



Swarm Intelligence Based Hybrid Neural Network Approach for Stock Price Forecasting

Gourav Kumar¹ · Uday Pratap Singh² · Sanjeev Jain³

Accepted: 2 August 2021 / Published online: 29 August 2021

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract

In this paper, a two-stage swarm intelligence based hybrid feed-forward neural network approach is designed for optimal feature selection and joint optimization of trainable parameters of neural networks in order to forecast the close price of Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index for multiple-horizon (1-day ahead, 5-days-ahead and 10-days ahead) forecasting. Although the neural network can deal with complex non-linear and uncertain data but defining its architecture in terms of number of input features in the input layer, the number of neurons in the hidden layer and optimizing the weights is a challenging problem. The back-propagation algorithm is frequently used in the neural network and has a drawback to getting stuck in local minima and overfitting the data. Motivated by this, we introduce a swarm intelligence based hybrid neural network model for automatic search of features and other learnable neural networks' parameters. The proposed model is a combination of discrete particle swarm optimization (DPSO), particle swarm optimization (PSO) and Levenberg–Marquardt algorithm (LM) for training the feed-forward neural networks. The DPSO attempts to search automatically the optimum number of features and the optimum number of neurons in the hidden layer of FFNN whereas PSO, simultaneously tune the weights and bias in different layers of FFNN. This paper also compares the forecasting efficiency of proposed model with another hybrid model obtained by integrating binary coded genetic algorithm and real coded genetic algorithm with FFNN. Simulation results indicate that the proposed model is effective for obtaining the optimized feature subset and network structure and also shows superior forecasting accuracy.

Keywords Discrete particle swarm optimization · Genetic algorithm · Hybrid neural network · Particle swarm optimization · Stock price time series · Swarm intelligence · Technical analysis

✉ Uday Pratap Singh
usinghiitg@gmail.com

Extended author information available on the last page of the article

Abbreviations

ABC	Artificial Bee Colony
ACO	Ant Colony Optimization
ACRO	Artificial Chemical Reaction Optimization
ADI	Accumulation Distribution Index
ADX	Average Directional Index
ANNs	Artificial Neural Networks
ANFIS	Adaptive Network Fuzzy Inference System
ARCH	Autoregressive Conditional Heteroskedasticity
ARIMA	Autoregressive Integrated Moving Average
ARMA	Autoregressive Moving Average
ARV	Average Relative Variance
ATR	Average True Range
BBBC	Big-Bang-Big Crunch (BBBC) Optimization
BCGA	Binary Coded Genetic Algorithm
BFO	Bacterial Foraging Optimization
BP	Back-Propagation
BSE	Bombay Stock Exchange
CCI	Commodity Channel Index
CI	Computational Intelligence
CL	Close Price
DE	Differential Evolution
DJIA	Dow Jones Industrial Average
DNN	Dynamic Neural Network
DPSO	Discrete Particle Swarm Optimization
EC	Evolutionary Computation
ELM	Extreme Learning Machine
EMA	Exponential Moving Average
ENN	Elman Neural Network
ES	Exponential Smoothing
FCRBFNN	Fully Complex-valued Radial Basis Function Neural Network
FFNN	Feed-Forward Neural Network
FL	Fuzzy Logic
FLANN	Functional Link Artificial Neural Network
GA	Genetic Algorithm
GARCH	Generalized Autoregressive Conditional Heteroskedasticity
GRNN	Generalized Regression Neural Network
GWO	Gray Wolf Optimization
HI	High Price
HS	Harmony Search
IHHO	Improved Harris's hawks optimization
ISCA	Improved Sine Cosine Algorithm
IT2FS	Interval Type-2 Fuzzy System
%K	Stochastic %K
LM	Levenberg–Marquardt Algorithm
LMBP	Levenberg–Marquardt Back-Propagation

LO	Low Price
MAAPE	Mean Arctangent Absolute Percentage Error
MACD	Moving Average Convergence Divergence
MOEA	Multi-objective Evolutionary Optimization
MOMM	Multiobjective Model-Metric (MOMM) Learning
MSE	Mean Squared Error
MTM	Momentum
NAS	Neural Architecture Search
NIA	Nature Inspired Algorithm
OBV	On-Balance Volume
OHLC	Open High Low Close
OP	Open Price
PSO	Particle Swarm Optimization
QRNN	Quantile Regression Neural Network
%R	William %R
RBFNN	Radial Basis Functional Neural network
RCGA	Real Coded Genetic Algorithm
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
ROC	Rate of Change
RSI	Relative Strength Index
RW	Random Walk
SI	Swarm Intelligence
SMA	Simple Moving Average
SMAPE	Symmetric Mean Absolute Percentage Error
TA	Technical Analysis
TSI	True Strength Index
UI	Ulcer Index

1 Introduction

Stock price time series exhibit complex non-linear, chaotic, highly volatile and dynamic behavior with irregular movements and hence, it is considered as highly unpredictable (Fama, 1970). Forecasting stock price time series is one of the most emerging areas of research and challenging tasks for researchers and financial analyst as stock market plays a crucial role in the financial growth of country as well as global economic status because economic developments of the countries are effected by the various financial activities (Lin et al., 2012). Fluctuation in stock market is governed by various macro-economic and micro-economic factors like political stability, government policies, general economic status, organization's growth, investor's expectations, global economic conditions, investor's psychology etc. (Haleh et al., 2011; Menkhoff, 1997). However, the exact factors that have greatest influence on the stock market are not known. The only information that is available from the stock market is the prices. The Stock price time series that can be represented as: $x_t = \{x_t \in R | t = 1, 2, 3, \dots, N\}$ is the set of prices

recorded at regular interval of time t . The major focus of stock price time series forecasting approaches is to forecast the future prices of the series on the account of the regular pattern present in current prices of the series itself.

Some previous approaches that have attempted the forecasting of stock market are fundamental analysis (Nassirtoussi et al., 2011), technical analysis (Menkhoff, 1997), traditional time series forecasting techniques (Zhang et al., 2008) such as autoregressive moving average (ARMA) (Box et al., 2015), exponential smoothing (ES) (Wang et al., 2012), autoregressive integrated moving average (ARIMA) (Adebiyi et al., 2014; Ariyo et al., 2014; Yao et al., 1999), autoregressive conditional heteroskedasticity (ARCH) and generalized autoregressive conditional heteroskedasticity (GARCH) (Guresen et al., 2011a; Kristjanpoller & Minutolo, 2018). However the stock price time series data are highly volatile, possess complex non-linear behavior, highly noisy, dynamic and chaotic in nature (Si & Yin, 2013). Hence, traditional time series forecasting methods have a limitation that they cannot capture the complex non-linear behavior of stock markets.

In recent years, computational intelligence (CI) and nature inspired optimization based hybrid models have been proposed for stock price time series forecasting. Computational intelligence (CI) is an evolving computation approach which mimics the thinking capability of human brain to learn and generalize in an environment of uncertainty and imprecision for solving complex real world problems (Ibrahim, 2016). The popular CI approaches that are collectively used to create hybrid models include artificial neural network (ANN), fuzzy Logic (FL), genetic algorithm (GA) and various nature inspired algorithms (NIA) such as particle swarm optimization (PSO), differential evolution (DE), ant colony optimization (ACO), artificial bee colony (ABC), bacterial foraging optimization (BFO) etc. ANN is self-organizing, data-oriented and auto-adaptive approach which can be effectively applied to forecast the stock price as it cannot incorporate complex model structure and have an ability to capture the non-linear relation between input and output without prior assumption about data (Guresen et al., 2011b; Lu et al., 2009). Due to capability of ANNs in order to deal with non-linear data and uncertainty, it is widely used for forecasting stock price time series (Liu & Wang, 2012). Feed-forward neural networks (FFNN) and recurrent neural network (RNN) are two most frequent used artificial neural networks technique for stock market forecasting (Atsalakis & Valavanis, 2009). Yan (2012) employed a fusion of generalized regression neural network (GRNN) model in order to automatic search of design parameters of ANN for time series forecasting problem. Zhou et al. (2016) et al. proposed a hybridization of dendritic neuron model and phase space reconstruction to predict the financial time series and obtained promising result. Gao et al. (2018) applied six nature inspired learning algorithms including PSO and GA to train the dendritic neuron model for solving 14 problems. Kim (2006) combined ANN with GA for selecting the feature instances and optimizing the inter-neuron connectionist weights and achieved the improved predictive ability of proposed model and reduced the training time. Gong et al. (2018) used a multiobjective learning algorithm namely multiobjective evolutionary algorithm (MOEA) to optimize the recurrent neural network (RNN) and termed the model as multiobjective model-metric (MOMM) learning which simultaneously

optimizes the network structure, data representation and the time series model separation for time series approximation and classification.

The biggest challenge in ANNs is defining its architecture in terms of number of nodes in input layer (input features), number of processing units in hidden layers and output layer as well as tuning the parameters of neural network such as connection weight and bias of the network, appropriate activation functions, learning rate etc. so as to obtain accurate result. Currently the architecture of ANNs have been defined manually by trial and error approaches or by human experts which is cumbersome process that takes lot of time and prone to error. Hence, there is growing need of automatic search of optimal neural network architecture. Various authors have attempted the automated search of neural network architecture (Elsken et al., 2018a). Han et al. (2016) proposed adaptive PSO to automatically optimize the network structure and weights of radial basis function neural network (RBFNN) simultaneously and verified the effectiveness of proposed model with five different problems including two time series prediction problems. Wang and Kumbasar (2019) applied PSO and big-bang-big crunch (BBBC) optimization techniques to tune the parameters of a hybrid system obtained by combining interval type-2 fuzzy system (IT2FS) with ANN and demonstrated the capability of proposed model by approximating two time-varying non-linear system. Kapanova et al. (2018) applied GA to develop an automatic approach that search for the optimal network architecture of ANN for fitting a specific function. Sakshi and Kumar (2019) combined the pruning of redundant weights and GA for optimizing the parameters of ANN terming the model as neuro-genetic technique and observed that the proposed model shows the fast convergence, short training time and higher success rate.

Defining the number of nodes in input layer and output layers are problem specific. However, the number of nodes in input layer can be obtained by selecting appropriate features. Selecting and reducing the dimension of input feature space for stock market forecasting is a challenging task. Dimensionality can be reduced either by feature selection or feature extraction techniques (Webb, 2003). In soft computing, high dimensionality in dataset may result in the problem of overfitting, high computational complexity and reduction in the performance of the model. Hence, by reducing the dimensions of feature space one can overcome these problems. Dimensionality reduction aims at representing the feature space with lower features while preserving the most of the information present in original dataset. Feature selection techniques minimize the feature space by excluding the irrelevant features and choosing the appropriate one. Various feature selection approaches have been used to enhance the interpretability of the model (Kumar et al., 2016; Guyon & Elisseeff, 2003).

Selecting the number of neurons in hidden layers is also a key challenge in ANNs because there is no well-defined method for defining the optimum number of neurons in hidden layers of ANNs. For this purpose researchers generally employ trial and error approaches. Selecting the too few or too many hidden neurons can cause the problem of underfitting or overfitting respectively, hence degrading the accuracy of forecasting model. In this study, we consider two issues; first one is automatic searching the network architecture i.e. number of neurons in input layer (optimum set of features) as well as number of neurons in hidden layer of FFNN. The

performance of correctly defined model depends upon the selection of appropriate input variables i.e. number of nodes in input layer. Hence, in this study, key features are selected using swarm intelligence technique.

Another challenging problem considered in this work is tuning the initial parameters (weights and bias) of FFNN. Updating the weight and bias in ANNs according to predefined criteria is known as training or learning of neural networks. Training in ANNs is governed by minimizing the loss function like mean squared error between actual and predicted value averaged over all training samples. Most of training methods such as conjugate gradient, Quasi-Newton method, gradient descent, Newton method, and Levenberg–Marquardt (LM) are gradient based (Basheer & Hajmee, 2000). Among the learning algorithms, a second order technique like LM (More, 1978) method can be successfully used to obtain more accurate result. The LM method is powerful optimization technique used in ANNs as it has a capability to speed-up the learning procedure and convergence of the networks. It uses back-propagation (BP) algorithm in which gradients are computed and propagated iteratively from last layer of the network to first layer until the error between actual and predicted value reached to minimum level or other stopping criteria such as number of epochs are satisfied (Hagan & Menhaj, 1994). Despite being good performance of LM method in some areas, it has two drawbacks: firstly, it gets trapped in local minima if the loss function is multimodal or/and non-differentiable (Yao, 1999). Secondly, it leads to the problem of vanishing gradient if network has more than three layers (Bartlett & Downs, 1990). These shortcoming of LM and ability of swarm intelligence based optimization techniques to deal with combinatorial and continuous optimization problems have motivated us to use nature inspired and swarm intelligence based methods to determine the optimize set of connection weights and bias of neural networks.

Due to weakness of LM to converge locally, it can be shown that optimization of network parameters such as weight and bias are strongly dependent on initial random value. If the initial parameters are located in local search space, there is a chance that networks get trapped in local solution. The local convergence problem can be overcome by applying global search techniques for training neural networks. Considering the drawback of local convergence of LM, in this article we present the combination of particle swarm optimization (PSO) which is a stochastic swarm intelligence based optimization techniques and feed-forward neural network (FFNN) trained with Levenberg–Marquardt back-propagation (LMBP) method for forecasting stock market indices. At the initial stage, PSO is employed to find the initial value of weight and bias of network to reduce the search space. After search space gets minimized, then the obtained weights and bias are assigned as initial parameters for FFNN algorithm.

In certain scenario input data has a wide range of values which reduces the efficiency of the FFNNs. Hence, to scale the data into small range, the data transformation approach such as min–max normalization technique is applied.

In this paper, we proposed a two stage swarm intelligence based hybrid intelligent mechanism that can forecast the 1-day-ahead, 5-days ahead and 10-days ahead close price of stock market by utilizing various technical indicators. The proposed model is developed by combining the discrete particle swarm optimization (DPSO),

particle swarm optimization (PSO) and Levenberg–Marquardt (LM) algorithm for training the feed-forward neural networks (FFNN). In this study, we obtain the joint optimization of topology as well as initial parameters of FFNN in two stages simultaneously. In the first stage, we employ DPSO in order to obtain the optimal topology (number of neurons in input layer and hidden layer) of the network due to its capability to handle the binary variables and in the second stage we employ PSO for evolving the initial weight and bias of FFNN. Finally, the obtained weights and bias and optimal number of neurons in input (optimal feature subset) and hidden layers are used to train the FFNN by using LM algorithm. The proposed approach is named as DPSO-PSO-FFNN. This paper also performs the comparison of the forecasting capability of proposed model with regular FFNN, Elman neural network (ENN) (Ren et al., 2018), adaptive network-based fuzzy inference system (ANFIS) (Jang, 1993) and another hybrid model obtained by combining evolutionary technique such as binary coded genetic algorithm (BCGA) and real coded genetic algorithm (RCGA) with FFNN and termed the model as BCGA-RCGA-FFNN. The predictive ability of proposed model has been verified by employing it to forecast the close price of five stock market indices namely Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index for multiple-horizon (1-day ahead, 5-days ahead and 10-days ahead) forecasting.

The major contributions of this work are summarized as under:

1. Since the only information available in the stock market is daily open, high, low and close prices and volume of share traded. We have created the pool of technical indicators from the available prices and volume by using TA python library. Feature selection is key challenge in soft computing domain. The first issue considered in this study is automatic selection of optimal subset of technical indicators from pool of technical indicators in order to reduce the network size and to attain higher accuracy. We employ the DPSO to obtain the optimal feature subset due to its capability to deal with combinatorial optimization.
2. To automatically determining the optimal number of neurons in hidden layer and optimizing the initial weights and bias of FFNN simultaneously, a swarm intelligence based hybrid ANN model termed as DPSO-PSO-FFNN is developed. The key objectives of the proposed model are, firstly to reduce the risk of being stuck in local minima by gradient based Levenberg–Marquardt (LM) learning algorithm, secondly to overcome the problem of overfitting and underfitting by automatically selecting the optimum number of neurons in hidden layer by DPSO and finally, to attain the higher prediction accuracy in stock price time series forecasting domain. The major contribution of this study is to automatically searching the optimal feature subset, optimal number of neurons in hidden layer and optimal initial weights and bias in FFNN simultaneously, instead of manual trial and error approach.
3. To develop another evolutionary computation based hybrid ANN model termed as BCGA-RCGA-FFNN by combining binary coded genetic algorithm (BCGA) and real coded genetic algorithm (RCGA) with FFNN for automatic search of reduced feature subset, number of neurons in hidden layer and parameters of FFNN simultaneously, in the same stock price time series forecasting domain in

order to perform the comparative analysis. The experimental results verify the superiority of DPSO-PSO-FFNN in comparison to BCGA-RCGA-FFNN.

The rest of paper is structured as follows: Sect. 2 presents the review of previous work. Section 3 introduces the framework for creating the proposed model. Section 4 describes the evaluation metrics and experimental setup. Section 5 demonstrates the experimental results and analysis. Finally, Sect. 6 presents the conclusion.

2 Related work

Stock market forecasting is a challenging task because of several factors that influence the movement of stock market. Researchers have proposed several statistical method and soft computing techniques to forecast stock market (Bisoai & Dash, 2014; Reza et al., 2015; Sands et al., 2015; Ulke et al., 2018; Zhong & Enke, 2017). Among the soft computing techniques, hybrid models are most widely used and accepted methods (Atsalakis & Valavanis, 2009; Kumar et al., 2020). In this section, we focus on related research work that have proposed an artificial neural network and swarm intelligence based hybrid model for forecasting of stock market indices of various emerging markets.

Darwish et al. (2020) have recently presented the comprehensive survey of swarm intelligence (SI) and evolutionary computing (EC) based hybrid ANN model for optimizing the hyper-parameters and architecture of deep neural networks (DNNs). They reported that the hybrid SI and EC based hybrid ANN models are flexible and effective in context to data analytics in large scale. In a very recent work Baldomonis et al. (2020) have focused on the review of hybrid models that utilize neuroevolution techniques for automatic search of the optimized network topology and learning hyper-parameters of DNNs. Liu et al. (2018) adopted a sequential model-based optimization approach for tuning the architecture of complex neural network model while training the surrogate model simultaneously, that guide the search in structure search space. Elsken et al. (2018b) presented the reviews of various studies that focused on the domain of automated neural architecture search (NAS) which guides the researchers through three dimensions viz. search space, search strategy and performance estimation strategy. Elsken et al. (2018c) proposed a Lamarckian evolution technique for automatic searching the multi-objective neural network architecture while dealing with the problem of computational resource constraints. Hu et al. (2020) recently developed a hybrid model by combining the novel nature inspired paradigm namely Improved Harris's hawks optimization (IHHO) algorithm with extreme learning machine (ELM) in order to predict two major stock indices of S&P 500 and DJIA. In this work, IHHO is utilized to obtain the optimize value of connectionist weights and bias of ELM. Rout and Koudjonou (2020) developed a hybrid evolutionary parallel approach comprised of adaptive linear combiner and functional link artificial neural network (FLANN) in combination with differential evolution (DE) to tackle with time series forecasting problem. In this article, the initial parameters of FLANN are evolved by using DE for faster convergence. Nayak et al. (2019) examined the forecasting ability of functional link neural network by

integrating artificial chemical reaction optimization (ACRO) technique and successfully tuned the parameters of the model to foresee the five real stock market prices. In another study Nayak et al. (2020) employed ACRO, PSO and genetic algorithm (GA) to get the optimal parameters of recurrent functional link neural network. Xiong et al. (2015) performed joint optimization of topology as well as parameters of fully complex-valued radial basis function neural networks (FCRBFNNs) by using DPSO and PSO to forecast the interval time series and obtained the promising results. Chan et al. (2019) construct a multilayer network by taking into account linear and non-linear relationship among stocks and combining Pearson correlation, Granger causality and conditional correlation to characterize the non-linear relationship between stocks. Sahoo and Mohanty (2020) applied a hybrid model obtained by combining ANN with gray wolf optimization technique (GWO) on Bombay stock exchange (BSE). In this article, the authors applied GWO in order to tune the parameters of ANN for forecasting the stock price for one-day, fifteen days and thirty days ahead. Gocken et al. (2019) deployed harmony search (HS) optimization technique to optimize the parameters and architecture of ANN, Jordan recurrent neural network, extreme learning machine (ELM) and recurrent ELM. The authors utilized the HS to determine the optimal number of input variables from 44 technical indicators and to tune the parameters which involves the optimal number of hidden layer neurons and context layer neurons as well as determination of appropriate transfer function.

Yang et al. (2019) combined the extreme learning method (ELM) a special case of single-layer FFNNs with Differential evolution (DE) for stock selection including stock price prediction and stock scoring using various fundamental indicators of A-share index of China stock market. The authors obtained the promising result by optimizing the parameters of ELM with DE. Senapati et al. (2018) integrated the Adaline neural network with modified PSO to optimize initial weights and bias of the neural network to foresee stock price of BSE by utilizing stock price time series data. Hu et al. (2018) attempted to forecast the direction of S&P 500 and DJIA indices of US market by using stock prices and Google trends. The authors used the improved sine cosine algorithm (ISCA) to obtain the tuned initial weight and bias of back-propagation neural network (BPNN). Pradeepkumar and Ravi (2017) performed the comparison of traditional volatility forecasting models with quantile regression neural network (QRNN) that is trained with PSO by using the financial time series of S&P 500 and NSE India stock indices and reported that PSO-QRNN yield the better volatility forecast in comparison with traditional approaches. Chiang et al. (2016) created an adaptive decision support system by using wavelet transformation for denoising the input signals obtained from various technical indicators of S&P 500 and BSE Sensex. In this work, the authors applied the input signal to ANN trained by PSO to evolve the initial weights of the networks. Qiu et al. (2016) utilized the financial indicators and macro-economic data to forecast the return of Japanese Nikkei 25 index using the classical back propagation (BP) learning algorithm. The authors used two global search techniques namely GA and simulated annealing (SA) to deal with local convergence issue of BP algorithm.

Gocken et al. (2016) captured the association between 45 technical indicators and Turkish stock market index by creating the hybrid ANN model obtained by

exploiting the capabilities of HS and GA for choosing the most appropriate technical indicators and for determining the optimal number of neurons in hidden layer simultaneously. The optimal subset of feature selected and number of neurons in single hidden layer by HS-ANN are 23 and 17 respectively and by GA-ANN are 26 and 2 respectively. Inthachot et al. (2016) applied GA to get the optimal subset of features from 44 technical indicators and feed the selected features as input to ANN model to foresee the trend in Thailand's SET50 index. Reza et al. (2015) created a four layers multi-agent framework to forecast the Germany's DAX index using fundamental and technical indicators both. The first layer collect data from external sources, second layer preprocess the data, in the third layer BAT algorithm is employed in order to tune the parameters of ANN and fourth layer is used for testing and reporting the results.

A significant number of research works from literature have shown that hybrid models are potential approaches for stock price time series forecasting. Hence, we attempt to model stock price time series forecasting problem using swarm intelligence based hybrid neural network approach.

3 Methodology

This article presents a four-phase framework to create a two stage swarm intelligence hybrid intelligent system to forecast stock price time series. The first phase is computation of technical indicator from stock prices and data pre-processing where we employ a data transformation technique on technical indicators to transform the data in a specified range. In the next phase, we use two stages simultaneously for the joint optimization of network architecture (topology) and initial parameters of FFNN. In the first stage, we apply DPSO for feature selection i.e. to find the optimum number of nodes (input features) in input layer and to obtain the optimum number of neurons in hidden layer and in the second stage we apply PSO as a global search technique to obtain the initial weights and bias of feed-forward neural networks (FFNNs). In the third phase, we create the FFNN by assigning the optimized feature set, optimum number of neurons in hidden layer and initial value of weight and bias obtained in previous phase and train the FFNN using Levenberg–Marquardt back-propagation (LMBP) algorithm (More, 1978). In the last phase, we examine the performance of proposed model using six performance metrics and then generate the forecasted stock prices. Figure 1 depicts the general architecture of proposed work. Each phase is explained in detailed in next sections.

3.1 Technical Indicators and Pre-processing

In the first phase, the technical indicators for each trading day are computed from the daily OHLC data that includes open (*OP*), high (*HI*), low (*LO*) and close (*CL*) stock prices and volume (*V*) of shares traded. Some of the technical indicators are taken from prior research (Inthachot et al., 2016; Kumar et al., 2016; Shynkevich et al., 2017) and others are selected in accordance with exploratory analysis and

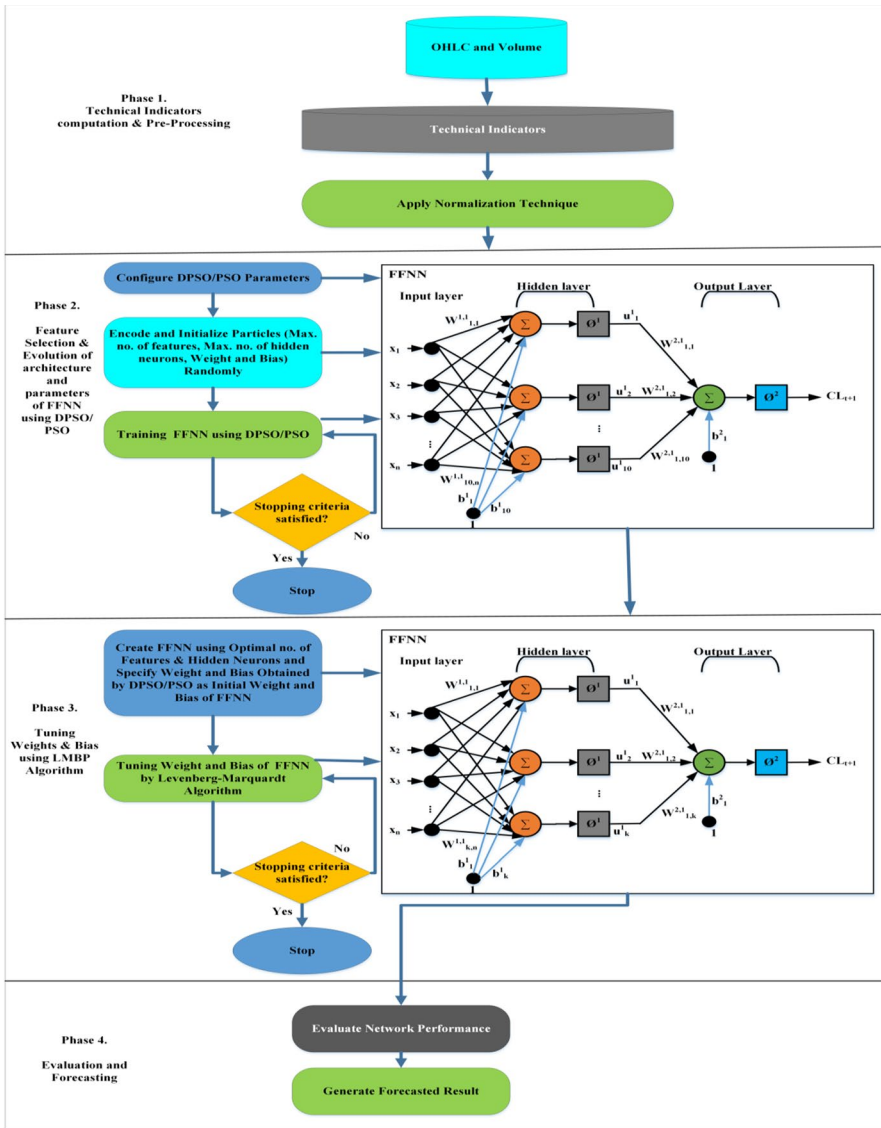


Fig.1 Flowchart of proposed work

their popularity in relevance to technical analysis. Table 1 gives the brief explanation about technical indicators and their mathematical formulations. Different combinations of indicators are used for forecasting the different financial market due to their inbuilt capability. Technical indicators (Technical analysis, Technical indicators) such as simple moving averages (*SMA*), exponential moving averages (*EMA*), moving averages convergence divergence (*MACD*), average directional index (*ADX*) and commodity channel index (*CCI*) are preferable for analyzing upward and

Table 1 Technical indicators description and their formulation

Technical indicator	Description	Formulation
OP_t	Open price at which a share is first traded on opening of market on a given trading day t	—
HI_t	Highest price of a particular share on a specific trading day t	—
LO_t	Lowest price of a share on a given trading day t	—
CL_t	Close price of a share is on a given trading day t	—
$SMA_1(n)$	Trend indicators which smooth out the time series by diminishing the noise from highly chaos stock price time series	$SMA_1(n) = \frac{1}{n} \sum_{x=0}^{n-1} CL_{t-x}$
$EMA_1(n)$	Type of moving average which assigns greater weight to most recent values of time series so that the indicator response more quickly to the recent information	$EMA_1(n) = \alpha (CL_t - EMA_{t-1}) + EMA_{t-1}$ $\alpha = \frac{2}{n+1}$
$MACD_1(n, k)$	Trend following momentum based technical indicator which indicates the association between fast (n) and slow (k) moving averages	$MACD_t = EMA_1(n) - EMA_1(k)$
$ADX_1(n)$	Determine the strength of trend in stock market instead of trend direction. Crossover between DI^+ and DI^- indicates the trend in stock price $ADX > 25$ indicates strong trend and $ADX < 20$ indicates weak trend in market	$ADX_t = \frac{ DI^+ - DI^- }{EMA(DI^+, n) + ATR(n)}$ $DI^+ = EMA(DI^+, n) / ATR(n)$ $DI^- = EMA(DI^-, n) / ATR(n)$
$CCI_1(n)$	Measures the variation between stock price and moving averages over specific period ago to determine direction and strength of trend and determine whether a stock is approaching a level of being overbought or oversold	$CCI_1(n) = \frac{TP_t - SMA_1(TP, n)}{0.015 * \sum_{i=1}^n TP_{t-i+1} - SMA_1(TP, n) / n}$
$MTM_1(n)$	Estimates the change in price of shares over past n -trading days and indicate the rate at which price is changing to determine trend in market	$MTM_1(n) = CL_t - CL_{t-n}$
$ROC_1(n)$	Based on momentum which indicates the percentage change in price over specific period of time for identifying overbought and oversold signals in the market	$ROC_1(n) = \frac{CL_t - CL_{t-n}}{CL_{t-n}} * 100$
$RSI_1(n)$	Measure the magnitude of relative gain to relative loss in order to identify the overbought and oversold signals during analysis. It is bounded between 0 and 100. When RSI is above 70 it gives overbought signal and RSI value below 30 gives oversold signals	$RSI_1(n) = 100 - \frac{100}{1 + \frac{\sum_{i=1}^n UP_{t-i}/n}{\sum_{i=1}^n DN_{t-i}/n}}$
$TSI_1(n, k)$	Identify the oversold and overbought levels to indicate the reversal in trend direction using signal line crossover and depicts the strength of trend through divergence	$TSI_1(n) = \frac{EMA(EMA(PC, n), k)}{EMA(EMA(PC, n), k)}$

Table 1 (continued)

Technical indicator	Description	Formulation
$\%K_t(n)$	Shows the position of current close price with reference to the previous lowest-low range over given period of time. It is bounded indicator and hence it also identifies oversold and overbought signals	$\%K_t(n) = \frac{CL_t - LL_{t-(n-1)}}{HH_{t-(n-1)} - LL_{t-(n-1)}}$
$\%D_t(n)$	Used as signal line with %K indicator and computed as n-day simple moving average of stochastic %K indicator to smooth out the noise	$\%D_t(n) = \frac{\sum_{i=0}^{n-1} \%K_{t-i}}{n}$
$\%R_t(n)$	Compare the position of current close with reference to previous highest-high range in look-back period say n to measure the momentum in stock market	$\%R_t(n) = \frac{HH_{t-(n-1)} - CL_t}{HH_{t-(n-1)} - LL_{t-(n-1)}}$
$ATR_t(n)$	Measures the strength of market volatility. It does not provide any information about directional movement	$ATR_t(n) = \frac{ATR_{t-1} * (n-1) + TR_t}{n}$
$UI_t(n)$	Volatility indicator that gauge the downside risk in stock market. A higher value of UI indicates higher downside risk and small value indicates lower risk in investment	$UI_t(n) = \sqrt{\sum_{i=0}^{n-1} (R_i^2)}$
ADI_t	Assess the forthcoming reversal in stock price time series. It is a volume based indicator which gauges the supply and demand by finding whether trades are actually accumulating (buying) or distributing (selling)	$R_t^i = (CL_t - \max(CL_{t-n})) / \max(CL_{t-n})$ $ADI_t = ADI_{t-1} + \frac{CMFV}{(CL_t - LO_t) - (HI_t - CL_t)}$ $CMFV = \left(\frac{HI_t - LO_t}{(HI_t - LO_t)} \right) * V_t$
OBV_t	It judges the crowd sentiments that assist in forecasting the trend in market. It uses volume of shares traded to predict movement in stock prices	$OBV_t = OBV_{t-1} + \begin{cases} V_t & \text{if } CL_t > CL_{t-1} \\ 0 & \text{if } CL_t = CL_{t-1} \\ -V_t & \text{if } CL_t < CL_{t-1} \end{cases}$

n is 10-day period times ago, $k > n$ is 15-day period ago, α is smoothing factor that decreases exponentially, DI^+ is positive directional indicator and DI^- is negative directional indicators, $DM^+ = (HI_t - HI_{t-1})$ and $DM^- = (LO_{t-1} - LO_t)$ are positive and negative directional movements, $TP = (CL_t + HI_t + LO_t) / 3$ is known as Typical price, UP_t is gain at time t and DN_t is loss at time t, $PC_t = CL_t - CL_{t-1}$ is change in close price, HH_{t-n} and LL_{t-n} are the highest high price and lowest low price in previous n trading day, $TR_t = \max(|HI_t - LO_t|, |HI_t - CL_{t-1}|, |LO_t - CL_{t-1}|)$ is true range, $CMFV$ is current money flow value, $\max(CL_{t-n})$ is maximum close price in last n trading day and V_t is volume of shares traded on a given trading day t

downward trend in the financial market. Indicators like momentum (*MTM*), rate of change (*ROC*), relative strength index (*RSI*), true strength index (*TSI*), are momentum based indicators used to measure the speed at which price changes over certain period of time and to identify the oversold or overbought signals in specific stock. Stochastic oscillators like stochastic %K (*%K*) and William %R (*%R*) are also momentum based indicators used to identify overbought and oversold signals. %D is used as signal line with %K to find trends in market. Average true range (*ATR*) and ulcer index (*UI*) measure the market volatility. Accumulation distribution index (*ADI*) and on-balance volume (*OBV*) are volume based indicators which use the volume of shares traded to measure the crowd sentiments. Overbought and oversold levels are the ranges of prices defined for specific stocks. When the stock prices reach at the level of overbought then there is a chance of downward reversal and when the prices cross the oversold boundary then the stock prices shows upward trend.

3.2 Data Normalization

Pre-processing of data is essential task in soft computing so as to enhance the performance of the proposed model as reliability and accuracy of the model depend on quality of data.

In order to minimize the variance among the values of selected features, data must be transformed to certain range. Mapping of feature values from original scale to smaller scale is an important task in many heuristic approaches particularly when dealing with data-driven forecasting problems (Azadeh et al., 2008). Data normalization is one of the most important transformation approach applied in stock price time series forecasting problems to prevent the learning of unwanted information in data and speed up the learning process (Bisoi & Dash, 2014; Reza et al., 2015).

There are various normalization techniques viz. sigmoid normalization, Z-score normalization and Min–Max normalization (Gocken et al., 2019). We employ the Min–Max normalization technique. Suppose that X_{min} and X_{max} are the minimum and maximum values of an attribute X. Then Min–Max normalization technique transform a value v of X to v^* in a new range $[X_{min}^*, X_{max}^*]$ by computing following transformation function:

$$v^* = \left(\frac{v - X_{min}}{X_{max} - X_{min}} \right) (X_{max}^* - X_{min}^*) + X_{min}^* \quad (1)$$

3.3 Artificial Neural Networks (ANNs)

ANN is a soft computing model that imitates the learning capability of human brain to solve the complex real life problems on the basis of past experience (Basheer & Hajmee, 2000; Haykin, 1999). ANNs are massively parallel adaptive distributed networks made up of basic nonlinear processing units known as neurons that are proposed to abstract and simulate the basic functional capability of human brain in order to acquire some of its computational power (Kohonen,

1988). ANN may be seen as weighted directed graph where the nodes represent the artificial neurons and directed weighted edges constitute the connections among the neurons (Kumar, 2004) as shown in Fig. 2. The neurons in ANN have an efficiency of manipulating and memorizing experimental knowledge and these neurons are arranged systematic into layer structure to create ANN. The ANN mimics the human brain with respect to two aspects: first, intelligence attained by neural network through learning process by utilizing historical data and second, strength (synaptic weight) of inter-neuron connection store the knowledge gained while learning process. ANN is a connectionist system composed of layered structure viz. input layer, one or more hidden layers and output layer. Inputs to the system are provided through input layer. These inputs are propagated through one or more intermediate layers of hidden neurons through inter-neuron connecting links. Every link is associated with a weight which is multiplied with input and result is provided to add which aggregates the weighted input and adds a scalar bias to obtain net output. The net output is then given to the activation (transfer) function which generates the final output of the network. In ANN inter-neuron connectionist weights and biased are adjusted to enhance the performance of network through learning process. Back-propagation (BP) algorithm is mostly used as error-correction technique to train ANN in which error between actual response and network response is computed and propagated in backward direction to modify weights (Goh, 1995). Most of training methods such as Levenberg–Marquardt (LM), Newton method, gradient descent, conjugate gradient and Quasi-Newton method and are gradient based (Basheer & Hajmee, 2000). Among the learning algorithms, a second order method such as Levenberg–Marquardt back propagation (LMBP) (More, 1978) method can be successfully used to obtain more accurate result. The LM method is powerful optimization technique used in ANNs as it provides the procedure to speed-up the training process and convergence of the neural network. Figure 2 shows the four-layered architecture of ANN with n nodes in input layer, q neurons in first hidden layer, r neurons in second hidden layer and k neurons in output layer. Mathematically, ANN is represented by:

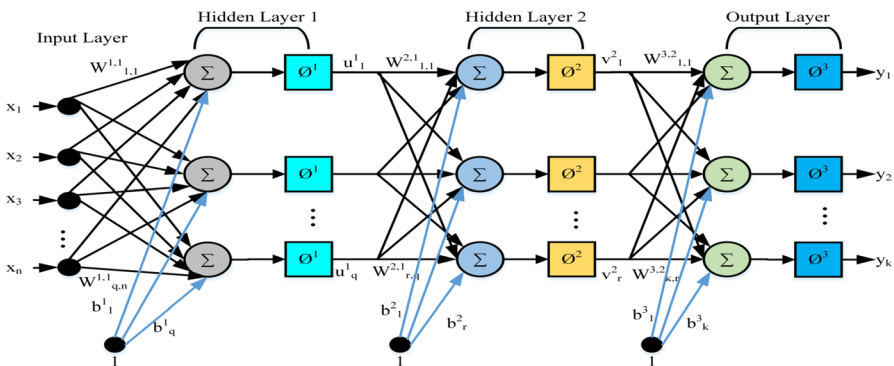


Fig. 2 Three-layered architecture of ANN

$$y_t = \vartheta^3 \left(\sum_{i=1}^k w_{t,m}^{3,2} v_m^2 + b_t^3 \right) \quad (2)$$

$$v_m^2 = \vartheta^2 \left(\sum_{i=1}^q w_{m,i}^{2,1} u_i^1 + b_m^2 \right) \quad (3)$$

$$u_i^1 = \vartheta^1 \left(\sum_{j=1}^n w_{i,j}^{1,1} x_j + b_i^1 \right) \quad (4)$$

where y_t, v_m^2, u_i^1 are outputs, $\vartheta^3, \vartheta^2, \vartheta^1$ are activation functions, b_t^3, b_m^2, b_i^1 are the bias values of output layer, second hidden layer and first hidden layer respectively for $t = 1, 2, \dots, k, m = 1, 2, \dots, r$ and $i = 1, 2, \dots, q$. Inputs to the ANN, $x_j (j = 1, 2, \dots, n)$, are provided to the input layer, $w_{t,m}^{3,2}$ are weights between second hidden layer and output layer, $w_{m,i}^{2,1}$ are weights between first hidden layer and second hidden layer and $w_{i,j}^{1,1}$ are weights between input layer and first hidden layer. The activation functions are used to squash the output to small range and to capture the non-linear relationship present between input and output. The hyperbolic tangent (*Tanh*) transfer function produce values between -1 and 1 and is defined as $Tanh(x) = (e^x - e^{-x}) / (e^x + e^{-x})$. The log-sigmoid transfer (*logsig*) function defines as: $logsig(x) = 1 / (1 + e^{-x})$ generate values between 0 and 1 and pure linear (*purelin*) activation function generates the output by giving the same value provided to it and mathematically it can be written as: $purelin(x) = x$.

3.4 Proposed DPSO-PSO-FFNN Approach

In this study, we focus on joint optimization of network topology (structure) and parameters of FFNN by using discrete particle swarm optimization (DPSO) and particle swarm optimization (PSO).

PSO is a population based stochastic searching method developed by Kennedy and Eberhart (1995) for solving optimization problems in continuous domain. PSO is motivated from the social behavior of bird flocking, fish schooling and swarm of insects. The process starts with population of random solutions that act as particles that form a swarm and search for the optimum solution by flying through the search space. Each particle constitutes a potential solution in D -dimensional search space. During the search process each particle in the swarm is evaluated and updated iteratively. Each particles travel to a new location by changing its velocity in accordance with its own best experience (personal best) together with the intelligence of whole swarm (global best). Initially PSO was developed for solving optimization problems in continuous domain, DPSO was later designed by Kennedy and Eberhart (1997) to expand PSO for solving discrete optimization problems.

When constructing an ANN, numerous parameters should be addressed and there are many ways to develop an ANN. The three-layered FFNN considered

in this study is simple ANN with one hidden layer possessing connections with input layer and output layer. Defining the optimal network topology such as number of input features, number of layers and number of neurons in each layer is crucial issue (Vaisla & Bhatt, 2010) in ANN model development. In order to determine the optimal architecture, the maximal topology of FFNN needs to be specifies in advance which comprises n input nodes, q neurons in hidden layer and k output neurons. It is observed that n and k are problem dependent. Our aim in this study is to forecast the close price of stock market index. Hence the number of node k in output layer is set to 1 and number of nodes n in input layer is determined by feature selection.

Feature selection approach aims at choosing the optimized subset of features and is an essential factor in stock market forecasting domain (Wang & Wang, 2015; Yu & Liu, 2003). Selection of inappropriate feature-subset may lead to a model that cannot capture the actual behavior of stock market. Hence, selection of relevant features is crucial task in ANN model development (Mandziuk & Jaruszewicz, 2011) because of the negative impact of irrelevant features on the efficiency of the model. Determining the optimum number of neurons in hidden is also key factor in artificial neural network because selecting the too few or too many hidden neurons can cause the problem of underfitting or overfitting respectively, hence degrading the accuracy of forecasting model. Therefore, in this study, we use the DPSO for determining the optimum number of input features and hidden neurons and PSO to evolve the weights and bias between various layers in the feed-forward neural network (FFNN). Both DPSO and PSO are used simultaneously for joint optimization of network architecture and parameters of FFNN creating a hybrid model named as DPSO-PSO-FFNN. Figure 3 shows the principal architecture of proposed DPSO-PSO-FFNN hybrid model. As depicted in figure, DPSO/PSO has three main steps: initialization, evaluation and update. Before discussing these operations in details, the coding issue needs to be considered.

1. Coding

Representation of candidate solution (particles) is most important operation in swarm intelligence based approaches. In this study, potential solution is represented by encoding the structure and parameters of FFNN as the particles of the swarm. Each particle is composed of three components: (i) feature selection (number of input nodes) (ii) determination of number of neurons in hidden layer (iii) evolving initial weight and bias. Figure 3 depicts the solution representation. Due to the nature of DPSO to deal with binary variables, optimum number input features and number of hidden neurons are determined by DPSO. As specified in Fig. 4 the input feature x_i ($i = 1, 2, 3 \dots n$) is represented by the control bit which is equal to 1 if the feature is selected otherwise it is 0. In the similar manner, hidden nodes H_j ($j = 1, 2, 3 \dots q$) exists in the network if corresponding bit is 1, otherwise 0. If the corresponding bit is set to zero, its related parameters are not taken into consideration because input feature and hidden node are not present in

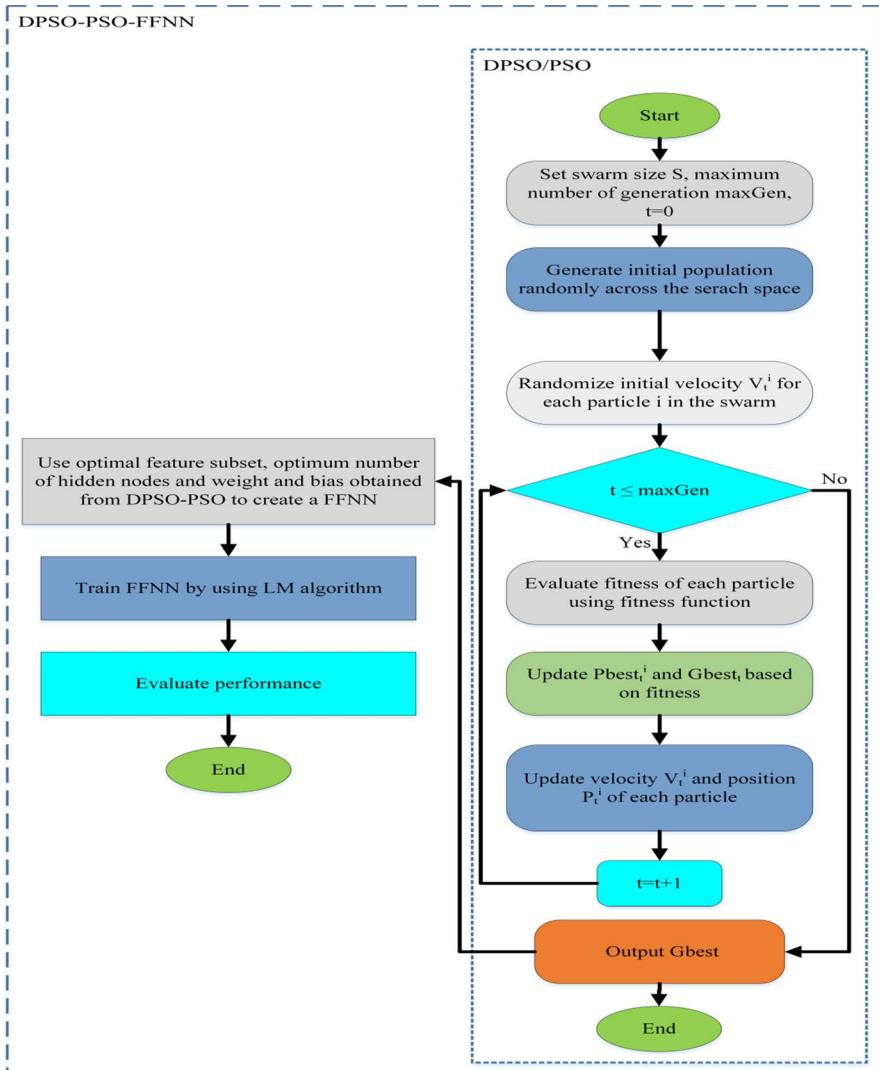


Fig. 3 Architecture of hybrid swarm intelligence based neural network

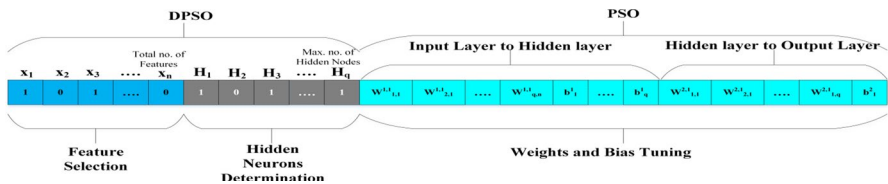


Fig. 4 Solution representation

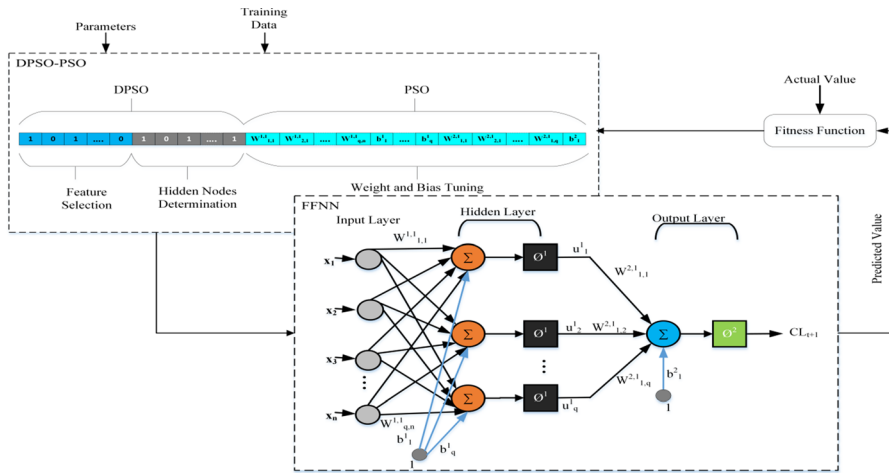


Fig. 5 DPSO-PSO-FFNN iterative process for joint optimization of network topology and parameters of FFNN

the network. Each particle in the PSO represents the weights and bias in different layers of FFNN. Figure 5 shows the iterative process for selecting optimum feature subset, determining the optimum number of neurons in hidden layer and tuning initial weight and bias using DPSO-PSO-FFNN.

2. Initialization

The dimensionality D of search space in DPSO-PSO is equal to number of parameters in FFNN to be optimized. The initial solution (particles) are FFNN with maximal structure and the initial weights and bias at $t=0$ are generated randomly between $[-1, 1]$. The total number of features N fed to the input layer is 20 and maximum number of neurons in hidden layer is set to be 10 by rule of thumb (Xiong et al., 2015). The initial velocity V_0^i of each particle is allocated the value of zero. To implement the DPSO and PSO, various parameters such as inertia coefficient (ω), personal coefficient (k_1), social coefficient (k_2), swarm size (S), maximum number of generations (maxGen) and two random numbers (r_1 and r_2) distributed uniformly in $U(0,1)$ should be properly determined in advance. A time varying inertia coefficient ω (Xiong et al., 2015), which decrease linearly from ω_{max} to ω_{min} is used, which is given by:

$$\omega = \omega_{max} - \frac{(\omega_{max} - \omega_{min})t}{T} \tag{5}$$

where t is current iteration and T is maximum number of iteration.

3. Evaluation

The fitness of each particle in the swarm is evaluated by computing root mean squared error between the actual value and predicted value as shown in Fig. 4. Given the training data $D = \{X_i, T_i\}_{i=1}^N$ where, T_i is the target value or actual value for the input feature, X_i . The fitness function or cost function is expressed as follows:

$$f(s, D) = \sqrt{\frac{1}{N} \sum_{i=1}^N (T_i - Y_i)^2} \quad (6)$$

where N is total number of samples in training data and Y_i is the output (predicted value) of input pattern X_i obtained from FFNN using the structure s (input features and hidden neurons) and assigning the parameters (weight and bias) generated at iteration t .

4. Update

Based on the fitness value of each solution (particle), personal best ($Pbest$), the best solution achieved so far by particle i and global best ($Gbest$), the best position attained so far by any other particle in the swarm are modified at each iteration and according to personal best and global best the velocity and position of each particle is also updated in every iteration.

In this work, we attempt to evolve both the structure (input features and number of hidden neurons) and parameters (weights and bias) of FFNN. Both structure and parameters of FFNN are encoded as candidate solution (particle) as shown in Fig. 3. For each solution (particle) in the swarm, the discrete (structure) and continuous (parameters) parts of new solution are evolved by DPSO and PSO respectively.

For the structure part s of the solution, let ${}^sP_t^i$ and ${}^sV_t^i$ represent the position and velocity of particle i at iteration t , respectively. The velocity and position of particle i in iteration $t + 1$, are updated by:

$${}^sV_{t+1}^i = \omega \times {}^sV_t^i + k_1 r_1 \times ({}^sPbest^i - {}^sP_t^i) + k_2 r_2 \times ({}^sGbest^i - {}^sP_t^i) \quad (7)$$

$${}^sP_{t+1}^i = \begin{cases} 1, & \text{if } rand < S({}^sV_{t+1}^i) \\ 0, & \text{Otherwise} \end{cases} \quad (8)$$

where $S({}^sV_{t+1}^i) = \left| 2/\pi \times \arctan((\pi/2) \times {}^sV_{t+1}^i) \right|$ is the probability of velocity that bit value in position vector is equal to 1 or 0 (Mirjalili & Lewis, 2013); $rand$ is real random number distributed uniformly in $U(0,1)$; sPbest and sGbest are personal best and global best of the structure (input features and hidden neurons) optimizing part of the solution.

To prevent $S({}^sV_{t+1}^i)$ approaching either 1 or 0, a constant velocity ${}^sV_{max}$ is set to 6 (Mirjalili & Lewis, 2013; Yeh, 2009) such that ${}^sV_{t+1}^i \in [-6, 6]$, and following relationship is applied:

$${}^s V_{t+1}^i = \begin{cases} {}^s V_{max} & \text{if } {}^s V_{t+1}^i > {}^s V_{max} \\ -{}^s V_{max} & \text{if } {}^s V_{t+1}^i < -{}^s V_{max} \\ {}^s V_{t+1}^i & \text{otherwise} \end{cases} \quad (9)$$

For the parameter (weights and bias) part p of the solution, the position ${}^p P_t^i$ and the velocity ${}^p V_t^i$ are updated by:

$${}^p V_{t+1}^i = \omega \times {}^p V_t^i + k_1 r_1 \times ({}^p Pbest^i - {}^p P_t^i) + k_2 r_2 \times ({}^p Gbest^i - {}^p P_t^i) \quad (10)$$

$${}^p P_{t+1}^i = {}^p P_t^i + {}^p V_{t+1}^i \quad (11)$$

where ${}^p Pbest$ and ${}^p Gbest$ are personal best and global best of the parameters tuning parts of the solution.

The structure and parameters part of the current solution are updated simultaneously in accordance with Eq. (8) and Eq. (11) of DPSO and PSO respectively. The process of updating $Pbest, Gbest$, velocity and position of particle continues until the count of iteration is equal to maximum generation number (maxGen) or tolerance reached to a minimum preset value. The final global best ($Gbest$) particle is the optimal solution comprises of optimal number of input features and hidden neurons and tuned weights and bias. The structure and parameters obtained by DPSO-PSO-FFNN are used to create a FFNN which is further trained by LM algorithm.

3.5 GA-ANN Forecasting Model

Genetic algorithm (GA) is metaheuristic and stochastic optimization technique inspired by the Darwinian natural selection and process of biological evolution (Holland, 1975). The process of GA begins by creating an initial random population of chromosomes. Each chromosome in GA represents the potential solution corresponding to the problem. During each generation, GA iteratively evolves the population of individual solution by applying various genetic operations namely selection, crossover and mutation. Over each successive generation, the population evolves towards the optimal solution.

In this study, we use the binary coded GA (BCGA) (Holland, 1975) for determining the optimum number of input features and hidden neurons and real coded GA (RCGA) (Wright, 1991) to evolve the weights and bias between various layers in the feed-forward neural network (FFNN). Both BCGA and RCGA are used simultaneously for joint optimization of network topology and parameters of FFNN developing a hybrid model named as BCGA-RCGA-FFNN. Figure 6 depicts the systematic approach of GA based feature selection, determination of hidden neurons and optimization of weight and bias of FFNN. The various steps of BCGA-RCGA-FFNN are as follows:

Step 1 Chromosome representation: In this study, potential solution is represented by encoding the structure and parameters of FFNN as the chromosome of

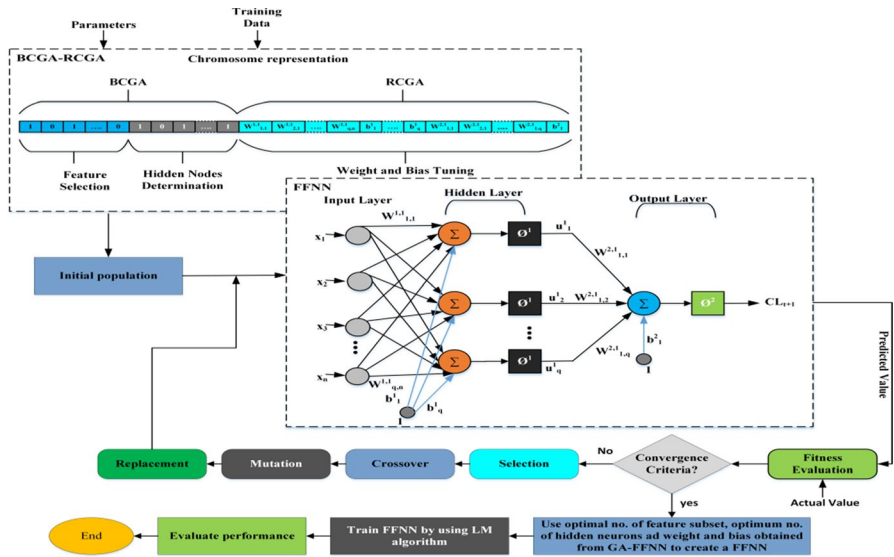


Fig. 6 BCGA-RCGA-FFNN iterative process for joint optimization of network topology and parameters of FFNN

GA. Each chromosome is comprised of three components: (1) feature selection (number of input nodes) (2) determination of number of neurons in hidden layer (3) evolving initial weight and bias. We use BCGA for optimum feature subset selection and optimum number of neurons in hidden layer where gene of chromosome is encoded as 1 if corresponding feature is selected otherwise 0. Similarly, gene is coded as 1 if the hidden neuron is selected otherwise 0. To evolve the weight and bias we use RCGA where chromosome is represented by randomly generated value between -1 and 1 .

Step 2 Initial population generation: The initial population of chromosome is randomly generated equal to population size.

Step 3 Fitness evaluation: The fitness value of each chromosome of the population is measured by computing the root mean squared error using Eq. (6) between the actual value and forecasted value obtained from FFNN using the chromosome C (input features and hidden neurons) and assigning the parameters (weight and bias) generated at generation t .

Step 4 Convergence criteria: The iterative process of GA is stopped when number of generation reaches to the value of maximum generations.

Step 5 Selection: In this step, chromosomes known as parents are chosen on the basis of selection operator to create new better offspring (solution) for the next generations. In this study, roulette wheel selection method is utilized to select parents' chromosomes. The roulette wheel selection operation is based on the probability of selection of each individual which is computed by dividing the fitness of each individual with the sum of fitness of all individuals in the population and individual is selected by rotating the wheel randomly which is partitioned based on the cumulative probability of each individual.

Step 6 Crossover: The crossover operator generates the offspring by combining the information from the pair of parents. In BCGA, firstly a point called crossover point is randomly selected and then based on the result of roulette wheel selection, one-point crossover, two-point crossover and uniform crossover is performed. In crossover, the information after the crossover point of two parents' chromosome is swapped to generate two new offspring. In RCGA, we use the blend crossover technique (Haupt & Haupt, 2004) which combines the value of genes of two parents' chromosomes on the basis of random parameter (γ) to generate two offspring.

Step 7 Mutation: The mutation operation is used to preserve the genetic diversity between successive generations of population. The mutation operator generates the new children by randomly changing one or more gene in the parent's chromosome. In case of BCGA, flipping mutation operator is used where randomly selected gene value in the parent's chromosome are flipped to generate new offspring. In RCGA, we use random mutation where mutated offspring is created in accordance with mutation probability (Haupt & Haupt, 2004).

Step 8 Replacement: Replace the current population with the newly generated children for the next generation. In this study, we use truncation procedure where chromosomes are sorted on the basis of their fitness and only the best members equal to population size are retained and rest are truncated.

Step 9 Go to step 3.

The final best chromosome obtained after the convergence criteria is satisfied is the optimal solution comprises of optimal number of input features and hidden neurons and tuned weights and bias. The structure and parameters obtained by BCGA-RCGA-FFNN are used to create a FFNN which is further trained by LM algorithm. Finally, the hybrid network is evaluated using various metrics.

4 Evaluation Metrics and Implementation of Forecasting Model

In this section we discuss the various performance metrics used for evaluating the forecasting ability of proposed model and experiment setup for the proposed method.

4.1 Evaluation Metrics

To evaluate and compare the forecasting efficiency and robustness of proposed model we have used various metrics including mean squared error, (MSE) root mean square error ($RMSE$), mean arctangent absolute percentage error ($MAAPE$), symmetric mean absolute percentage error ($SMAPE$), Theil's Inequality Coefficient ($UofTheil$) and average relative variance (ARV). The mathematical formulations of these metrics are defines as follows:

The $RMSE$, defined in Eq. (13), is the square root of mean of squared error (MSE). Error is defined as deviation between actual value and forecasted value. In this metrics, square root is taken to bring the mean square error to the scale of data

(Botchkarev, 2018). The *RMSE* value close to zero indicates that the predicted value agrees with the actual value.

$$MSE = \frac{1}{N} \sum_{i=1}^N (A_i - P_i)^2 \quad (12)$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (A_i - P_i)^2} \quad (13)$$

where A_i is the actual value and P_i is the forecasted value of i th observation of test dataset generated by the prediction model and N is the total number of observations in test dataset or training data set.

The *MAAPE*, Eq. (14), is modification of mean absolute percentage error (MAPE) which evaluate the arctangent of mean of absolute percentage error between observed value and forecasted value (Kim & Kim, 2016). This metrics is easy to interpret as it gives error in terms of percentages. The required value of *MAPE* is zero, so closer the value of this measure to zero shows that prediction performance is higher. Due to the absolute percentage error, this metric overcome the issue of positive and negative values nullifying each other and hence avoiding misleading results.

$$MAAPE = 100 \times \frac{1}{N} \sum_{i=1}^N \arctan\left(\left|\frac{A_i - P_i}{A_i}\right|\right) \quad (14)$$

The *SMAPE*, defined in Eq. (15) overcome the drawback of MAPE that it leads to high penalty on positive errors than negative errors (Hyndman & Koehler, 2006).

$$SMAPE = 100 \times \frac{1}{N} \sum_{i=1}^N \left(\frac{|A_i - P_i|}{(|A_i| + |P_i|)/2} \right) \quad (15)$$

The Theil's Inequality coefficient (Theil, 1966) given by Eq. (16), determines how much a predicted time series is close to the actual time series. This metric relates the performance of model with a random walk (RW) model (Hyndman & Koehler, 2006). If, U of Theil > 1 , then the model has worst performance with respect to RW model. If, U of Theil $= 1$, then the model's performance is equal to RW model. If, U of Theil < 1 , then the model perform better than RW model. Hence, the ideal value of U of Theil tends to zero.

$$U \text{ of Theil} = \frac{\sqrt{\frac{1}{N} \sum_{i=1}^N (A_i - P_i)^2}}{\sqrt{\frac{1}{N} \sum_{i=1}^N (A_i)^2 + \frac{1}{N} \sum_{i=1}^N (P_i)^2}} \quad (16)$$

The average relative variance (*ARV*) metrics is defined by the Eq. (17). This metric relates the performance of model with the mean considering as forecasted value

(Hyndman & Koehler, 2006). The performance of model is acceptable if, $ARV < 1$, and closer the value to zero, implies that the model tends to be accurate.

$$ARV = \frac{\sum_{i=1}^N (P_i - A_i)^2}{\sum_{i=1}^N (P_i - \bar{A})^2} \quad (17)$$

4.2 Implementation of Proposed Model

The simulation is performed on a system having 2.10 GHz i3 processor with 4 GB RAM and the proposed models are implemented in Matlab 2019b.

1. Experimental data description

To demonstrate and compare the forecasting ability of proposed model, we have used historical data of five major stock indices of developed and emerging economies viz. Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index for a period and number of observations shown in Table 2. The data examined in this study is obtained from yahoo finance. Each dataset is divided into three subsets, viz. training set (70%), validation set (15%) and testing set (15%). Training dataset is utilized to learn the parameters of the proposed model. Testing dataset is utilized to test and compare the forecasting capability of proposed model and validation dataset is employed to assess the model generalization capability and to halt training when successive iterations stops improving the result.

2. Implementation of swarm intelligence based neural network

In the first phase, we perform the technical analysis by using OHLC (open price, high price, low price, close price) data and volume of shares traded in order to compute the technical indicators. In this paper, a set of 16 technical indicators along with four daily prices are used to generate a feature set of 20 indicators which are passed to data pre-processing module. These technical indicators are computed by using TA python library (Technical analysis library in python). In the data preprocessing step,

Table 2 Description of stock market index data

Stock index	From	To	No. of samples	Train set	Validation set	Test set
Nifty 50	Jan 4, 2010	Sept 30, 2019	2417	1691	363	363
Sensex	Jan 2, 2010	Dec 31, 2019	2439	1707	366	366
S&P 500	Jan 2, 2010	Dec 30, 2019	2496	1748	374	374
DAX	Jan 4, 2010	Dec 30, 2019	2512	1758	377	377
SSE Composite Index	Jan 4, 2010	Dec 31, 2019	2410	1686	363	361

we employ Min–Max normalization method to map the original data in the range $[-1, 1]$ in order to maintain the actual relationship among the original features set.

In the second phase, we apply DPSO/BCGA and PSO/RCGA simultaneously to determine the optimum number of features and number neurons in hidden layer and to evolve the initial weight and bias of feed-forward neural networks (FFNN). To obtain the optimized network topology with minimum error, the network structure comprises number of hidden layers, number of neurons in each layer, type of transfer function and different parameters of PSO has been investigated. The FFNN with maximal topology (20-10-1) i.e. 20 features (nodes) in input layer, 10 neurons in hidden layer and 1 neuron in output layer is used. In the next phase, we apply LMBP algorithm (More, 1978) to train the FFNN created by using the optimized feature subset, optimum number of neurons in hidden layer and tuned initial weights and bias obtained in previous stage. The optimized parameters of the proposed model are depicted in Table 3. The dimension of each particle (or chromosome) in population is equal to number of network's parameters to be optimized. Finally we evaluate the model and obtained the 1-day ahead, 5-days ahead and 10-days ahead stock prices for five stock indices.

5 Experimental Results and Discussion

Deciding the input features for stock market forecasting problem is a challenging task. Firstly the optimum feature subset is chosen for ANN model. In regular FFNN we applied all the 20 features explored in this study. Tables 4 and 5 depicts the initial feature set and outcome of selected features of five indices. In DPSO-PSO-FFNN and BCGA-RCGA-FFNN model, we minimize the set of feature to an optimal feature subset. Among the 20 features, the optimal feature subset selected by DPSO-PSO-FFNN is 9, 6, 11, 3 and 5 features for 1-day ahead forecasting, 8, 7, 10, 11 and 11 features for 5-days ahead forecasting and 7, 9, 10, 10 and 10 features for Nifty

Table 3 Tuned parameters of the model

DPSO/PSO		BCGA/RCGA		FFNN	
Parameter	Value	Parameter	Value	Parameter	Value
Population size	80	Population size	80	Network topology (input-hidden-output)	20–10–1
Max. inertia coefficient, ω_{max}	0.9	Crossover rate	0.7	Activation function	Tanh
Min. inertia coefficient, ω_{min}	0.4	Mutation rate	0.1	Output activation function	Purelin
Personal coefficient, k_1	1.2	gamma	0.4	Learning rate, lr	0.01
Social coefficient, k_2	1.2	Selection pressure, beta	8	Momentum constant, mc	0.9
Max. velocity, V_{max}	6	maxGen	100		
maxGen	100	–	–	–	–
Tolerance	10^{-8}	–	–	–	–

Table 4 Initial feature set and result of feature selection

	Technical indicators				Nifty 50				Sensex				S&P 500			
	Is selected? (1:Yes) (0:No)				Is selected? (1:Yes) (0:No)				Is selected? (1:Yes) (0:No)				Is selected? (1:Yes) (0:No)			
	1-day	5-days	10-days	10-days	1-day	5-days	10-days	10-days	1-day	5-days	10-days	10-days	1-day	5-days	10-days	10-days
	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA
1	0	0	1	1	0	0	0	0	1	1	0	0	1	0	1	0
2	0	0	1	0	0	1	0	1	0	1	0	0	1	1	0	0
3	0	0	1	0	1	0	0	0	0	0	1	1	0	0	0	1
4	1	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1	1	0	1	1	0	0	1	0
6	0	0	0	1	0	1	1	1	1	1	0	0	1	0	1	0
7	1	1	0	1	0	1	1	1	1	1	0	1	1	0	1	1
8	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0
9	1	0	1	0	0	1	1	1	1	0	1	1	0	1	1	0
10	0	0	1	0	0	1	1	1	1	0	1	1	0	1	0	0
11	0	1	1	0	0	0	1	1	0	1	0	1	0	1	1	1
12	1	0	1	0	0	0	0	0	0	1	0	1	1	1	0	1
13	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0
14	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0
15	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
16	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0	0
17	0	1	0	1	0	1	1	0	1	1	1	0	0	1	0	1

Table 4 (continued)

Technical indicators	Nifty 50			Sensex			S&P 500			
	Is selected? (1:Yes) (0:No)			Is selected? (1:Yes) (0:No)			Is selected? (1:Yes) (0:No)			
	1-day	5-days	10-days	1-day	5-days	10-days	1-day	5-days	10-days	
	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA
18 $UI_t(10)$	0	0	0	1	1	1	0	0	0	0
19 ADI_t	1	1	0	0	1	0	1	0	1	0
20 OBV_t	0	0	1	1	1	1	0	1	1	1

Table 5 Initial feature set and result of feature selection

	Technical indicators	DAX						SSE Composite Index					
		Is selected? (1:Yes) (0:No)						Is selected? (1:Yes) (0:No)					
		1-day		5-days		10-days		1-day		5-days		10-days	
	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	
1	OP_t	1	0	1	1	1	1	1	1	1	0	1	
2	HI_t	0	1	1	1	1	0	1	0	1	0	1	
3	LO_t	0	0	0	0	0	0	0	0	0	1	1	
4	CL_t	1	1	0	1	0	1	0	1	1	0	0	
5	$SMA_t(10)$	0	1	1	1	1	0	1	1	1	1	1	
6	$EMA_t(10)$	0	0	0	0	1	0	1	1	0	1	1	
7	$MACD_t(10,15)$	0	0	1	0	0	1	0	1	1	1	0	
8	$ADX_t(10)$	0	0	0	0	1	0	1	0	0	0	0	
9	$CCI_t(10)$	0	1	1	0	0	1	1	1	1	0	0	
10	$MTM_t(10)$	0	1	1	1	0	0	0	1	1	1	1	
11	$ROC_t(10)$	0	0	0	0	1	0	0	1	1	0	1	
12	$RSI_t(10)$	0	1	1	0	1	1	0	1	0	1	0	
13	$TSI_t(10,15)$	0	1	1	0	0	0	0	0	0	0	0	
14	$\%K_t(10)$	0	0	1	0	0	0	0	0	0	1	0	
15	$\%D_t(10)$	0	0	0	0	0	0	0	0	1	1	0	
16	$\%R_t(10)$	0	0	0	0	1	0	0	0	0	1	0	
17	$ATR_t(10)$	0	0	0	1	0	0	1	0	0	0	0	

Table 5 (continued)

	Technical indicators	DAX						SSE Composite Index					
		Is selected? (1:Yes) (0:No)						Is selected? (1:Yes) (0:No)					
		1-day		5-days		10-days		1-day		5-days		10-days	
		DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA	DPSO	BCGA
18	$UI_t(10)$	0	0	1	1	1	0	0	0	0	0	0	0
19	ADI_t	0	1	0	0	1	1	0	0	1	0	1	0
20	OBV_t	1	0	1	1	0	0	0	0	1	0	1	0

50, Sensex, S&P 500, DAX and SSE Composite Index respectively whereas BCGA-RCGA-FFNN has selected 8, 11, 8, 8 and 7 features for 1-day ahead forecasting, 10, 10, 8, 8 and 11 for 5-days ahead forecasting and 7, 9, 8, 5 and 7 features for 10-days ahead forecasting as optimal feature subset for Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index respectively.

Table 6 shows the minimal neural network topology selected by DPSO-PSO-FFNN and BCGA-RCGA-FFNN model for various stock indices for 1-day, 5-days and 10-days ahead forecasting. In regular FFNN, we used 10 neurons in hidden layer arbitrarily. DPSO-PSO-FFNN model has selected 2, 3, 3, 3 and 3 neurons in hidden layer for 1-day ahead forecasting, 2, 2, 2, 5 and 3 hidden neurons for 5-days ahead forecasting and 3, 5, 4, 4 and 2 hidden neurons for 10-days ahead forecasting for Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index respectively whereas BCGA-RCGA-FFNN has selected 2, 2, 3, 2 and 4 neurons in hidden layer for 1-day ahead forecasting, 2, 2, 2, 2 and 2 hidden neurons for 5-days ahead as well as 10-days ahead forecasting for Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index respectively. DPSO-PSO-FFNN model has selected 9, 6, 11, 3 and 5 input nodes (features) for 1-day ahead forecasting, 8, 7, 10, 11 and 11 input nodes for 5-days ahead forecasting and 7, 9, 10, 10 and 10 input nodes as shown in Table 4 as input to FFNN for Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index respectively whereas BCGA-RCGA-FFNN has selected 8, 11, 8, 8 and 7 nodes in input layer for 1-day ahead forecasting, 10, 10, 8, 8 and 11 input nodes for 5-days ahead forecasting and 7, 9, 8, 5 and 7 input nodes for 10-days ahead forecasting for Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index respectively. Hence, by reducing the number of features in input node and number of neurons in hidden layer, we have successfully reduced the number of parameters in proposed model thereby reducing the computational complexity of the model.

To evaluate the 1-day, 5-days and 10-days ahead forecasting ability of proposed model named as DPSO-PSO-FFNN, we have computed six metrics defined in Sect. 4 and compare its performance with ANFIS, ENN, regular FFNN and BCGA-RCGA-FFNN model for multiple-horizon forecasting. Table 7 depicts the 1-day ahead forecasting performance of five models for forecasting the closing price of five indices such as Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index using test dataset. The MSE, RMSE, MAAPE, SMAPE, Theil's U and ARV values of DPSO-PSO-FFNN for Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index are all minimum which make it clear that DPSO-PSO-FFNN has successfully improved over other forecasting techniques considered in this study. From the Theil's U metric, it is observed that DPSO-PSO-FFNN and BCGA-RCGA-FFNN both are not a RW model for five datasets and perform better than RW model. In addition, performance metrics values for FFNN are all larger than other two hybrid models. Hence, swarm intelligence approaches and evolutionary algorithms have a capability to optimize the parameters of regular FFNN for time series forecasting problems.

The 1-day ahead forecasting of closing price of Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index for test dataset using proposed model are displayed in Figs. 7, 8, 9, 10 and 11 respectively. It is observed from the plots of five indices that

Table 6 Selected network topology

Model	Network topology (Input-Hidden-Output)					
	1, 5, 10-days		5-days		10-days	
	1-day	DPSO-PSO-FFNN	BCGA-RCGA-FFNN	DPSO-PSO-FFNN	BCGA-RCGA-FFNN	DPSO-PSO-FFNN
Stock Index						
Nifty 50	20-10-1	9-2-1	8-2-1	8-2-1	10-2-1	7-3-1
Sensex	20-10-1	6-3-1	11-2-1	7-2-1	10-2-1	9-5-1
S&P 500	20-10-1	11-3-1	8-3-1	10-2-1	8-2-1	10-4-1
DAX	20-10-1	3-3-1	8-2-1	11-5-1	8-2-1	10-4-1
SSE Composite Index	20-10-1	5-3-1	7-4-1	11-3-1	11-2-1	10-2-1
					BCGA-RCGA-FFNN	7-2-1

Table 7 1-day ahead forecasting performance of the proposed model and its comparison with other models

Stock Index	Model	MSE	RMSE	MAAPE (%)	SMAPE (%)	Theil's U	ARV
Nifty 50	ANFIS	2.00E-03	4.89E-02	13.90	3.92	3.69E-02	5.37E-02
	ENN	1.90E-03	4.33E-02	20.03	5.89	5.46E-02	9.60E-03
	FFNN	4.31E-04	2.08E-02	7.84	2.23	1.71E-02	8.02E-04
	BCGA-RCGA-FFNN	3.83E-04	1.96E-02	7.82	2.22	1.61E-02	7.42E-04
Sensex	DPSO-PSO-FFNN	3.60E-04	1.90E-02	7.57	2.14	1.560E-02	6.73E-04
	ANFIS	4.10E-03	6.40E-02	20.15	6.43	5.61E-02	9.76E-01
	ENN	1.30E-03	3.63E-02	20.07	6.12	5.80E-02	8.60E-03
	FFNN	2.97E-04	1.72E-02	8.88	2.59	1.51E-02	6.41E-04
S&P500	BCGA-RCGA-FFNN	2.64E-04	1.63E-02	8.45	2.49	1.43E-02	5.46E-04
	DPSO-PSO-FFNN	2.53E-04	1.59E-02	8.42	2.43	1.40E-02	4.78E-04
	ANFIS	2.70E-03	5.22E-02	15.04	4.64	4.74E-02	1.02E-00
	ENN	1.50E-03	3.91E-02	24.33	7.51	6.94E-02	1.75E-02
DAX	FFNN	3.29E-04	1.82E-02	9.33	2.45	1.65E-02	7.13E-04
	BCGA-RCGA-FFNN	2.75E-04	1.66E-02	8.57	2.28	1.51E-02	6.23E-04
	DPSO-PSO-FFNN	2.68E-04	1.64E-02	8.19	2.15	1.49E-02	6.03E-04
	ANFIS	4.8E-03	6.91E-02	18.71	5.65	6.23E-02	1.32E-00
DAX	ENN	5.06E-02	2.25E-01	40.25	13.33	2.05E-01	1.17E-01
	FFNN	1.02E-02	1.01E-01	26.99	7.94	9.04E-02	1.69E-02
	BCGA-RCGA-FFNN	6.28E-04	2.51E-02	10.39	3.11	2.27E-02	1.14E-03
	DPSO-PSO-FFNN	5.98E-04	2.45E-02	9.59	2.83	2.31E-02	9.80E-04

Table 7 (continued)

Stock Index	Model	MSE	RMSE	MAAPE (%)	SMAPE (%)	Theil's U	ARV
SSE Composite Index	ANFIS	8.40E-03	9.15E-02	13.19	3.47	8.25E-02	1.43E-00
	ENN	1.91E-01	4.37E-01	65.40	19.61	3.19E-01	4.47E-01
	FFNN	1.37E-02	1.17E-01	24.53	6.94	1.07E-01	4.59E-02
	BCGA-RCGA-FFNN	1.00E-03	3.23E-02	8.30	2.42	2.93E-02	8.30E-03
	DPSO-PSO-FFNN	9.32E-04	3.05E-02	7.88	2.25	2.78E-02	8.30E-03

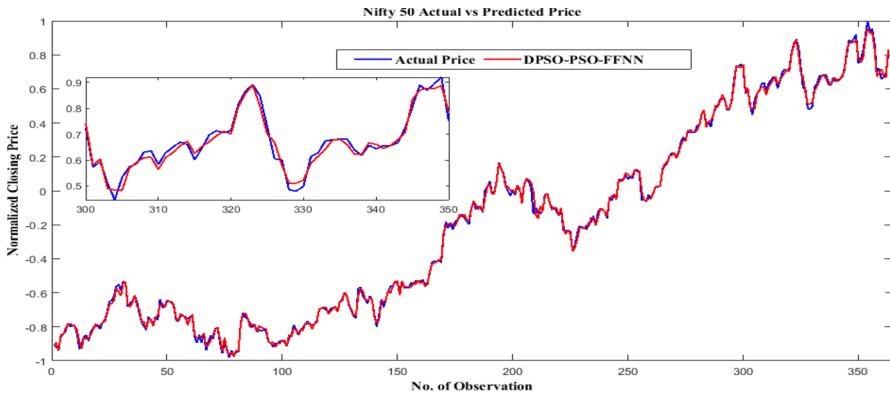


Fig. 7 Forecasting of 1-day ahead close price of Nifty 50

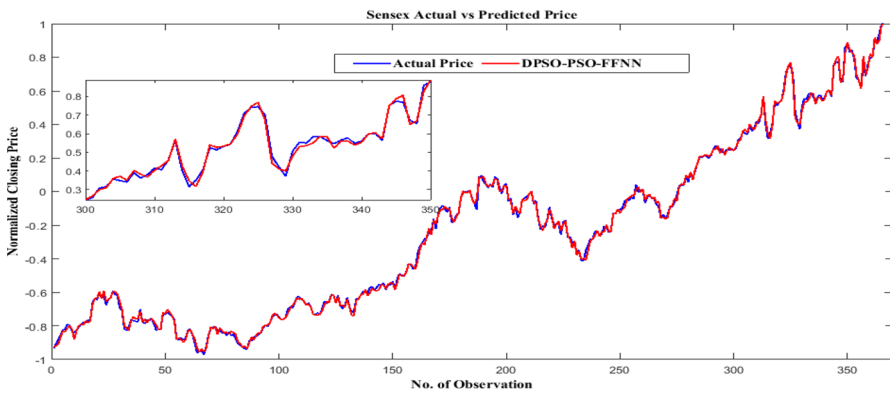


Fig. 8 Forecasting of 1-day ahead close price of Sensex

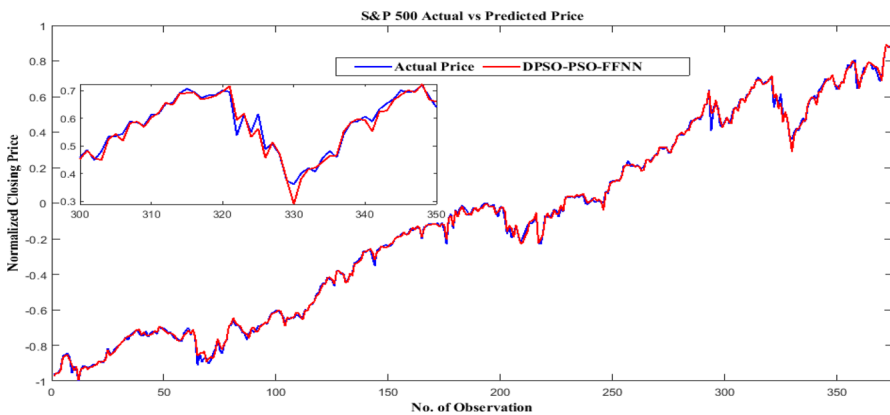


Fig. 9 Forecasting of 1-day ahead close price of S&P 500

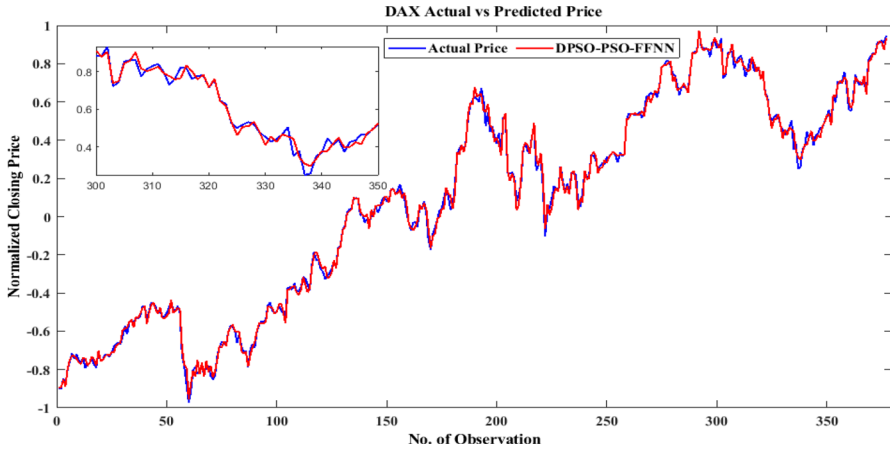


Fig. 10 Forecasting of 1-day ahead close price of DAX

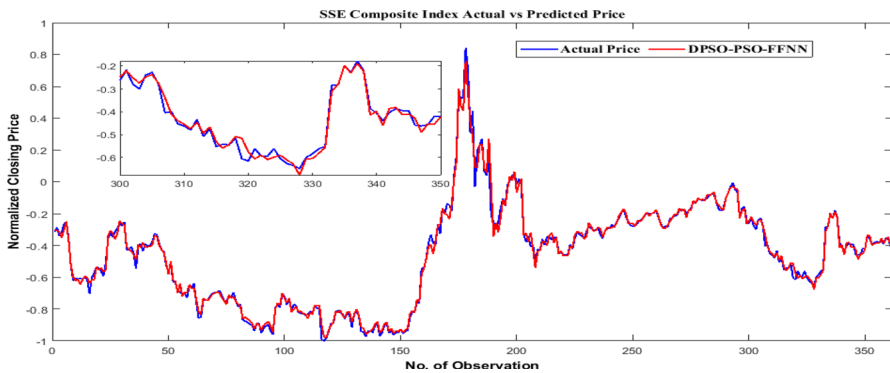


Fig. 11 Forecasting of 1-day ahead close price of SSE Composite Index

the actual price and predicted price by proposed model are almost overlapping and hence the proposed model is superior in terms of forecasting performance.

The 5-days ahead forecasting performance for the five stock indices with five models is presented in Table 8 and the actual versus predicted price by DPSO-PSO-FFNN for 5-days ahead is plotted and shown by the Figs. 12, 13, 14, 15 and 16 for Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index respectively. Similar to 1-day ahead forecasting, on the basis of results we can infer that proposed model performs better than other models for 5-days ahead forecasting. However, the proposed model performs better in 1-day ahead forecasting as compared to the 5-days ahead forecasting for the five stock indices.

The 10-days ahead forecasting performance for the five stock indices with the same five models is presented in Table 9 and the actual versus predicted price by DPSO-PSO-FFNN for 10-days ahead is plotted and shown by the Figs. 17, 18, 19, 20 and 21 for Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index respectively. Similar

Table 8 5-days ahead forecasting performance of the proposed model and its comparison with other models

Stock Index	Model	MSE	RMSE	MAAPE (%)	SMAPE (%)	Theil's U	ARV
Nifty 50	ANFIS	2.20E-03	4.72E-02	17.30	5.80	4.05E-02	6.52E-01
	ENN	2.80E-03	5.28E-02	28.65	8.38	7.83E-02	2.41E-02
	FFNN	5.59E-02	2.36E-01	27.13	8.39	1.85E-01	8.37E-01
Sensex	BCGA-RCGA-FFNN	2.00E-03	4.49E-02	17.68	5.89	3.85E-02	3.70E-03
	DPSO-PSO-FFNN	1.90E-03	4.40E-02	17.15	5.55	3.78E-02	3.50E-03
	ANFIS	1.12E-02	1.05E-01	23.80	7.69	9.12E-02	2.31E-00
S&P500	ENN	2.70E-03	5.19E-02	25.79	7.82	7.53E-02	1.23E-02
	FFNN	3.29E-02	1.81E-01	21.56	6.39	1.43E-02	4.22E-01
	BCGA-RCGA-FFNN	1.80E-03	4.20E-02	17.18	5.05	3.65E-02	3.60E-03
DAX	DPSO-PSO-FFNN	1.70E-03	4.14E-02	16.53	4.70	3.60E-02	3.40E-03
	ANFIS	6.50E-03	8.08E-02	18.41	5.77	7.23E-02	7.85E-01
	ENN	2.60E-03	5.13E-02	27.44	8.47	8.81E-02	1.42E-02
DAX	FFNN	2.61E-02	1.61E-01	21.56	6.27	3.35E-02	2.61E-01
	BCGA-RCGA-FFNN	1.00E-03	3.18E-02	12.89	3.64	2.88E-02	2.00E-03
	DPSO-PSO-FFNN	9.43E-04	3.07E-02	12.04	3.49	2.78E-02	1.70E-03
DAX	ANFIS	1.42E-02	1.19E-01	23.93	7.39	1.05E-01	3.85E-00
	ENN	4.60E-01	6.78E-01	83.61	37.02	7.48E-01	1.51E-00
	FFNN	1.22E-02	1.10E-01	27.52	7.41	1.00E-01	2.55E-02
DAX	BCGA-RCGA-FFNN	3.10E-03	5.60E-02	18.73	5.61	5.00E-02	5.60E-03
	DPSO-PSO-FFNN	2.80E-03	5.33E-02	15.94	4.89	4.79E-02	4.90E-03

Table 8 (continued)

Stock Index	Model	MSE	RMSE	MAAPE (%)	SMAPE (%)	Theil's U	ARV
SSE Composite Index	ANFIS	1.32E-02	1.14E-01	19.54	5.80	1.03E-01	2.22E-00
	ENN	1.98E-01	4.45E-01	63.22	18.59	3.59E-01	5.62E-01
	FFNN	2.25E-02	1.49E-01	29.67	8.42	1.41E-01	3.47E-01
	BCGA-RCGA-FFNN	4.60E-03	6.76E-02	13.96	3.88	5.58E-02	3.59E-02
	DPSO-PSO-FFNN	9.08E-04	3.01E-02	8.09	2.37	2.73E-02	6.70E-03

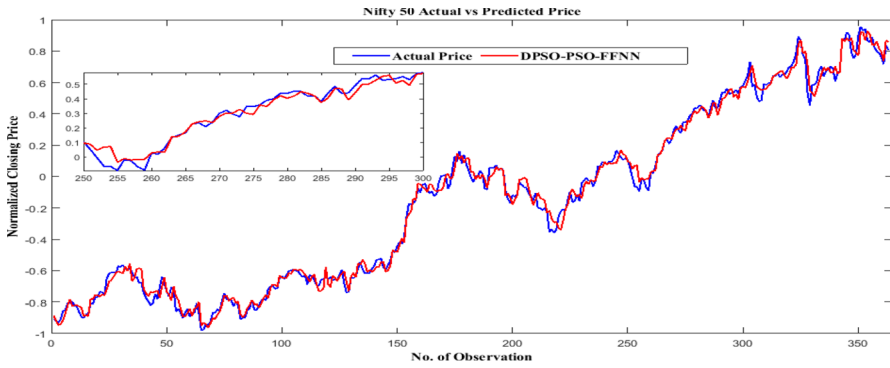


Fig. 12 Forecasting of 5-days ahead close price of Nifty 50

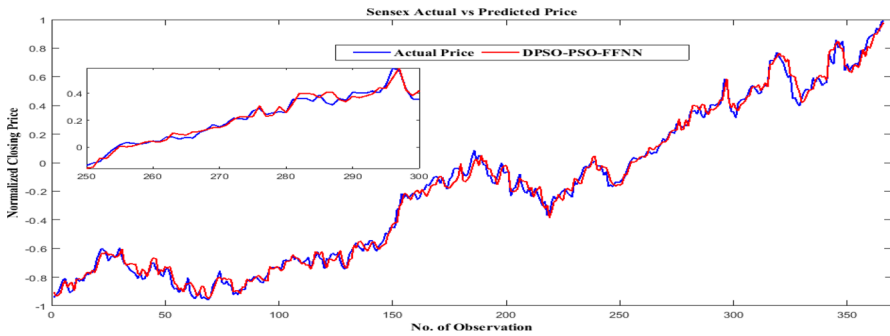


Fig. 13 Forecasting of 5-days ahead close price of Sensex

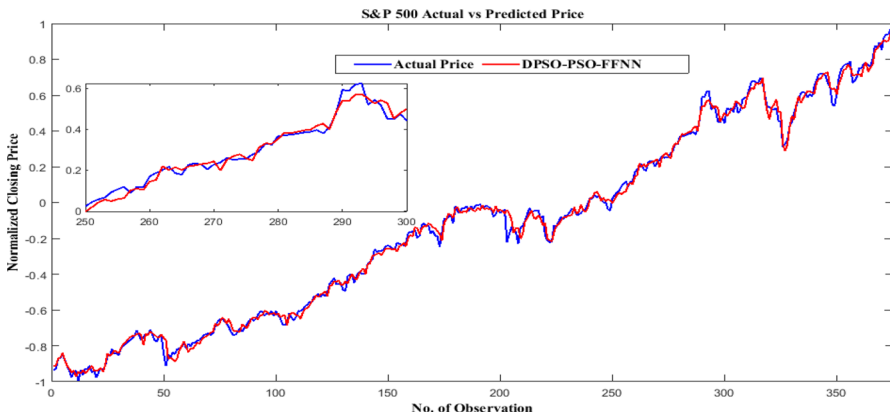


Fig. 14 Forecasting of 5-days ahead close price of S&P 500

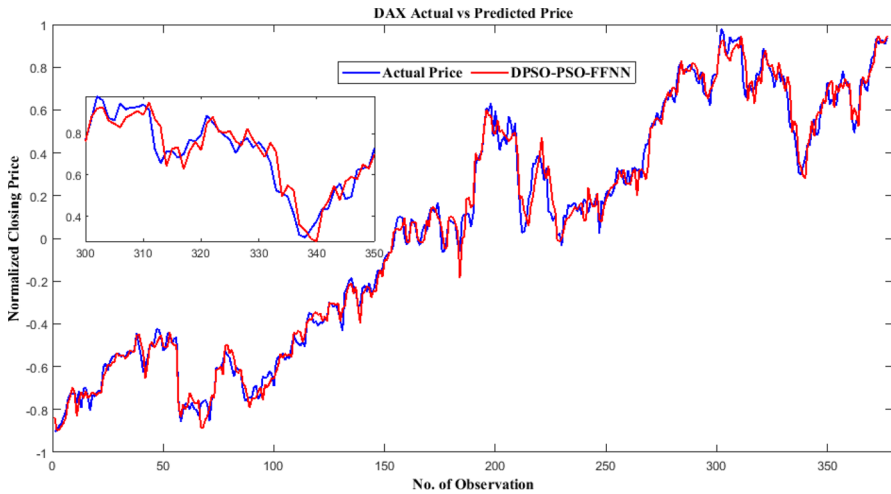


Fig. 15 Forecasting of 5-days ahead close price of DAX

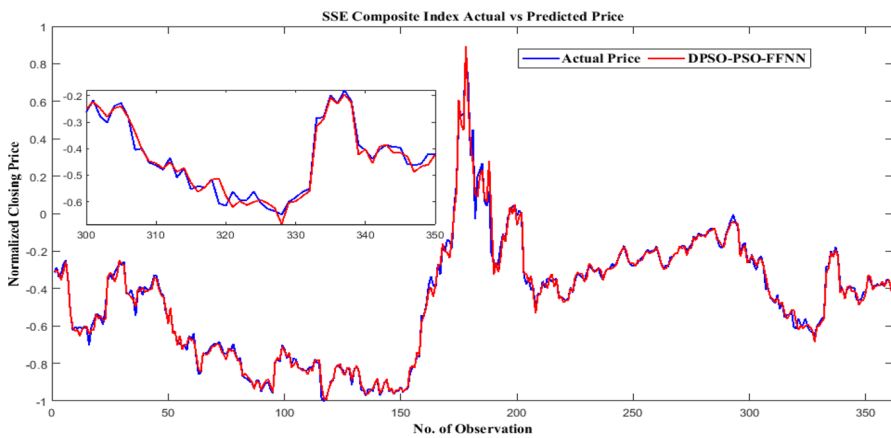


Fig. 16 Forecasting of 5-days ahead close price of SSE Composite Index

to 1-day ahead and 5-days ahead forecasting, experimental results show that proposed model performs better than other models for 10-days ahead forecasting. However, the forecasting performance of the proposed model for 1-day ahead and 5-days ahead is superior as compared to the 10-days ahead forecasting for the five stock indices.

Table 9 10-days ahead forecasting performance of the proposed model and its comparison with other models

Stock Index	Model	MSE	RMSE	MAAPE (%)	SMAPE (%)	Theil's U	ARV
Nifty 50	ANFIS	2.55E-02	1.59E-01	31.82	10.12	1.29E-01	2.66E-0
	ENN	4.40E-03	6.66E-02	26.92	8.38	6.64E-02	1.01E-02
	FFNN	3.24E-02	1.80E-01	20.58	6.89	1.25E-01	9.19E-01
	BCGA-RCGA-FFNN	3.30E-03	5.71E-02	20.25	6.81	4.74E-02	6.70E-03
	DPSO-PSO-FFNN	2.90E-03	5.43E-02	19.94	5.57	4.51E-02	6.10E-03
	ANFIS	1.02E-02	1.01E-01	20.99	6.55	8.68E-02	1.42E-0
Sensex	ENN	3.90E-03	6.20E-02	23.25	7.34	8.47E-02	6.11E-02
	FFNN	2.16E-02	1.41E-01	18.28	4.91	9.79E-02	2.56E-01
	BCGA-RCGA-FFNN	2.80E-03	5.28E-02	15.50	4.76	4.57E-02	6.00E-03
	DPSO-PSO-FFNN	2.50E-03	4.99E-02	15.49	4.45	4.32E-02	4.90E-03
	ANFIS	7.20E-03	8.49E-02	22.35	7.09	8.01E-02	2.14E-0
	ENN	2.70E-03	5.22E-02	24.70	7.78	7.59E-02	2.29E-02
S&P500	FFNN	3.35E-02	1.82E-01	24.91	7.49	1.58E-01	4.24E-01
	BCGA-RCGA-FFNN	1.60E-03	3.95E-02	15.07	4.78	3.76E-02	3.50E-03
	DPSO-PSO-FFNN	1.40E-03	3.73E-02	14.96	4.68	3.54E-02	3.30E-03
	ANFIS	1.74E-02	1.32E-01	30.48	9.13	1.16E-01	2.48E-00
	ENN	4.34E-02	2.08E-01	42.97	14.07	1.80E-01	7.51E-01
	FFNN	1.43E-02	1.19E-01	33.56	11.73	1.08E-01	2.58E-02
DAX	BCGA-RCGA-FFNN	6.10E-03	7.81E-02	24.43	7.57	7.08E-02	1.06E-02
	DPSO-PSO-FFNN	5.70E-03	7.54E-02	24.00	7.10	6.86E-02	1.02E-06

Table 9 (continued)

Stock Index	Model	MSE	RMSE	MAAPE (%)	SMAPE (%)	Theil's U	ARV
SSE Composite Index	ANFIS	1.07E-02	1.03E-01	19.91	5.95	9.32E-02	2.81E-00
	ENN	1.20E-01	3.47E-01	58.45	20.08	2.95E-01	4.02E-01
	FFNN	2.16E-01	4.65E-01	66.09	19.61	4.19E-01	5.10E-00
	BCGA-RCGA-FFNN	3.38E-02	1.84E-01	38.18	10.40	1.73E-01	1.47E-00
	DPSO-PSO-FFNN	6.00E-03	7.78E-02	18.88	5.23	7.06E-02	3.68E-02

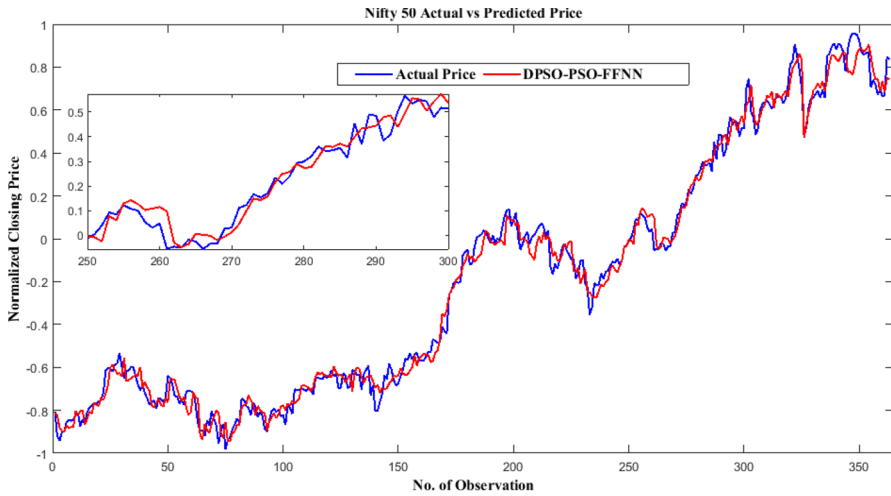


Fig. 17 Forecasting of 10-days ahead close price of Nifty 50

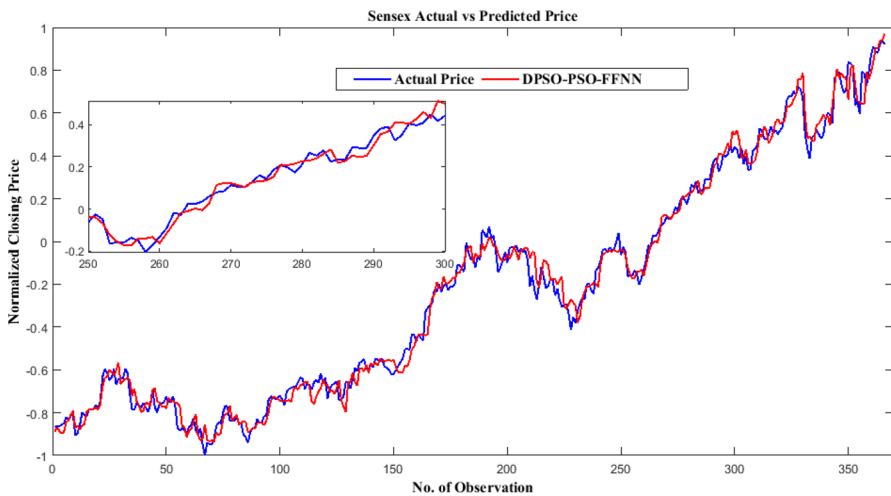


Fig. 18 Forecasting of 10-days ahead close price of Sensex

6 Conclusion

This study proposed a two-stage swarm intelligence based hybrid system by combining the DPSO technique that searches automatically for optimum feature subset of technical indicators and optimum number of hidden neurons and PSO for tuning the initial weight and bias of feed-forward neural network (FFNN) and LMBP algorithm for training the FFNN. The forecasting ability of proposed model has been verified by forecasting the daily close price of five major stock

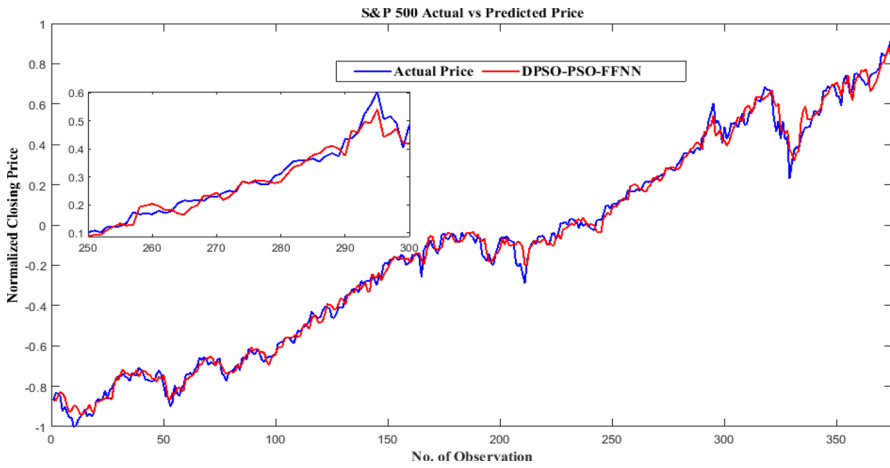


Fig. 19 Forecasting of 10-days ahead close price of S&P 50

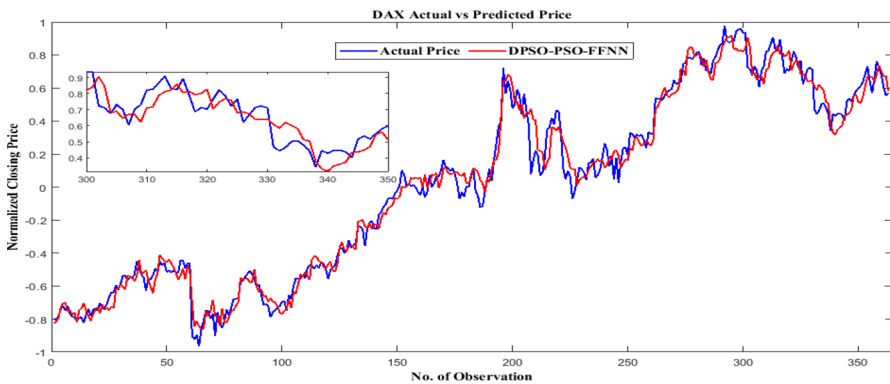


Fig. 20 Forecasting of 10-days ahead close price of DAX

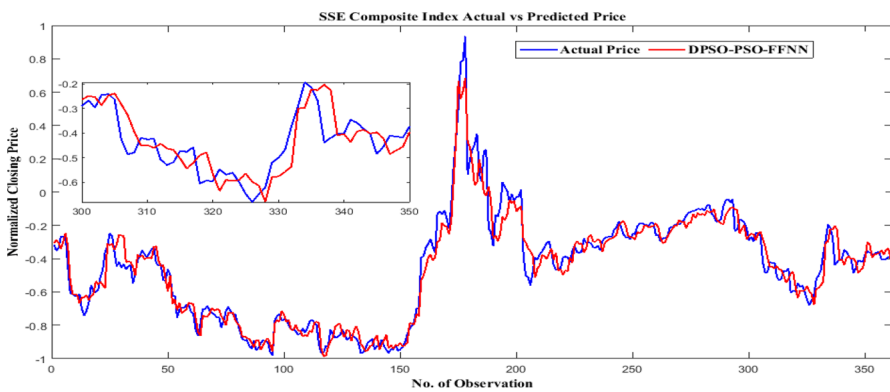


Fig. 21 Forecasting of 10-days ahead close price of SSE Composite Index

indices namely Nifty 50, Sensex, S&P 500, DAX and SSE Composite Index for multiple-horizon (1-day ahead, 5-days ahead and 10-days ahead) forecasting. In this study, we have developed another evolutionary computational based hybrid model (BCGA-RCGA-FFNN) by combining BCGA and RCGA with FFNN. The main conclusion drawn from this study can be summarized as follows:

1. The proposed swarm intelligence based hybrid model effectively obtains the optimal subset of technical indicators for five stock price series datasets. The proposed model reduces the feature set and enhances the accuracy of the model and minimizes the complexity of regular FFNN. Hence, the chosen technical indicators are efficient for modeling stock price time series forecasting problem.
2. The automatic network architecture search approach has effectively incorporated the global search capability of DPSO/PSO with FFNN to obtain the joint optimization of the feature set, the number of hidden neurons, and initial parameters of FFNN in two stages simultaneously. The local search capability of the LMBP algorithm has been improved, thereby reducing the risk of being stuck in local minima and avoiding the problem of underfitting and overfitting.
3. Experimental results obtained by applying the proposed model to five stock indices for multiple-horizon (1-day ahead, 5-days ahead and 10-days ahead) indicate the superior forecasting performance of the proposed model in comparison with ANFIS, ENN, regular FFNN and BCGA-RCGA-FFNN. However, the proposed model performs better in 1-day ahead forecasting as compared to 5-days and 10-days ahead forecasting. Hence, DPSO-PSO-FFNN is a promising approach for searching the features and network architecture in the stock price time series forecasting domain for multiple-horizon forecasting.

In this paper, we considered only technical indicators; future research will be focused on long-term forecasting using fundamental indicators and macro as well as micro-economic data that impact the stock markets. The proposed model can also be used to deal with other financial time series problems such as inflation forecasting, exchange rate forecasting, volatility prediction etc. The work can be expanded by considering the other parameters of ANN such as learning rate, activation function etc. In future, other nature inspired and evolutionary algorithms will also be considered to improve the parameters of the model.

Author's contribution GK: Conceptualization, writing-original draft preparation, software, visualization. UPS: Methodology, result validation, writing- reviewing and editing. SJ: Writing- reviewing and supervision.

Funding No funding received from any organization.

Declaration

Conflict of interest The authors declare that they have no conflict of interest.

References

- Adebiyi, A. A., Adewumi, A. O., & Ayo, C. K. (2014). Comparison of ARIMA and artificial neural networks models for stock price prediction. *Journal of Applied Mathematics*, 2014, 1–7.
- Ariyo, A. A., Adewumi, A. O., & Ayo, C. K. (2014). Stock price prediction using the ARIMA model. In *UKSim-AMSS 16th international conference on computer modelling and simulation* (pp. 106–112). IEEE.
- Atsalakis, G. S., & Valavanis, K. P. (2009). Surveying stock market forecasting techniques—part II: Soft computing methods. *Expert Systems with Applications*, 36(3), 5932–5941.
- Azadeh, A., Saberi, M., Ghaderi, S., Gitiforouz, A., & Ebrahimipour, V. (2008). Improved estimation of electricity demand function by integration of fuzzy system and data mining approach. *Energy Conversion and Management*, 49(8), 2165–2177.
- Baldominos, A., Saez, Y., & Isasi, P. (2020). On the automated, evolutionary design of neural networks: Past, present, and future. In *Neural computing and applications* (pp. 1–27).
- Bartlett, P., & Downs, T. (1990). *Training a neural network with a genetic algorithm*. University of Queensland.
- Basheer, I. A., & Hajmee, M. (2000). Artificial neural networks: Fundamentals, computing, design, and application. *Journal of Microbiological Methods*, 43(1), 3–31.
- Bisoi, R., & Dash, P. K. (2014). A hybrid evolutionary dynamic neural network for stock market trend analysis and prediction using unscented Kalman filter. *Applied Soft Computing*, 19, 41–56.
- Botchkarev, A. (2018). Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology (pp. 1–37). arXiv preprint. <https://arxiv.org/abs/1809.03006>.
- Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time series analysis: Forecasting and control* (5th ed.). Wiley.
- Chen, W., Jiang, M., & Jiang, C. (2019). Constructing a multilayer network for stock market. *Soft Computing* 1–17.
- Chiang, W. C., Enke, D., Wu, T., & Wang, R. (2016). An adaptive stock index trading decision support system. *Expert Systems with Applications*, 59, 195–207.
- Darwish, A., Hassanien, A. E., & Das, S. (2020). A survey of swarm and evolutionary computing approaches for deep learning. *Artificial Intelligence Review*, 53(3), 1767–1812.
- Elsken, T., Metzen, J. H., & Hutter, F. (2018). Efficient multi-objective neural architecture search via Lamarckian evolution. arXiv preprint. <https://arxiv.org/abs/1804.09081>.
- Elsken, T., Metzen, J. H., Hutter, F. (2018). Neural architecture search: A survey. arXiv preprint. <https://arxiv.org/abs/1808.05377>.
- Elsken, T., Metzen, J. H., & Hutter, F. (2018). Neural architecture search: A survey. arXiv preprint. <https://arxiv.org/abs/1808.05377>.
- Fama, E. (1970). *Efficient capital markets: A review of theory and empirical work*. R. Lowbridge (Module Leader).
- Gao, S., Zhou, M., Wang, Y., Cheng, J., Yachi, H., & Wang, J. (2018). Dendritic neuron model with effective learning algorithms for classification, approximation, and prediction. *IEEE Transactions on Neural Networks and Learning Systems*, 30(2), 601–614.
- Gocken, M., Ozcalici, M., Boru, A., & Dostogru, A. T. (2016). Integrating metaheuristics and artificial neural networks for improved stock price prediction. *Expert Systems with Applications*, 44, 320–331.
- Gocken, M., Ozcalici, M., Boru, A., & Dostogru, A. T. (2019). Stock price prediction using hybrid soft computing models incorporating parameter tuning and input variable selection. *Neural Computing and Applications*, 31(2), 577–592.
- Goh, A. T. (1995). Back-propagation neural networks for modeling complex systems. *Artificial Intelligence in Engineering*, 9(3), 143–151.
- Gong, Z., Chen, H., Yuan, B., & Yao, X. (2018). Multiobjective learning in the model space for time series classification. *IEEE Transactions on Cybernetics*, 49(3), 918–932.
- Guresen, E., Gulgun, K., & Daim, T. U. (2011a). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, 38(8), 10389–10397.
- Guresen, E., Kayakutlu, G., & Daim, T. U. (2011b). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, 38(8), 10389–10397.

- Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3, 1157–1182.
- Hagan, M. T., & Menhaj, M. B. (1994). Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks*, 5(6), 989–993.
- Haleh, H., Moghaddam, B. A., & Ebrahimijam, S. (2011). A new approach to forecasting stock price with EKF data fusion. *International Journal of Trade, Economics and Finance*, 2(2), 109–114.
- Han, H. G., Lu, W., Hou, Y., & Qiao, J. F. (2016). An adaptive-PSO-based self-organizing RBF neural network. *IEEE Transactions on Neural Networks and Learning Systems*, 29(1), 104–117.
- Haupt, R. L., & Haupt, S. E. (2004). *Practical genetic algorithms* (pp. 51–56). Wiley.
- Haykin, S. (1999). *Neural network: A comprehensive foundation* (2nd ed.). Prentice Hall.
- Holland, J. H. (1975). *Adaptation in natural and artificial systems*. University of Michigan Press.
- Hu, H., Ao, Y., Bai, Y., Cheng, R., & Xu, T. (2020). An improved Harris's Hawks optimization for SAR target recognition and stock market index prediction. *IEEE Access*, 8, 65891–65910.
- Hu, H., Tang, L., Zhang, S., & Wang, H. (2018). Predicting the direction of stock markets using optimized neural networks with Google Trends. *Neurocomputing*, 285, 188–195.
- Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4), 679–688.
- Ibrahim, D. (2016). An overview of soft computing. *Procedia Computer Science*, 102, 34–38.
- Inthachot, M., Boonjing, V., & Intakosum, S. (2016). Artificial neural network and genetic algorithm hybrid intelligence for predicting Thai stock price index trend. *Computational Intelligence and Neuroscience*, 2016, 1–8.
- Jang, J. S. (1993). ANFIS: Adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(3), 665–685.
- Kapanova, K. G., Dimov, I., & Sellier, J. M. (2018). A genetic approach to automatic neural network architecture optimization. *Neural Computing and Applications*, 29(5), 1481–1492.
- Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks* (Vol. 4, pp. 1942–1948). IEEE.
- Kennedy, J., & Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. In *IEEE international conference on systems, man, and cybernetics. Computational cybernetics and simulation* (Vol. 5, pp. 4104–4108).
- Kim, K. J. (2006). Artificial neural networks with evolutionary instance selection for financial forecasting. *Expert Systems with Applications*, 30(3), 519–526.
- Kim, S., & Kim, H. (2016). A new metric of absolute percentage error for intermittent demand forecasts. *International Journal of Forecasting*, 32(3), 669–679.
- Kohonen, T. (1988). An introduction to neural computing. *Neural Networks*, 1(1), 3–16.
- Kristjanpoller, W., & Minutolo, M. C. (2018). A hybrid volatility forecasting framework integrating GARCH, artificial neural network, technical analysis and principal components analysis. *Expert Systems with Applications*, 109, 1–11.
- Kumar, G., Jain, S., & Singh, U. P. (2020). Stock market forecasting using computational intelligence: A survey. *Archives of Computational Methods in Engineering* 1–33.
- Kumar, D., Meghwani, S. S., & Thakur, M. (2016). Proximal support vector machine based hybrid prediction models for trend forecasting in financial markets. *Journal of Computational Science*, 17, 1–13.
- Kumar, S. (2004). *Neural networks: A classical approach* (2nd ed., pp. 61–65). Tata McGraw-Hill Education.
- Lin, C. S., Chiu, S. H., & Lin, T. Y. (2012). Empirical mode decomposition-based least squares support vector regression for foreign exchange rate forecasting. *Economic Modelling*, 29(6), 2583–2590.
- Liu, C., Barret, Z., Maxim, N., Jonathon, S., Wei, H., Li-Jia, L., Li, F. F., Alan, Y., Jonathan, H., & Kevin, M. (2018). Progressive neural architecture search. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 19–34).
- Liu, F., & Wang, J. (2012). Fluctuation predictions of stock market index by Legendre neural network with random time strength function. *Neurocomputing*, 83, 12–21.
- Lu, C. J., Lee, T. S., & Chiu, C. C. (2009). Financial time series forecasting using independent component analysis and support vector regression. *Decision Support Systems*, 47(2), 115–125.
- Mandziuk, J., & Jaruszewicz, M. (2011). Neuro-genetic system for stock index prediction. *Journal of Intelligent and Fuzzy Systems*, 22(2), 93–123.
- Menkhoff, L. (1997). Examining the use of technical currency analysis. *International Journal of Finance and Economics*, 2(4), 307–318.

- Mirjalili, S., & Lewis, A. (2013). S-shaped versus V-shaped transfer functions for binary particle swarm optimization. *Swarm and Evolutionary Computation*, 9, 1–14.
- More, J. J. (1978). The Levenberg-Marquardt algorithm: Implementation and theory. In *Numerical analysis* (pp. 105–116). Springer.
- Nassirtoossi, A. K., Wah, T. Y., & Ling, D. N. C. (2011). A novel FOREX prediction methodology based on fundamental data. *African Journal of Business Management*, 5(20), 8322–8330.
- Nayak, S. C., Kumar, K. V., & Jilla, K. (2020). ACRRFLN: Artificial chemical reaction of recurrent functional link networks for improved stock market prediction. In *Computational intelligence in data mining* (pp. 311–325). Springer.
- Nayak, S. C., Misra, B. B., & Behera, H. S. (2019). ACFLN: Artificial chemical functional link network for prediction of stock market index. *Evolving Systems*, 10(4), 567–592.
- Pradeepkumar, D., & Ravi, V. (2017). Forecasting financial time series volatility using particle swarm optimization trained quantile regression neural network. *Applied Soft Computing*, 58, 35–52.
- Qiu, M. Y., Song, Y., & Akagi, F. (2016). Application of artificial neural network for the prediction of stock market returns: The case of the Japanese stock market. *Chaos, Solitons and Fractals*, 85, 1–7.
- Ren, G., Cao, Y., Wen, S., Huang, T., & Zeng, Z. (2018). A modified Elman neural network with a new learning rate scheme. *Neurocomputing*, 286, 11–18.
- Reza, H., Shahrabi, J., & Hadavandi, E. (2015). A bat-neural network multi-agent system (BNNMAS) for stock price prediction: Case study of DAX stock price. *Applied Soft Computing*, 29, 196–210.
- Rout, M., & Koudjonou, K. M. (2020). An evolutionary algorithm based hybrid parallel framework for Asia foreign exchange rate prediction. In *Nature inspired computing for data science* (pp. 279–295). Springer.
- Sahoo, S., & Mohanty, M. N. (2020). Stock market price prediction employing artificial neural network optimized by Gray Wolf optimization. In *New paradigm in decision science and management* (pp. 77–87). Springer.
- Sakshi, S., & Kumar, R. (2019). A neuro-genetic technique for pruning and optimization of ANN weights. *Applied Artificial Intelligence*, 33(1), 1–26.
- Sands, T. M., Tayal, D., Morris, M. E., & Monteiro, S. T. (2015). Robust stock value prediction using support vector machines with particle swarm optimization. In *IEEE congress on evolutionary computation (CEC)* (pp. 3327–3331). IEEE.
- Senapati, M. R., Das, S., & Mishra, S. (2018). A novel model for stock price prediction using hybrid neural network. *Journal of the Institution of Engineers (india): Series B*, 99(6), 555–563.
- Shynkevich, Y., McGinnity, T. M., Coleman, S. A., Belatreche, A., & Li, Y. (2017). Forecasting price movement using technical indicators: Investing the impact of varying window length. *Neurocomputing*, 264, 71–88.
- Si, Y. W., & Yin, J. (2013). OBST-based segmentation approach to financial time series. *Engineering Applications of Artificial Intelligence*, 26(10), 2581–2596.
- Technical analysis. [Online]. Retrieved from <https://www.investopedia.com/technical-analysis-4689657>. Accessed March 20, 2020.
- Technical analysis library in python. [Online]. Retrieved from <https://technical-analysis-library-inpython.readthedocs.io/en/latest/ta.html>. Accessed May 30, 2020.
- Technical indicators. [Online]. Retrieved from https://school.stockcharts.com/doku.php?id=technical_indicators. Accessed March 30, 2020.
- Theil, H. (1966). *Applied economic forecasting*. North-Holland Pub. Co.
- Ulke, V., Sahin, A., & Subasi, A. (2018). A comparison of time series and machine learning models for inflation forecasting: Empirical evidence from the USA. *Neural Computing and Applications*, 30(5), 1519–1527.
- Vaisla, K. S., & Bhatt, A. K. (2010). An analysis of the performance of artificial neural network technique for stock market forecasting. *International Journal on Computer Science and Engineering*, 2(6), 2104–2109.
- Wang, J., & Kumbasar, T. (2019). Parameter optimization of interval Type-2 fuzzy neural networks based on PSO and BBBC methods. *IEEE/CAA Journal of Automatica Sinica*, 6(1), 247–257.
- Wang, J., & Wang, J. (2015). Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks. *Neurocomputing*, 156, 68–78.
- Wang, J. J., Wang, J. Z., Zhang, Z. G., & Guo, S. P. (2012). Stock index forecasting based on a hybrid model. *Omega*, 40(6), 758–766.
- Webb, A. R. (2003). *Statistical pattern recognition* (pp. 307–352). Wiley.

- Wright, A. H. (1991) Genetic algorithms for real parameter optimization. In *Foundations of genetic algorithms* (Vol. 1, pp. 205–218).
- Xiong, T., Bao, Y., Hu, Z., & Chiong, R. (2015). Forecasting interval time series using a fully complex-valued RBF neural network with DPSO and PSO algorithms. *Information Sciences*, 305, 77–92.
- Yan, W. (2012). Toward automatic time-series forecasting using neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 23(7), 1028–1039.
- Yang, F., Chen, Z., Li, J., & Tang, L. (2019). A novel hybrid stock selection method with stock prediction. *Applied Soft Computing*, 80, 820–831.
- Yao, J., Tan, C. L., & Poh, H. L. (1999). Neural networks for technical analysis: A study on KLCI. *International Journal of Theoretical and Applied Finance*, 2(2), 221–241.
- Yao, X. (1999). Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9), 1423–1447.
- Yeh, W. C. (2009). A two-stage discrete particle swarm optimization for the problem of multiple multi-level redundancy allocation in series systems. *Expert System Application*, 36, 9192–9200.
- Yu, L., & Liu, H. (2003). Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of the 20th international conference on machine learning* (pp. 856–863).
- Zhang, D. Y., Song, H. W., & Chen, P. (2008). Stock market forecasting model based on a hybrid ARMA and support vector machines. In *International conference on management science and engineering 15th annual conference proceedings* (pp. 1312–1317). IEEE.
- Zhong, X., & Enke, D. (2017). Forecasting daily stock market return using dimensionality reduction. *Expert Systems with Applications*, 67, 126–139.
- Zhou, T., Gao, S., Wang, J., Chu, C., Todo, Y., & Tang, Z. (2016). Financial time series prediction using a dendritic neuron model. *Knowledge-Based Systems*, 105, 214–224.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Authors and Affiliations

Gourav Kumar¹ · Uday Pratap Singh² · Sanjeev Jain³

¹ School of Computer Science and Engineering, Shri Mata Vaishno Devi University, Katra, Jammu and Kashmir 182320, India

² School of Mathematics, Shri Mata Vaishno Devi University, Katra, Jammu and Kashmir 182320, India

³ Department of Computer Science and Engineering, Indian Institute of Information Technology, Design and Manufacturing, Jabalpur, MP 482005, India