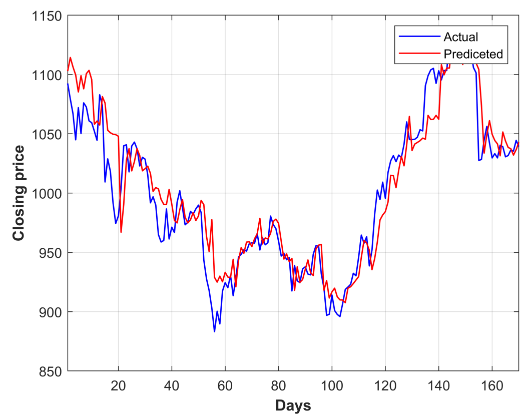
(a) Reliance



(b) Tata Motors



(c) Maruti



(d) HCL Technologies

Figure 7: (a-d) Actual versus predicted forecasting closing price for stocks using DANN and $K =$ (a) 70, (b) 80, (c) 30, and (d) 70 days slide window size.

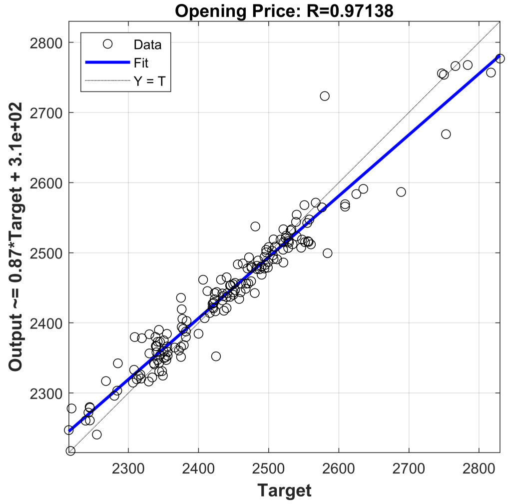
5. Discussions

When validation data for forecasting opening price of HCL Technologies is within the training dataset range or matches a similar training dataset pattern, we have found that when employing solely an ANN model (Figure 3(a)), the prediction also falls inside that range. ANN using simply the stochastic MCCV fold (Figure 3(b)) approach reduces the difference between real and predicted data compared to ANN prediction alone. ANN with K-window size (Figure 3(c)) provides predictions that are almost identical to the real data, with just a little variance. Lastly, ANN combined with the stochastic MCCV fold and K-window size (DANN) (Figure 3(d)) provides the highest accuracy when compared to all of the other combinations as shown in Figure 3. Tables 2 to 5 completes the comparison of potential ANN combinations with a statistical metric assessment. The results illustrate the effectiveness of various architectures in enhancing predictive accuracy. According to Table 5, using DANN results in an overall higher accuracy. Specifically, using an ANN with a K=70 sliding window size and a stochastic MCCV fold yields MSE value of 239.965521, MAE value of 11.887598, MPE value of 0.01189, and r value of 0.972921 for opening price forecasting, and MSE value of 574.587406, MAE value of 17.453347, MPE value 0.017466, and r value of 0.931509 for closing price forecasting for HCL Technologies. Based on this information, we select an ANN combination that integrates both K-window size and stochastic MCCV fold, and we assess the overall accuracy for four different stocks. In Table 6, we analyzed the accuracy of DANN across K=30, 40, 50, 60, 70, and 80 days. To check the reliability of the dynamic model, we are not only measuring the accuracy by one of the Pearson's correlation coefficient r; we assessed it with different statistical parameters too. These parameters include MSE, MAE, and MPE, which provide a comprehensive understanding of the model's performance across various conditions. By using a combination of these metrics, we can ensure a robust evaluation of the model's predictive capabilities. From Table 6, we are selecting the K-window size with the highest overall accuracy across all statistical parameters. Table 7 displays the optimal K-window size that provides the highest level of accuracy for forecasting Reliance, Tata Motors, Maruti, and HCL Technologies' respective stocks' opening and closing prices.

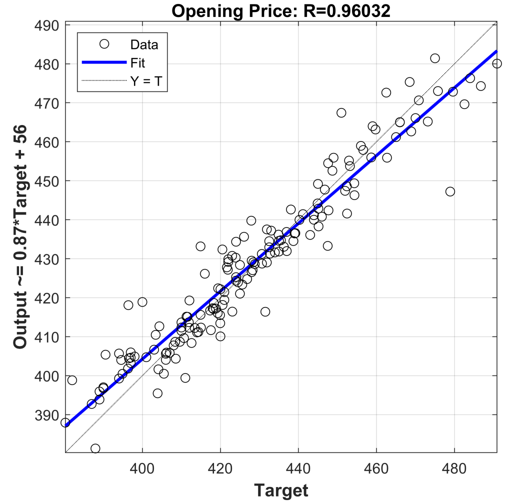
Table 7: Following the training and validation procedure, select the highest accuracy K-window size for forecasting opening and closing prices for proposed DANN

Stock	**Optimum K-window size**	
	Open	**Close**
Reliance	70	70
Tata Motors	80	80
Maruti	70	30
HCL Technologies	70	70

Furthermore, using the MATLAB platform, we have sketched a plot regression in Figure 8 & 9 based on predicted and actual stock prices for opening and closing prices respectively.

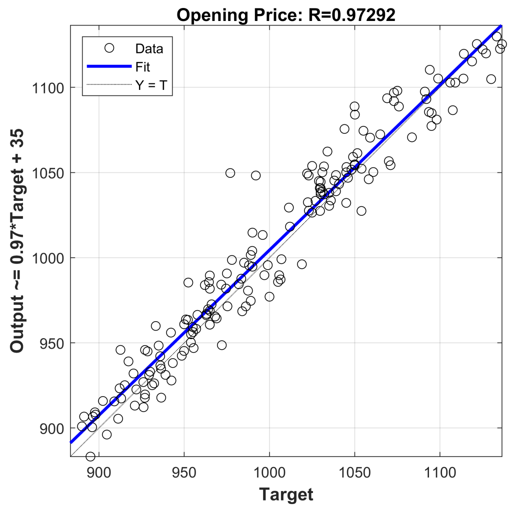


(a) Reliance



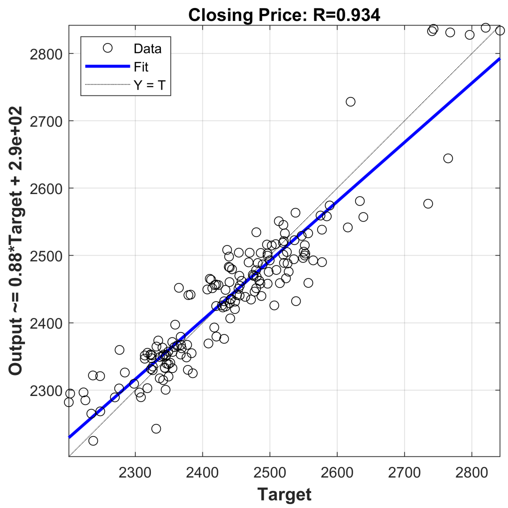
(b) Tata Motors

(c) Maruti



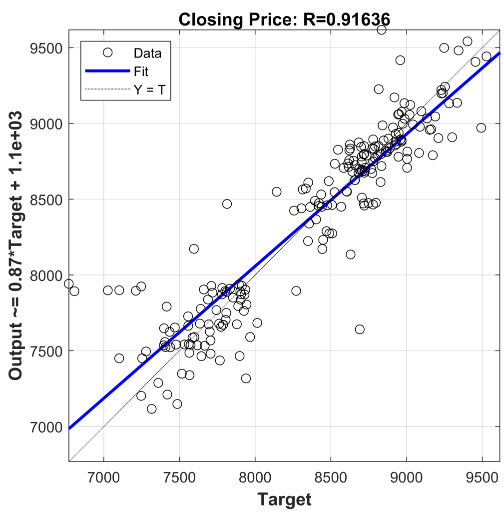
(d) HCL Technologies

Figure 8: (a-d) Regression plots comparing target vs. output opening prices using DANN and K= (a) 70, (b) 80, (c) 70, and (d) 70 days slide window size.

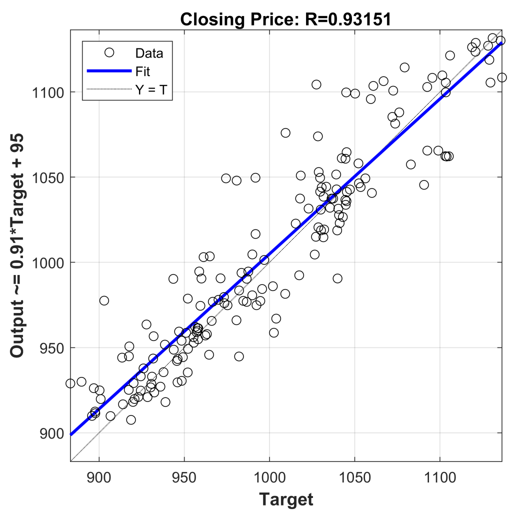


(a) Reliance

(b) Tata Motors



(c) Maruti



(d) HCL Technologies

Figure 9: (a-d) Regression plots comparing target vs. output closing prices using DANN and K= (a) 70, (b) 80, (c) 30, and (d) 70 days slide window size.

5.1. Further Forecasting for Opening and Closing Prices Using DANN Model for Unseen Data

We obtained the best accuracy for ten days of opening price and closing prediction with a sliding window size of K=70 days for HCL Technologies, which helped validate the dynamic model further. Therefore, we used the 70-day window size to predict the opening and closing prices for the next 10 days. We then compared these predictions to the actual opening prices of those 10 days, which were not included in the training and validation process, or more accurately, to unseen data. We obtained a data pattern trend that was nearly identical to the real data, and Table 8 represents the measured statistical assessment values. Stopping is done if, after six iterations, the loss function on the validation dataset doesn't go down compared to the mean squared error of the first six iterations. If this happens, the model stops learning more. As shown in Figure 10(c), from the 14th to the 20th iteration, the validation dataset's loss function did not drop, so we stopped further validation at the 14th iteration to prevent the model from overfitting. Figure 10(a) and Figure 10(b) display the forecasted opening price and line regression graph. After training and validation to DANN with optimal K=70 window size, the final bias and weight values in the matrix form for HCL Technologies are as follows:

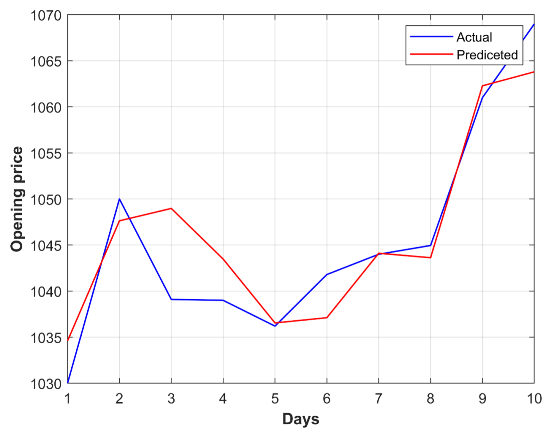
$$\mathrm{IW} = \begin{pmatrix} 0.878602 & -0.27862 & 0.273628 & 0.809154 & -0.61331 & -0.74164 \\ -0.027860 & 1.061771 & -1.00749 & 0.066639 & 0.632902 & -0.12623 \\ -0.58308 & -0.85304 & 0.784608 & -0.10562 & -0.85668 & -0.70738 \end{pmatrix}_{3\times 6},$$

$$b_1 = \begin{bmatrix} -1.66776 \\ -0.20129 \\ -1.55085 \end{bmatrix}_{3\times 1}, \quad b_2 = \begin{bmatrix} 0.367778 \\ 0.236885 \end{bmatrix}_{2\times 1},$$

$$\mathrm{LW} = \begin{pmatrix} 0.534473 & 0.876636 & -0.27937 \\ 0.640762 & 0.556244 & -0.51662 \end{pmatrix}_{2\times 3}.$$

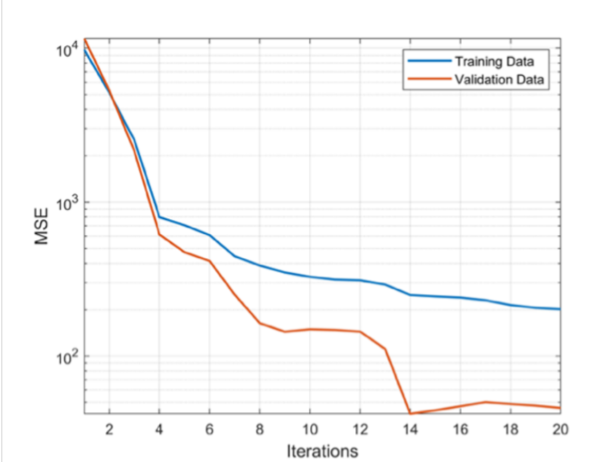
Table 8: Statistical metrics for assessing performance for HCL Technologies' next ten days opening price.

MSE	**MAE**	**MPE**	r
19.68337	3.42683	0.003281	0.92172



(a) Forecast of HCL Technologies' opening price for the following ten days.

(b) Regression line comparing target vs. output



(c) Stopping criteria of validation

Figure 10: (a-c) HCL Technologies: Forecasting, Regression Analysis and Stopping criteria of validation

6. Conclusion

To the best of our knowledge, no research has been done on the application of the FRM approach to ANN, a regression task for the prediction of the opening and closing prices for the following ten days, using the dynamic K-window size and the stochastic MCCV fold. The suggested methodology was evaluated using four distinct stock sectors chosen from the NIFTY50 INDEX. Comparing the suggested dynamic technique to the standard ANN simulation, the findings show that it could achieve great performance in terms of opening and closing price prediction performance measures. These days, several industries are using ANN approaches to suit their models. One of the foundations of forecasting is that no approach should be considered as the official method for a given model. When using ANN approaches for forecasting, it is true that not every model will perform the same for every stock under consideration. Nevertheless, we must assess the effectiveness of any algorithm that fits a given set of data using error values or a correlation factor. Thus, the combination of the model and data is crucial. The model's accuracy in forecasting the stock markets is essential for developing trading and investment methods that yield more profits. Financial time series are noisy, yet the data they contain is crucial for identifying potential patterns in the future. ANNs heavily emphasize data preparation as input data, and we included this attention into our own model as well. One thing to keep in mind is that, depending on how the stock market trend develops over time, the number of dynamic window sizes may vary for data from different time frames. Thus, in order to obtain precise predictions, we must regularly simulate the data. As research progresses, there will be many chances to control ANN procedures using optimization and data pretreatment techniques from statistical and machine learning (ML) methods including particle swarm optimization (PSO), principal component analysis (PCA), support vector machines (SMV) and regression analysis. This approach will allow researchers to adapt their methodologies and findings to enhance understanding in diverse areas. By systematically evaluating the results, the study can provide valuable insights that transcend its initial application.

Conflict of Interest

The authors have made equal contributions and provided their agreement for publishing. The authors declare that they have no competing interests. The authors additionally state that they possess no identifiable relationships or competing financial interests that may have seemed to influence the work presented in this study.

Data Availability Statement

Datasets generated and analyzed during the current study are available from Yahoo Finance, (https://finance.yahoo.com/).The datasets generated during and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Funding

The authors did not receive support from any organization for the submitted work.

Acknowledgement

The authors acknowledge the Department of Mathematics, School of Advanced Sciences, Vellore Institute of Technology, Vellore, India, for permitting the research at the FIST Lab (SR/FST/MS-II/2023/139), sponsored by the Department of Science and Technology (DST), Government of India. The authors express appreciation to the anonymous reviewers and editors for their constructive input and insightful suggestions, which have greatly enhanced this work. Their knowledge as well as dedication have significantly enhanced our work.

References

1. M. Al-Baali, *Descent property and global convergence of the fletcher–reeves method with inexact line search*, IMA Journal of Numerical Analysis **5** (1985), no. 1, 121–124.
2. Esteban Alfaro, Noelia García, Matías Gámez, and David Elizondo, *Bankruptcy forecasting: An empirical comparison of adaboost and neural networks*, Decision Support Systems **45** (2008), no. 1, 110–122.
3. G. Avinash, V. Ramasubramanian, Mrinmoy Ray, Ranjit Kumar Paul, Samarth Godara, and G. H. Harish Nayak, *Hidden markov guided deep learning models for forecasting highly volatile agricultural commodity prices*, Applied Soft Computing **158** (2024), 111557.
4. Søren Bisgaard and Murat Kulahci, *Time series analysis and forecasting by example*, John Wiley & Sons, Hoboken, NJ, USA, 2011.
5. Priyanka Chandra and Raja Das, *A hybrid rsa-ipa optimizer for designing an artificial neural network to study the jeffery-hamel blood flow with copper nanoparticles: Application to stenotic tapering artery*, Results in Engineering **20** (2023), 101542.
6. Ying-Hua Chang and Chen-Wei Huang, *Utilizing genetic algorithms in conjunction with ann-based stock valuation models to enhance the optimization of stock investment decisions*, AI **5** (2024), no. 3, 1011–1029.
7. Thamiso Diphofu and P. Kaelo, *A modified extended fletcher–reeves conjugate gradient method with an application in image restoration*, International Journal of Computer Mathematics **102** (2025), no. 6.
8. Roger Fletcher and Michael J. D. Powell, *A rapidly convergent descent method for minimization*, The Computer Journal **6** (1963), no. 2, 163–168.
9. Roger Fletcher and Colin M. Reeves, *Function minimization by conjugate gradients*, The Computer Journal **7** (1964), no. 2, 149–154.
10. Mustafa Göçken, Mehmet Özçalıcı, Aslı Boru, and Ayşe Tuğba Dosdoğru, *Integrating metaheuristics and artificial neural networks for improved stock price prediction*, Expert Systems with Applications **44** (2016), 320–331.
11. Liu Guanghui, Han Jiye, and Yin Hongxia, *Global convergence of the fletcher-reeves algorithm with inexact linesearch*, Applied Mathematics-A Journal of Chinese Universities **10** (1995), no. 1, 75–82.
12. Eka Hayana Hasibuan, Surya Hendraputra, G. S. Achmad Daengs, and Liharman Saragih, *Comparison fletcher-reeves and polak-ribiere ann algorithm for forecasting analysis*, Journal of Physics: Conference Series, vol. 2394, IOP Publishing, Dec 2022, p. 012008.
13. H. S. Hota, Richa Handa, and Akhilesh Kumar Shrivas, *Time series data prediction using sliding window based rbf neural network*, International Journal of Computational Intelligence Research **13** (2017), no. 5, 1145–1156.
14. Razan Jamous, Hosam ALRahhal, and Mohamed El-Darieby, *A new ann-particle swarm optimization with center of gravity (ann-psocog) prediction model for the stock market under the effect of covid-19*, Scientific Programming **2021** (2021), 6656150.
15. Fathe Jeribi, R. John Martin, Ruchi Mittal, Hassan Jari, Abdulrahman Hassan Alhazmi, Varun Malik, S. L. Swapna, S. B. Goyal, Manoj Kumar, and Shubhranshu Vikram Singh, *A deep learning based expert framework for portfolio prediction and forecasting*, IEEE Access **12** (2024), 103810–103829.
16. Binita Kumari and Tripti Swarnkar, *Forecasting daily stock movement using a hybrid normalization based intersection feature selection and ann*, Procedia Computer Science **218** (2023), 1424–1433.

17. Balakrishnama Manohar and Raja Das, *Artificial neural networks for prediction of covid-19 in india by using backpropagation*, Expert Systems **40** (2023), no. 5, e13105.

18. Balakrishnama Manohar, Raja Das, and M. Lakshmi, *A hybridized lstm-ann-rsa based deep learning models for prediction of covid-19 cases in eastern european countries*, Expert Systems with Applications **256** (2024), 124977.

19. Balakrishnama Manohar, Raja Das, M. Lakshmi, Jaykumar Ishvarbhai Prajapati, and K. Madhusudhan Reddy, *Artificial neural network-based stock price prediction using levenberg-marquardt algorithm*, Artificial Intelligence-based Solutions for Industrial Applications (S. B. Goyal et al., eds.), CRC Press, Boca Raton, 2024, pp. 315–337.

20. Shashi Kant Mishra, Suvra Kanti Chakraborty, Mohammad Esmael Samei, and Bhagwat Ram, *A q-polak–ribière–polyak conjugate gradient algorithm for unconstrained optimization problems*, Journal of Inequalities and Applications **2021** (2021), no. 1, 25.

21. Amod Murkute and Tanuja Sarode, *Forecasting market price of stock using artificial neural network*, International Journal of Computer Applications **124** (2015), no. 12, 11–15.

22. Sarat Chandra Nayak, Bijan Bihari Misra, and Himansu Sekhar Behera, *An adaptive second order neural network with genetic-algorithm-based training (asonn-ga) to forecast the closing prices of the stock market*, International Journal of Applied Metaheuristic Computing **7** (2016), no. 2, 39–57.

23. Bharat Bhushan Negi, Mansi Aliveli, Shishir Kumar Behera, Raja Das, Arindam Sinharoy, Eldon R. Rene, and Kannan Pakshirajan, *Predictive modelling and optimization of an airlift bioreactor for selenite removal from wastewater using artificial neural networks and particle swarm optimization*, Environmental Research **219** (2023), 115073.

24. Norita Md Norwawi, *Sliding window time series forecasting with multilayer perceptron and multiregression of covid-19 outbreak in malaysia*, Data Science for COVID-19 (Utku Kose, Deepak Gupta, Victor Hugo C. de Albuquerque, and Ashish Khanna, eds.), Elsevier, 2021, pp. 547–564.

25. Jigar Patel, Sahil Shah, Priyank Thakkar, and Ketan Kotecha, *Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques*, Expert Systems with Applications **42** (2015), no. 1, 259–268.

26. Ajit Mohan Pattanayak, Aleena Swetapadma, and Biswajit Sahoo, *Exploring different dynamics of recurrent neural network methods for stock market prediction-a comparative study*, Applied Artificial Intelligence **38** (2024), no. 1, 2371706.

27. Paramita Ray, Bhaswati Ganguli, and Amlan Chakrabarti, *Multivariate bayesian time-series model with multi-temporal convolution network for forecasting stock market during covid-19 pandemic*, International Journal of Computational Intelligence Systems **17** (2024), no. 1, 170.

28. Sipra Sahoo and Mihir Narayan Mohanty, *Stock market price prediction employing artificial neural network optimized by gray wolf optimization*, New Paradigm in Decision Science and Management (Srikanta Patnaik, Andrew W. H. Ip, Madjid Tavana, and Vipul Jain, eds.), Springer, Singapore, 2020, pp. 77–87.

29. Philipp Schlieper, Mischa Dombrowski, An Nguyen, Dario Zanca, and Bjoern M. Eskofier, *Data-centric benchmarking of neural network architectures for the univariate time series forecasting task*, Forecasting **6** (2024), no. 3, 718–747.

30. Gülhan Toğa, Berrin Atalay, and M. Duran Toksari, *Covid-19 prevalence forecasting using autoregressive integrated moving average (arima) and artificial neural networks (ann): Case of turkey*, Journal of Infection and Public Health **14** (2021), no. 7, 811–816.

31. Anjar Wanto, Muhammad Zarlis, Dedy Hartama, Sawaluddin, T. Julfia, H. Nasution, and Saifullah, *Analysis of artificial neural network backpropagation using conjugate gradient fletcher reeves in the predicting process*, Journal of Physics: Conference Series, vol. 930, IOP Publishing, Dec 2017, p. 012018.

32. Fujin Zhuang, Xiao Chen, Punyaphol Horata, and Khamron Sunat, *Research on hybrid architecture neural networks for time series prediction*, IEEE Access **13** (2025), 101507–101531.

A. Appendix

A.1. Global Convergence of FRM

Let us consider wolf line search rule given $0 < \epsilon_1 < \frac{1}{2}$. Select step size α_i satisfying

$$L(x_i + \alpha_i d_i) \leq L(x_i) + \epsilon_1 \alpha_i g_i^T d_i \tag{A.1}$$

$$|g(x_i + \alpha_i d_i)^T d_i| \leq -\frac{1}{2} g_i^T d_i \tag{A.2}$$

To prove the global convergence, assuming the below statement (H):

- The level set $M_0 = \{x \mid L(x) \leq L(x_1)\}$ is bounded.
- $L \in C^1$. Furthermore, g satisfies the Lipschitz condition on M_0. That is, for any $x, y \in M_0$, there exists a constant $M > 0$ such that,

$$\|g(x) - g(y)\| \leq M\|x - y\|.$$

Lemma A.1 *Suppose that the sequence $\{x_i\}$ is generated by employing the Flecther-Reeves algorithm using the linesearch (A.1) and (A.2). Then $0 < \xi_i < 2$. where*

$$\xi_i = -\frac{g_i^T d_i}{\|g_i\|^2}.$$

Proof: We know that $\sum_{n=0}^{i} \left(\frac{1}{2}\right)^n < 2$, $d_1 = -g_1$, by (3.2) and (A.2), we have

$$\xi_i = 1 - \frac{g_i^T d_{i-1}}{\|g_{i-1}\|^2} \geq 1 - \frac{1}{2}\xi_{i-1};$$

$$\xi_i \leq 1 + \frac{1}{2}\xi_{i-1} \leq \cdots \leq \sum_{n=0}^{i-1} \left(\frac{1}{2}\right)^n < 2;$$

$$\xi_i \geq 1 - \frac{1}{2}\left(1 + \frac{1}{2}\xi_{i-2}\right) \geq 1 - \sum_{n=1}^{i-1} \left(\frac{1}{2}\right)^n > 0.$$

Therefore, $0 < \xi_i < 2$. From the above, it is clear that at each iterate i, $g_i^T d_i < 0$. □

Lemma A.2 *Suppose that (H) holds. Then,*

$$\sum_{i=1}^{\infty} (\eta_i + \eta_{i-1})^2 < +\infty, \quad \textit{where } \eta_i = -\frac{g_i^T d_i}{\|d_i\|}.$$

Proof: By (A.2), we get

$$\|g_{i+1} - g_i\| \|d_i\| \geq (g_{i+1} - g_i)^T d_i \geq -\frac{1}{2} g_i^T d_i,$$
$$\|g_{i+1} - g_i\| \geq \frac{1}{2}\eta_i.$$

By (H) (ii), we have $\alpha_i \|d_i\| \geq c_1 \eta_i$ and $c_1 = \frac{1}{2M}$. Substituting it into (A.1), we get

$$L(x_{i+1}) \leq L(x_i) - \epsilon_1 \alpha_i \|d_i\| \eta_i \leq L(x_i) - \epsilon_1 c_1 \eta_i^2.$$

This together with (H)(i) gives,

$$\sum_{i=1}^{\infty} \eta_i^2 \leq \frac{1}{c_1 \epsilon_1} \sum_{i=1}^{\infty} [L(x_i) - L(x_{i+1})] < \infty.$$

Since

$$\eta_i \eta_{i-1} \leq \frac{\eta_i^2 + \eta_{i-1}^2}{2},$$

Therefore,

$$\sum_{i=1}^{\infty} (\eta_i + \eta_{i-1})^2 \leq \infty.$$

□

A.2. Corollary of a Hypothesis

To prove global convergence in the subsequent section, we assume that for some function $L(x)$ there exists a positive constant c_2 such that

$$\|g_i\| \geq c_2,\ i = 1, 2, 3, \ldots \tag{A.3}$$

Where sequence $\{x_i\}$ is obtained by the fletcher-reeves algorithm using the inexact line search (A.1) & (A.2). We may presume in the following discussion that there exists a positive constant c_3 such that $c_3 = sup_i \ \|g_i\| < \infty$, which is plausible due to (H)(i).

Corollary A.1 *Assume that the inequality* (A.3) *holds. Then*

$$\|d_i\| < \frac{2c_3^2\sqrt{i}}{c_2}.$$

Proof: It is follows from (3.2) that,

$$\|d_i\|^2 = \|g_i\|^2 - 2\beta_{i-1}g_i^T d_{i-1} + \beta_{i-1}^2\|d_{i-1}\|^2 \tag{A.4}$$

By (3.4), (A.2) and Lemma A.1, We have

$$\begin{aligned}
\|d_i\|^2 &\leq \|g_i\|^2 - \beta_{i-1}g_{i-1}^T d_{i-1} + \beta_{i-1}^2\|d_{i-1}\|^2 \\
&\leq \|g_i\|^2 + \frac{\|g_i\|^2}{\|g_{i-1}\|^2}\left(2\|g_{i-1}\|^2\right) + \frac{\|g_i\|^4}{\|g_{i-1}\|^4}\|d_{i-1}\|^2 \\
&= \|g_i\|^4\left(\frac{3}{\|g_i\|^2} + \frac{\|d_{i-1}\|^2}{\|g_{i-1}\|^4}\right).
\end{aligned}$$

Hence,

$$\begin{aligned}
\frac{\|d_i\|^2}{\|g_i\|^4} &\leq \frac{3}{\|g_i\|^2} + \frac{\|d_{i-1}\|^2}{\|g_{i-1}\|^4} \\
&\leq \frac{3}{c_2^2} + \frac{\|d_{i-1}\|^2}{\|g_{i-1}\|^4} \\
&\leq \cdots \leq (i-1)\frac{3}{c_2^2} + \frac{\|d_1\|^2}{\|g_1\|^4} < \frac{4i}{c_2^2}.
\end{aligned}$$

And so,

$$\|d_i\| < \frac{2\sqrt{i}\|g_i\|^2}{c_2} \leq \frac{2\sqrt{i}c_3^2}{c_2}.$$

□

Corollary A.2 *Assume that both* (A.3) *and (H) hold. Then*

$$\liminf_i \ \|d_i\| \ > 0.$$

Proof: By Lemma A.2, It is obvious that

$$\lim_{i\to\infty} \ \|\eta_i\| \ = 0. \tag{A.5}$$

Assume that

$$\liminf_i \|d_i\| = 0,$$

i.e., there is an infinite subset I of $\mathbb{N} = \{1, 2, 3, \dots\}$ such that

$$\lim_{i \in I} \|d_i\| = 0. \tag{A.6}$$

From (A.5), we know that there exists a positive integer i_0 such that for all $i \geq i_0$,

$$\eta_i < \frac{c_2^7}{2c_3^6} + \frac{3c_2^3}{2c_3^2}. \tag{A.7}$$

By (3.2), (A.2) and (A.3), we have for all i,

$$\begin{aligned} \|d_{i+1}\|^2 &= \|g_{i+1}\|^2 - 2\beta_i g_{i+1}^T d_i + \beta_i^2 \|d_i\|^2 \\ &\geq \|g_{i+1}\|^2 + \beta_i g_i^T d_i + \beta_i^2 \|d_i\|^2 \\ &\geq c_2^2 + \|d_i\| \left[\frac{c_2^4}{c_3^4} \|d_i\| - \frac{c_3^2}{c_2^2} \eta_i \right]. \end{aligned} \tag{A.8}$$

Hence, from (A.5) and (A.6), we have

$$\liminf_{i \in I} \|d_{i+1}\|^2 \geq c_2^2 + \lim_{i \in I} \|d_i\| \left[\frac{c_2^4}{c_3^4} \|d_i\| - \frac{c_3^2}{c_2^2} \eta_i \right] = c_2^2 > \frac{c_2^2}{4}.$$

Therefore, there is a positive integer $i_1 \in I$ with $i_1 \geq i_0$ such that,

$$\|d_{i_1+1}\|^2 \geq \frac{c_2^2}{4}, \quad \text{or} \quad \|d_{i_1+1}\| \geq \frac{c_2}{2}. \tag{A.9}$$

Letting $i = i_1 + 1$ in (A.8), we get

$$\|d_{i_1+2}\|^2 \geq c_2^2 + \|d_{i_1+1}\| \left[\frac{c_2^4}{c_3^4} \|d_{i_1+1}\| - \frac{c_3^2}{c_2^2} \eta_{i_1+1} \right].$$

So, from (A.9) and (A.7), we obtain

$$\|d_{i_1+2}\|^2 \geq c_2^2 + \frac{c_2}{2} \left[\frac{c_2^4}{c_3^4} \frac{c_2}{2} - \frac{c_3^2}{c_2^2} \left(\frac{c_2^7}{2c_3^6} + \frac{3c_2^3}{2c_3^2} \right) \right] = \frac{c_2^2}{4}, \quad \text{or} \quad \|d_{i_1+2}\| \geq \frac{c_2}{2}.$$

Iteratively, we arrive at $\|d_{i_1+k}\| \geq c_2/2$ for $k = 2, 3, \dots$, which contradicts (A.6). Therefore,

$$\liminf_{i} \|d_i\| > 0.$$

□

Corollary A.3 *Assume that (H) and* (A.3) *are hold. Then, for every $i \geq i_2$, there is a positive integer $i_2 > 0$,*

$$\|d_i\| > \frac{c_2^2}{2c_3^2} \|d_{i-1}\|.$$

Proof: From Corollary A.2, we have

$$\liminf_{i} \|d_i\| > 0.$$

We may assume that there exists a constant $c_4 > 0$, such that

$$\|d_i\| \geq c_4, \quad i = 1, 2, \dots \tag{A.10}$$

Using the inequality (A.8), we have

$$\frac{\|d_{i+1}\|^2}{\|d_i\|^2} \geq \frac{c_2^2}{\|d_i\|^2} + \frac{c_2^4}{c_3^4} - \frac{c_3^2}{c_2^2} \frac{\eta_i}{\|d_i\|}. \tag{A.11}$$

By (A.10), (A.11) and corollary A.1, lead to the conclusion that

$$\frac{\|d_{i+1}\|^2}{\|d_i\|^2} \geq \frac{c_2^2}{\frac{4i}{c_2^2}c_3^4} + \frac{c_2^4}{c_3^4} - \frac{c_3^2}{c_2^2}\frac{\eta_i}{c_4}$$

and

$$\liminf_i \frac{\|d_{i+1}\|^2}{\|d_i\|^2} \geq \frac{c_2^4}{c_3^4} \geq \frac{c_2^4}{4c_3^4}.$$

Consequently, there exists a positive integer i_2 such that for all $i \geq i_2$,

$$\frac{\|d_i\|^2}{\|d_{i-1}\|^2} \geq \frac{c_2^4}{4c_3^4}, \quad \text{or} \quad \|d_i\| > \frac{c_2^2}{2c_3^2}\|d_{i-1}\|.$$

□

A.3. Global Convergence

Theorem A.1 *Suppose (H) is true. Let x_1 can be any starting point and the sequence $\{x_i\}$ is formed by the Fletcher-Reeves algorithm using the line search (A.1)-(A.2). Then*

$$\liminf_i \|g_i\| = 0.$$

Proof: If possible assume that,

$$\liminf_i \|g_i\| > 0,$$

i.e., inequality (A.3) holds. Take inner products of $-g_i$ on both sides of (3.2), we obtain

$$-g_i^T d_i = \|g_i\|^2 - \frac{\|g_i\|^2}{\|g_{i-1}\|^2} g_i^T d_{i-1}.$$

So, by (A.2) and (A.3), we get

$$\eta_i = \frac{1}{\|d_i\|^2}\left[\|g_i\|^2 - \frac{\|g_i\|^2}{\|g_{i-1}\|^2} g_i^T d_{i-1}\right] \geq \frac{c_2^2}{\|d_i\|} + \frac{c_3^2}{c_2^2\|d_i\|}\frac{1}{2}g_{i-1}^T d_{i-1},$$

$$\frac{1}{\|d_i\|} \leq \frac{1}{c_2^2}\eta_i + \frac{c_3^2}{c_2^4\|d_i\|}\left(-\frac{1}{2}g_{i-1}^T d_{i-1}\right).$$

Using corollary A.3,

$$\frac{1}{\|d_i\|} \leq \frac{1}{c_2^2}\eta_i + \frac{c_3^2}{c_2^4\frac{c_2^2}{2c_3^2}}\frac{1}{2}\eta_{i-1} \leq \max\left\{\frac{1}{c_2^2}, \frac{c_3^4}{c_2^6}\right\}(\eta_i + \eta_{i-1}).$$

From lemma A.2, we have

$$\sum_{i=2}^{\infty} \frac{1}{\|d_i\|^2} \leq \sum_{i=2}^{\infty}\left[\max\left\{\frac{1}{c_2^2}, \frac{c_3^4}{c_2^6}\right\}\right]^2 (\eta_i + \eta_{i-1})^2 < \infty.$$

By corollary A.1, we have

$$\sum_{i=2}^{\infty} \frac{1}{\|d_i\|^2} > \sum_{i=2}^{\infty} \frac{c_2^2}{4ic_3^4} = +\infty.$$

This is contradictory to our assumption. Hence the result. □

Jaykumar Ishvarbhai Prajapati and Raja Das (Corresponding author.),
Department of Mathematics, School of Advanced Sciences,
Vellore Institute of Technology, Vellore - 632 014, Tamil Nadu, India.
E-mail address: jaykumar.ishvarbhai2022@vitstudent.ac.in,
rajadas@vit.ac.in.(corresponding author's e-mail)