Nayak and Misra *Financial Innovation* (2019) 5:38
https://doi.org/10.1186/s40854-019-0153-1

Financial Innovation

RESEARCH Open Access

A chemical-reaction-optimization-based neuro-fuzzy hybrid network for stock closing price prediction

Sarat Chandra Nayak[1*] and Bijan Bihari Misra[2]

* Correspondence: saratnayak234@gmail.com; saratchandranayak@kpritech.ac.in
[1]Department of Computer Science and Engineering, CMR College of Engineering & Technology, Hyderabad 501401, India
Full list of author information is available at the end of the article

Abstract

Accurate prediction of stock market behavior is a challenging issue for financial forecasting. Artificial neural networks, such as multilayer perceptron have been established as better approximation and classification models for this domain. This study proposes a chemical reaction optimization (CRO) based neuro-fuzzy network model for prediction of stock indices. The input vectors to the model are fuzzified by applying a Gaussian membership function, and each input is associated with a degree of membership to different classes. A multilayer perceptron with one hidden layer is used as the base model and CRO is used to the optimal weights and biases of this model. CRO was chosen because it requires fewer control parameters and has a faster convergence rate. Five statistical parameters are used to evaluate the performance of the model, and the model is validated by forecasting the daily closing indices for five major stock markets. The performance of the proposed model is compared with four state-of-art models that are trained similarly and was found to be superior. We conducted the Deibold-Mariano test to check the statistical significance of the proposed model, and it was found to be significant. This model can be used as a promising tool for financial forecasting.

Keywords: Artificial neural network, Neuro-fuzzy network, Multilayer perceptron, Chemical reaction optimization, Stock market forecasting, Financial time series forecasting

Introduction

Forecasting stock market behavior is quite uncertain due to high market volatility, nonlinearity, their complex dynamic systems, and the time-varying nature of markets. In addition, markets will respond arbitrarily to changes in the current political climate and other macroeconomic factors (Hsu et al., 2016; Kotha & Sahu, 2016). These characteristics of stock market must be captured and accounted for in models in order to establish intelligent techniques for forecasting market prices. Achieving the best forecasting accuracy with the lowest volume of input data and least complex model architecture is the key objective of market analysts, forecasters, and researchers (Nayak et al., 2018). Stock market researchers focus on developing models/methodologies to effectively forecast prices, with the goal of maximizing profits through appropriate trading strategies. However, in reality, this is a critical, demanding, and challenging job.

Early approaches to realistically solving this problem by observing the hidden laws of real stock index data were based on verities in statistical and computational models (Zhang, 2003; Adhikari & Agrawal, 2014). In recent years, there has been tremendous development in the field of artificial intelligence and soft computing methodologies, including artificial neural networks (ANN), evolutionary algorithms, and fuzzy systems. Newly developed data mining techniques and advancements in computational intelligence capabilities have been applied to build intelligent information systems for modeling complex, dynamic, and multivariate nonlinear systems (Nayak et al., 2018; Adhikari & Agrawal, 2014). In particular, soft computing methodologies have been applied successfully to areas such as data classification (Alatas, 2011; Alatas, 2012; Nayak et al., 2015), financial forecasting (Nayak et al., 2018), credit scoring (Addo et al., 2018; Tomczak & Zięba, 2015; Chow, 2018), portfolio management (Xu et al., 2011a), business failure prediction, and risk level evaluation (Daubie & Meskens, 2002; Chandra et al., 2009), and they have been found to produce significantly improved results. ANNs have proven to be an effective modeling procedure in stock market forecasting when the input-output mapping contains both regularities and exceptions (Nayak et al., 2018; Zhang, 2003; Adhikari & Agrawal, 2014; Gu et al., 2018; Board, 2017). ANNs also allow for adaptive fine-tuning of the model, its parameters, and the nonlinear description of the problem. Stock markets, characterized by chaos and uncertainty, also behave with regularities and exceptions in terms of data. The advantages of ANNs have made them the center of attention for researchers developing neural-network-based forecasting models for stock market prediction.

Multilayer perceptrons (MLP) are the most reliable and frequently used models among the ANN methodologies. Nonlinear processing of elements and massive interconnectivity are the most important characteristics of an MLP. Some successful applications of MLPs include financial time series forecasting (Yu et al., 2009), market trend analysis (Turchenko et al., 2011), macroeconomic data forecasting (Aminian et al., 2006), and stock exchange movement prediction (Mostafa, 2010). These studies establish the generalization capability and enhanced prediction accuracy of MLP-based forecasting methods. Other applications of MLPs are also found in railway traffic (Zhuo et al., 2007), airline passenger traffic (Blinova, 2007), and maritime traffic forecasting (Mostafa, 2004). From the literature, it can also be observed that MLPs have been successfully applied to forecasting short-term electric load consumption demand (Darbellay & Slama, 2000) and air pollution (Rahman et al., 2017). MLPs are generally trained with the backpropagation of error learning algorithms. However, suffering from slow convergence, sticking to local minima, and computational complexities are the well-known drawbacks of backpropagation learning based MLPs (Calderon & Cheh, 2002). At present, there is no formal method of deriving an optimized MLP network for a given classification or prediction task (Ecer, 2013). To overcome the local minima, more nodes can be added consecutively to the hidden layers. Multiple hidden layers and more neurons in each layer may increase the computational intricacy of the model. Hence, there is no direct method for deciding on the optimal MLP structure for solving a particular problem. The refining process may suffer from elongated computational time, accomplished through iterative testing of a range of architectural parameters and

adopting the most successful architecture. Defining the optimal architecture and parameters for an MLP is a matter of experimentation, which is computationally expensive. In order to circumvent the limitations of gradient-descent-based learning, several nature-inspired learning techniques have been developed and applied successfully in the literature. However, their performance essentially depends upon fine tuning several algorithm-specific control parameters. Choosing appropriate control parameters is a difficult task and requires intensive human intervention. Inappropriate selection of algorithm parameters may add to the computational burden or land the model in local optima. Therefore, adopting an optimization technique that requires very few controlling parameters without losing performance can be better choice for solving real-world problems. From a review of the existing research on stock market forecasting, it is observed that 1) achieving superior forecasting accuracy by adapting less complex models is an important area of present-day research, and 2) with the aim of better forecasting accuracies, researchers are moving toward adopting hybrid neural-network models with a large number of evolutionary search optimization algorithms.

Fuzzy Logic (FL) systems are effective in input-output and in-process parameter relationship modeling. The concept of fuzzy logic was introduced by Lotfi Zadeh (Zadeh, 1965) as a way of processing data by allowing partial set membership rather than crisp set membership. The essence of fuzzy set theory lies in its ability to handle vague, ambiguous, and imprecise information, unlike classical set and probabilistic theories, which can handle only dichotomous or Boolean information. Both ANNs and FL have been extensively used by researchers in modeling to describe human thinking and reasoning in a mathematical framework (Liu, 2004; Dubois & Prade, 1980). The integration of ANNs and FL provides a more synergetic effect than either one used individually. The advantage of ANN learning and fuzzy "if-then" rules with suitable membership functions are hybridized to obtain a high degree of accuracy in generating nonlinear input-output relationships. These hybrid systems combine the learning and connectedness architecture of neural networks with the human-like logical reasoning capability of fuzzy systems, thereby taking advantage of both aspects (Zadeh, 1965; Liu, 2004; Dubois & Prade, 1980; Alalaya et al., 2018; Romahi & Shen, 2000; Abraham et al., 2001; Nayak et al., 2012; Nayak et al., 2016; Guan et al., 2018; Ghosh et al., 2009; Kuo et al., 2001; Esfahanipour & Aghamiri, 2010; Singh et al., 2016; Castellano et al., 2007; Keles_ & Keles_, 2013; Boyacioglu & Avci, 2010; Quek, 2005; Abbasi & Abouec, 2008; Yunos et al., 2008; Atsalakis & Valavanis, 2009; Fouladvand et al., 2015). The applications of neuro-fuzzy integrations in data mining problems are discussed in Section 2.

The group method of data handling (GMDH) is another approach aimed at identifying the functional structure of a model hidden within the empirical data. It uses feed-forward networks based on short-term polynomial transfer functions, whose coefficients are obtained using regression, combined with emulation of the self-organizing activity behind neural network learning. Previous research shows that it is the optimal model, and it has a simpler structure than traditional neural models with a higher accuracy with inaccurate, small, or noisy datasets. A GMDH-type neural network based on a genetic algorithm (GA) has been applied to predict the

stock price index of the petrochemical industry in Iran (Shaverdi et al., 2012). The results obtained are found to be excellent and highly effective in stock price prediction. Neuro-fuzzy GMDH networks using evolutionary algorithms were proposed in (Najafzadeh et al., 2018; Najafzadeh & Bonakdari, 2016), and the models were found to produce very precise predictions. GMDH networks developed using gravitational search algorithms (GSA), as well as particle swarm optimization (PSO) and back propagation (BP) algorithms have been utilized to predict scour at abutments in armored beds, and they have produced accurate prediction results (Najafzadeh et al., 2015). A GMDH gene expression programming (GMDH-GEP) model performed well in predicting free span expansion rates below pipelines under waves (Najafzadeh & Saberi-Movahed, 2018).

Chemical reaction optimization (CRO) is an evolutionary optimization technique inspired by the nature of chemical reactions (Alatas, 2011; Alatas, 2012). This optimization method does not necessitate a local search to improve it, and it includes both local and global search capabilities (Alatas, 2011; Alatas, 2012). Unlike other optimization techniques, CRO does not entail many parameters that must be specified at beginning. It is only necessary to define the number of initial reactants prior to implementation (Alatas, 2011; Alatas, 2012). As the initial reactants are scattered over a feasible global search expanse, optimal solutions can be obtained with little iteration; hence, a significant reduction in computational time is achieved. CRO has been applied to solve many problems successfully, outperforming many existing evolutionary algorithms. There have been some applications of CRO to data mining, classification rule discovery and other domains (Lam et al., 2012; Lam et al., 2010; Pan et al., 2011; Xu et al., 2010; Xu et al., 2011b; James et al., 2011; Lam & Li, 2010; Truong et al., 2013) as well as in financial forecasting (Nayak et al., 2013; Nayak et al., 2017a; Nayak et al., 2017b), which is discussed in Section 2.

The objective of this study is to develop a hybrid model that can effectively forecast stock index prices with more precise accuracy. The hybrid model uses an MLP with one hidden layer as its base architecture. The input vectors are fuzzified by applying a Gaussian membership function, and each input is associated with a degree of membership to different classes, which in effect increases the dimensionality of the input vector. The Gaussian membership function provides a smoother transition between members and non-members in comparison to triangular and trapezoidal membership functions. Again, when compared with a bell membership function, it has fewer parameters, which makes it easier to use. The robust optimization capability of CRO, with its low number of tuning parameters, motivated us to adopt it as the learning technique. The CRO adjusts the weight and bias vector of the MLP model. The proposed CRO-based neuro-fuzzy network (CNFN) model is validated by forecasting the daily closing indices of the DJIA, BSE, FTSE, TAIEX, and NASDAQ stock markets. The performance of the proposed model is compared to other models, such as CRO based MLP (MLPCRO), back propagation based MLP (MLP-BP), adaptive neuro-fuzzy inference system (ANFIS), and radial basis functional neural network (RBFNN), that are trained similarly.

The foremost contributions of this article are as follows:

- Proposal of an integrated framework of ANNs and FL.
- Employment of CRO to adjust the weight and input vectors of the model.
- Analysis of state-of-art techniques for short- as well as long-term forecasting of real stock closing prices.
- Rigorous quantitative analysis using five of the latest techniques on data from five stock markets over a period of 13 years and 8 months.
- Statistical validation of the hypothesis indicating a significant difference between the proposed and comparative models.
- Adaptive model training to reduce the computational cost of the forecasting model.

The rest of the article is organized as follows. Related research is explored in Section 2. CRO is thoroughly discussed in Section 3. Section 4 presents the proposed CNFN approach. The experimental results are summarized in Section 5, and a clear analysis is carried out to establish the proposed model. Finally, Section 6 concludes the paper.

Related studies

Over the last few decades, rapid growth in the economies of developed countries has amplified the need for more competent and sophisticated forecasting models. In order to outperform conventional statistical methods, several artificial intelligence systems have been developed and found to be experimentally efficient (Nayak et al., 2017a; Nayak et al., 2017b; Zhong & Enke, 2017; Enke & Thawornwong, 2005). Over the decades, a number of forecasting models based on soft computing techniques, such as ANN (Enke & Thawornwong, 2005), FL and its hybridization (Alalaya et al., 2018; Romahi & Shen, 2000; Abraham et al., 2001), GA-based ANN (Nayak et al., 2012) have been applied to stock index forecasting. However, due to the presence of great noise and the complex dimensionality of stock price data, the results of these models may not be deemed satisfactory. Evaluation of clustering algorithms for financial risk analysis using multiple-criteria decision making has been carried out by researchers (Kou et al., 2014). The results demonstrate the effectiveness of the proposed method in evaluating clustering algorithms. Group decision making models for economic interpretations have also been presented by researchers in (Li et al., 2016; Zhang et al., 2019). A survey of existing studies and methodologies used to assess and measure systemic financial risk combined with machine learning technologies can be found in (Kou et al., 2019). A complete description of the applications of ANN in the literature, including methodologies, datasets, and functionalities, is given in Table 1.

On the contrary, several studies have led to a stronger foundation in the field of neuro-fuzzy systems (Alalaya et al., 2018; Romahi & Shen, 2000; Abraham et al., 2001; Nayak et al., 2012; Nayak et al., 2016; Guan et al., 2018; Ghosh et al., 2009; Kuo et al., 2001; Esfahanipour & Aghamiri, 2010; Singh et al., 2016; Castellano et al., 2007; Keles_ & Keles_, 2013; Boyacioglu & Avci, 2010; Quek, 2005; Abbasi & Abouec, 2008; Yunos et al., 2008; Atsalakis & Valavanis, 2009; Fouladvand et al., 2015). A GA-based fuzzy neural network was applied to measure qualitative effects on stock prices on the Taiwan stock market (Kuo et al., 2001). Studies have shown the importance of variable selection to the success of any network. A neuro-fuzzy

Table 1 Review of ANN articles on financial time series prediction

Citation	Methodology	Year	Application area/Dataset	Functionalities/Results
(Yu et al., 2009)	Neural network metamodeling	2009	Financial time series/S&P 500, Euro/USD	Better performance over RW, ARIMA, ES, BPNN
(Turchenko et al., 2011)	MLP with back propagation learning	2011	Short term prediction of stock prices of Fiat company,	91% of prediction result < 5% error and 33% results have < 15 error
(Aminian et al., 2006)	Artificial neural networks	2006	Forecasting economic data/ U.S. Real Gross Domestic Production and Industrial Production	Neural networks significantly outperform linear regression due to nonlinearities inherent in the data sets
(Mostafa, 2010)	MLP	2010	Kuwait Stock Exchange data 2001–2003	MLP performed better to generalized regression neural networks
(Zhuo et al., 2007)	Improved back propagation neural network	2007	Railway passenger traffic volume from 1980 to 1998	Better to standard back propagation neural network
(Blinova, 2007)	28-time-lagged feed-forward ANN	2007	Intraregional and interregional passenger traffic for 2006–2010 of Russian air transport network	ANN models developed adequately described the passenger traffic demand for the next two or 3 year
(Mostafa, 2004)	ARIMA and ANN	2004	Forecasting the Suez canal traffic flow	The models gave useful insight into the behavior of maritime traffic flows
(Darbellay & Slama, 2000)	ANN and ARIMA	2000	Forecasting the short-term evolution of the Czech electric load	ANN are superior
(Rahman et al., 2017)	Feed-forward NN and Elman NN with different learning methods	2017	Temporal and spatial atmospheric pollution index of Sterlitamak city	Quite effective prediction with 83% accuracy
(Calderon & Cheh, 2002)	Review and limitations of ANN based models	2002	Audit and risk assessment	Scope and limitations of ANN models are explored
(Ecer, 2013)	MLP, SVM, RBFNN	2013	Turkish banking failures/ 34 Turkish commercial banks, 17 of which failed the periods 1994–2001 and contains 36 ratios available for those types of banks	MLP performed better with 2.94% error.
(Zhong & Enke, 2017)	ANN with PCA, fuzzy robust PCA, kernel-based PCA	2017	Forecasting daily direction of S&P 500 Index ETF return based on 60 financial and economic features	ANN + PCA gives better result than other approaches
(Zhang, 2003)	ARIMA+ANN	2003	Bench mark time series datasets	The hybrid model had superior performance
(Enke & Thawornwong, 2005)	Multilayer feed-forward NN and Generalized regression NN, Probabilistic NN	2005	Monthly data (March 1976–December 1999), total 286 periods from S&P 500	Effectiveness of NN models were established for level estimation and classification. Trading strategies guided by NN models were able to generate higher profits.
(Ture & Kurt, 2006)	ANN	2006	Future expectations index, CDI interest tax rate, Selic interest tax rate/Formal employment, Brent oil price, Domestic market automobile sales, Consumer confidence index, Investors participation	Percentage of change in direction (POCID) is 93.62% for test set and 87.50% for validation set.
(Niaki & Hoseinzade, 2013)	ANN	2013	S&P 500 index 365 trading days	POCID of linear regression is 51.75. ANN had better POCID than LR

Table 1 Review of ANN articles on financial time series prediction *(Continued)*

Citation	Methodology	Year	Application area/Dataset	Functionalities/Results
(Alalaya et al., 2018)	ANN + Fuzzy	2018	Amman stock exchange of financial and banking sector from 1/2010 to 12/2016 with record data 265	Fuzzy-neural models are found prominent in terms of MSE, MAD

inference system adopted on a Takagi–Sugeno–Kang (TSK)-type fuzzy rule based system was developed for stock price prediction on the Tehran Stock Exchange Index (TEPIX) (Esfahanipour & Aghamiri, 2010). The results of this study had 97.8% accuracy. A comparative study of ANFIS and multiple linear regressions was carried out by authors in (Singh et al., 2016) for rainfall runoff modeling, and the ANFIS model was found to be successful.

Castellano et al. studied different hybrid soft-computing-based systems by combining ANNs with evolutionary algorithms and fuzzy systems (Castellano et al., 2007). This study reveals that the use of neuro-fuzzy and evolutionary fuzzy systems can have strong performance as well as learning and adaptive capabilities. A neuro-fuzzy classification tool called NEFCLASS was proposed by Keles and Keles (Keles_ & Keles_, 2013) to establish the adaptation of a fuzzy-rule-based system. An adaptive network-based fuzzy inference system for the prediction of stock market return was presented by Boyacioglu and Avci in (Boyacioglu & Avci, 2010) to forecast the Istanbul stock market (ISE). The experimental results revealed that the model successfully forecast the ISE national 100 index.

To forecast stock prices in the US stock exchange in 2005, Queck used ANFIS and a neuro-fuzzy network and found it to be successful (Quek, 2005). An adaptive neuro-fuzzy inference system was utilized to study the stock price trends in the Tehran stock exchange (Abbasi & Abouec, 2008). Their study was able to forecast the stock market with a low level of error. In order to predict the daily stock movements on the Kuala Lumpur Composite Index (KLCI), Yunos et al. developed a hybrid neuro-fuzzy model in 2008 (Yunos et al., 2008). Their experimental results revealed that the ANFIS algorithm is competent in forecasting when compared to any other ANN-based forecasting model. A neuro-fuzzy adaptive control system was proposed by Atsalakis and Valvanis (Atsalakis & Valavanis, 2009) to forecast the next day's stock price trend on the NYSE and ASE. The results showed superior performance to the buy and hold strategy. To speed up the iteration and improve the performance of the neuro-evolutionary algorithm, a modified neuro-evolutionary algorithm for mobile robot navigation was proposed (Fouladvand et al., 2015). The study showed that fuzzy logic is an easy way to add initial knowledge to the neuro-evolution algorithm to avoid learning from zero.

Chemical reaction optimization is one of the newly established meta-heuristics for optimization. Prospective readers may follow the complete guidelines for the implementation of CRO in this tutorial (Lam & Li, 2012). A real coded version of chemical reaction optimization (RCCRO) was proposed in (Lam et al., 2012) to solve continuous optimization problems. They also proposed an adaptive scheme

for RCCRO for performance improvement. The performance of RCCRO has been compared with a large number of techniques and experimented on using a set of standard continuous benchmark functions. The results show the suitability of CRO in solving problems in the continuous domain. CRO has also been successfully applied for population transition in peer-to-peer live streaming (Lam et al., 2010). CRO is employed to maximize the probability of universal streaming by manipulating the population transition probability matrix. This simulation result show that CRO outperforms many commonly used strategies for this problem. Minimizing the number of coding links in a network for a given target transmission rate to improve the network efficiency is a nondeterministic polynomial time hard problem. Pan et al. (Pan et al., 2011) adopted chemical reaction optimization to develop an algorithm for solving this complex problem and found that the CRO-based framework outperformed any other existing algorithm. The CRO has also been successfully applied to a grid scheduling problem (Xu et al., 2010) and task scheduling problem (Xu et al., 2011b). The authors have compared the efficiency of several versions of CRO with other algorithms such as GA, simulated annealing (SA), and PSO and found it to be superior. CRO is also used to replace backpropagation-based training of ANNs in a classification problem (James et al., 2011). The simulation results show that the CRO-based ANN outperformed other optimization techniques such as GA and support vector machine (SVM). Allocating available channels to unlicensed users in order to maximize the utility of radio channels is a complex task. A CRO-based algorithm has been developed by the authors (Lam & Li, 2010) to solve the radio spectrum allocation problem, and it outperforms other evolutionary techniques. Two different types of chemical reaction optimization, canonical CRO and super molecule based CRO (S-CRO), were proposed in (Xu et al., 2011a) for the problem of stock portfolio selection; the results suggested that S-CRO is promising in handling the stock portfolio optimization problem. A new chemical reaction optimization with a greedy strategy algorithm (CROG) was proposed to solve the 0–1 knapsack problem (Truong et al., 2013). The article designed a new repair function integrating a greedy strategy, and random selection is used to repair the infeasible solutions. The experimental results show the superiority of CROG over GA, ant colony optimization, and quantum-inspired evolutionary algorithms. Applications of CRO for forecasting BSE stock prices are found in (Nayak et al., 2013; Nayak et al., 2017a; Nayak et al., 2017b).

The above studies have found ANN, FL, and CRO to be highly effective soft and evolutionary computing techniques for data mining problems. Also, the review of the related literature showed the unique promise of neuro-fuzzy hybrid networks based on CRO in accurately predicting stock prices.

Chemical reaction optimization

This section describes the basic concepts of artificial chemical reaction optimization. The concept loosely couples mathematical optimization techniques with properties of chemical reactions. A chemical reaction is a natural process of transforming unstable chemical substances (reactants/molecules) into stable ones. A chemical reaction starts with unstable molecules with excessive energy. The

molecules interact with each other through a sequence of elementary reactions, producing intermediate chemical products. At the end of the process, they are converted into molecules that require minimum energy to support their existence. The energy associated with a molecule is called enthalpy (minimization problem) and/or entropy (maximization problem). During a chemical reaction this energy changes with the change in the intra-molecular structure of a reactant and becomes stable at a certain point. These reactions may be monomolecular or bimolecular depending on the number of reactants taking part. CRO algorithm begins with a set of initial reactants in a solution. Then reactants are consumed and produced via chemical reactions. The algorithm is terminated when the termination criterion is met, which is similar to the state when no further reactions can take place (inert solution). According to the above algorithm concept, CRO can be represented by Fig. 1.

There are two major components of CRO, i.e., the molecule and the elementary reactions, which are discussed in the following subsection.

Molecule

A molecule is the basic manipulated agent in CRO, similar to an individual in optimization techniques. A change in molecular structure means switching to another potential solution in the search space. The energy associated with a molecule is termed KE (kinetic energy) and PE (potential energy). The transformation of a molecule m to m' is only possible when $PEm' \leq PEm + KEm$. KE helps a

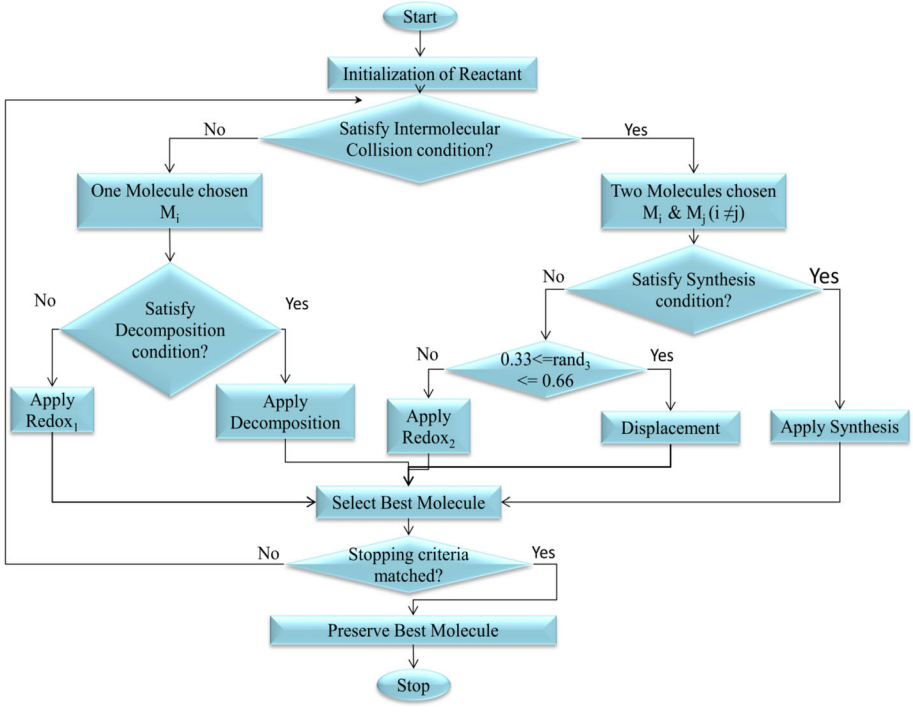

Fig. 1 Chemical reaction optimization process

molecule to shift to a higher potential state and provides the ability to evade local optima. Hence, there are more chances to have favorable structures in future alterations. In CRO, the inter conversion between the KE and PE among molecules can be achieved through certain elementary chemical reactions, similar to the number of steps in optimization techniques. As the algorithm evolves, the molecules reach lower and lower energy states (KE and PE) and ensure convergence.

Elementary chemical reactions

Different chemical reactions are applied as operators for exploration as well as exploitation of the search space. Based on the number of molecules taking part in a chemical reaction, the reactions may be divided into two categories: monomolecular (one molecule takes part in the reaction) or bimolecular (two molecules taking part in the chemical reaction). The monomolecular reactions (Redox1 and Decomposition) assist in intensification, while the bimolecular reactions (Synthesis, Redox2, and Displacement) give the effect of diversification. We explain the chemical reactions in the following subsections using binary encoding of molecules.

Decomposition reaction

Decomposition reaction occurs when a molecule splits into two fragments on a collision with the wall of the container. The products are quite different from the original reactants. In general, we represent the decomposition of a molecule, m, into $m1'$ and $m2'$ as follows:

$$\underbrace{[0,1,1,0,1]}_{m} \rightarrow \underbrace{[1,1,1,0,1]}_{m1'} + \underbrace{[0,1,0,0,1]}_{m2'}$$

We examine every bit value of m. If m(i) is equal to one, its value is copied to ${m_1}'$ (m), and the value of ${m_2}'$ (i) is set at random. If m(i) is equal to zero, its value is copied to ${m_2}'$ (i), and the value of ${m_1}'$ (i) is set at random. Since ${m_1}'$ and ${m_2}'$ are completely different, they can be treated as two different solutions in the search space and may increase the exploration capability of the CRO. The reaction, $m \rightarrow {m_1}' + {m_2}'$, is acceptable only if $P\ (m1'\ ) + PE(m2'\ ) > KE\ (m) + PE\ (m)$.

Redox1 reaction

In this reaction, a molecule is allowed to collide on the wall of the container. This is also called on-wall-ineffective collision. As a result, a small change to the molecular structure happens. A new product, m', is formed on flipping a random bit of m as follows:

$$\underbrace{[1,0,1,1,0]}_{m} \rightarrow \underbrace{[1,0,1,0,0]}_{m'}$$

The chemical system, $m \rightarrow m'$, is acceptable if $K\ (m) + PE\ (m) < PE\ (m'\ )$, otherwise it is rejected.

Synthesis reaction

Here two molecules, m_1 and m_2, that synthesize to form a single product, m', that is much different from the original molecules. The reaction can be shown as follows:

$$\underbrace{1,0,1,1,0,1}_{m1} + \underbrace{1,1,0,1,0,1}_{m2} \rightarrow \underbrace{1,1,0,1,0,1}_{m'}$$

Here, the corresponding bit values of the reactants are compared. If there is a matching, then the bit value of any molecule is copied to the product. If there is no matching, then either the bit value of $m1$ or $m2$ will be copied based on a random value. The new chemical system, $m1 + m2 \rightarrow m'$, is acceptable if $K\ (m1) + KE\ (m2) + PE\ (m1) + PE\ (m2) < PE\ (m'\ )$.

Redox2 reaction

In this type of reaction, two molecules, $m1$ and $m2$, are reacting with each other to produce two new products, $m1\ '$ and $m2\ '$. This reaction can be represented as follows:

$$\underbrace{1,0,1,1,0,1}_{m1} + \underbrace{0,0,1,0,1,1}_{m2} \rightarrow \underbrace{1,0,1,0,0,1}_{m1'} + \underbrace{0,0,1,1,1,1}_{m2'}$$

We select two random points within one along the length of the reactant. Then, the bit values between these points are swapped to get two new products. If $K\ (m1) + K\ (m2) + PE\ (m1) + PE\ (m2) < PE\ (m1'\ ) + PE\ (m2'\ )$, then the chemical system $m1 + m2 \rightarrow m1'\ + m2'$will be accepted, otherwise it is rejected.

Displacement reaction

Here two new molecules are formed as products of the collision of two reactants. The reaction can be represented as follows:

$$\underbrace{1,0,1,1,0,1}_{m1} + \underbrace{0,0,1,0,1,1}_{m2} \rightarrow \underbrace{0,0,1,1,1,1}_{m1'} + \underbrace{1,0,1,0,0,1}_{m2'}$$

We compare the corresponding bit values of the two reactants. We swapped the bit values based on a random value to produce new products. If $KE(m1) + KE(m2) + PE(m1) + PE(m2) < PE\ (m1'\ ) + PE(m2'\ )$, the chemical system $m1 + m2 \rightarrow m1'\ + m2'$ will be accepted, otherwise it is rejected.

Reactant update

In this step, a chemical equilibrium test is performed. If the newly generated reactants provide a better function value, the new reactant set is included and the worse reactant is excluded,similar to reversible chemical reactions. The reactants are updated according to their enthalpy (fitness value).

Termination criterion check

The CRO is terminated when the termination criterion (i.e., maximum number of iterations) has been met.

Artificial chemical reaction is more robust and uses less tunable parameters compared to other optimization methods (Alatas, 2011; Alatas, 2012; Lam & Li, 2012).

It needs only the number of initial reactants. In this work, we adopted the basic algorithm of chemical reaction optimization suggested in (Alatas, 2011; Alatas, 2012). We used binary encoding for the reactants and the uniform population method for the initial population generation. The initial reactants are evenly initialized in the feasible searching space. As per this method, all vectors in a space can be obtained as a linear combination of elements of the base set. Absence of one of elements in the base set creates a reduction in that dimension corresponding to this element. Therefore, it is important that initial reactants contain reactants that hold each element of the base set. Also, the initial reactants must be regular and hold the base set. The uniform population method, which can be used to generate the initial reactant pool, is defined by Algorithm 1.

Algorithm 1: **Generate Initial Reactant**

*/*R is the reactants set; I is the indices set and I_e is the enlarged indices set*/*

1. *Create two reactants such as one of them R[1] contains all upper bounds for variables and the other R[2] contains all lower bounds for variables.*
2. *Index ←3*
3. *k ←2*
4. ***While** R is not saturated do*
   - *4.1.Let i_e be an element of I_e and each i_e are enlarged with bit value and this bit value corresponds to part.*
   - *4.2.i ←1*
   - *4.3.**While** R is not saturated and all reactants are not generated for a specific value of k (and r <= 2^k-2) **do***
   - *4.4.i is a k-bit number and i_e corresponds to the enlarged value of i. Each bit of I is enlarged up to length of corresponding part of R[0] and R[1].*
     - ***for** j←1 to n **do***
       - ***if** j^{th} bit of i_e is 1 then*
         - *j^{th} value of R[Index] is equal to R[1]*r*
       - ***else***
         - *j^{th} value of R[Index] is equal to R[2]*r*
       - ***end if***
       - *r is a random real number in interval [0, 1]*
     - ***end for***
   - *4.5.Index ←Index +1*
   - *4.6.i←i+1*
   - *4.7.**End While***
   - *4.8.k←k+1*
5. ***End While***

There are many nature-inspired evolutionary algorithms, and variants of them have been proposed and applied to solving nonlinear problems. However, their performance may vary from one dataset to another. According to the "no free lunch theorem" there is no single state of the art constraint handling technique that can outperform all others on every problem. Hence, choosing a suitable optimization technique for solving a particular problem involves numerous trial and error methods. The efficiency of these optimization techniques is characterized by tuning the parameters. For better convergence of the algorithm, suitably fine-tuned parameters are required. In order to search for the global optimum solution, the algorithm requires the appropriate selection of parameters, which makes use of the algorithm difficult. Hence, an optimization technique requiring fewer parameters, small number of computations, as well as good approximation capability will be the better choice. Chemical reaction optimization is one such technique (Alatas,

2011; Alatas, 2012). These facts motivated us to adopt CRO over others. The basic CRO algorithm is discussed in Algorithm 2. As mentioned in this algorithm, it is only necessary to set the value for the initial reactant.

Algorithm 2: Chemical Reaction Optimization

```
1.  Set ReacNum                          /*Define number of initial reactants*/
2.  Initialization of search spaces R /*Initialize using Algorithm 1*/
3.  Set IterationNum = 0
4.  for  i = 1 to ReacNum do
          Calculate the enthalpy e (Mi)
5.  end for
6.  While (termination criterion not met) do
7.  for i = 1 to ReacNum do
    /*Apply all reactions over the reactants of Mi */
    Get rand1 randomly in interval [0, 1]
          if rand1 ≤ 0.5 then
             Get rand2 randomly in interval [0, 1]
             if rand2 ≤ 0.5 then
                  Decomposition (Mi)
             else
                  Redox1 (Mi)
             end if
          else
             Select another molecule Mj (Mi ≠ Mj)
             Get rand3 randomly in interval [0, 1]
          if 0 ≤ rand3 ≤ 0.33 then
                Synthesis (Mi, Mj)
         else if 0.33 ≤ rand3 ≤ 0.66 then
               Displacement (Mi, Mj)
             else
               Redox2 (Mi, Mj)
             end if
         end if
       Apply Reversible Reaction for increased enthalpy to update reactants
8.  end for
9.  IterationNum=IterationNum+1
10. End While
11. Employ the reactant with best enthalpy as the optimal weight & biases set for the CNFN
    model.
```

CNFN method

This section presents the CRO-based neuro-fuzzy (CNFN) forecasting model. The overall high-level steps of the model are shown in Fig. 2.

The proposed method uses a single hidden layer MLP as the base model. The input vector to the model is expanded into different membership functions. The expanded input vector in effect increases the dimensionality of the input vector. Hence, the hyper plane generated by the fuzzy net provides greater discrimination capability in the input pattern space. The Gaussian membership function (MF) smooths the transition between members and non-members, in comparison to triangular and trapezoidal MF. When compared with a bell MF, it has fewer parameters, which makes it easier to use. Gaussian MFs are not triangular, rather they overcome some of the drawbacks of triangular MF, such as avoiding the straight line segment and sharp bends at the corner points in triangular MF. The Gaussian membership function is used in the fuzzification process, which is specified by two parameters {c, σ}, as shown in Eq. 1.

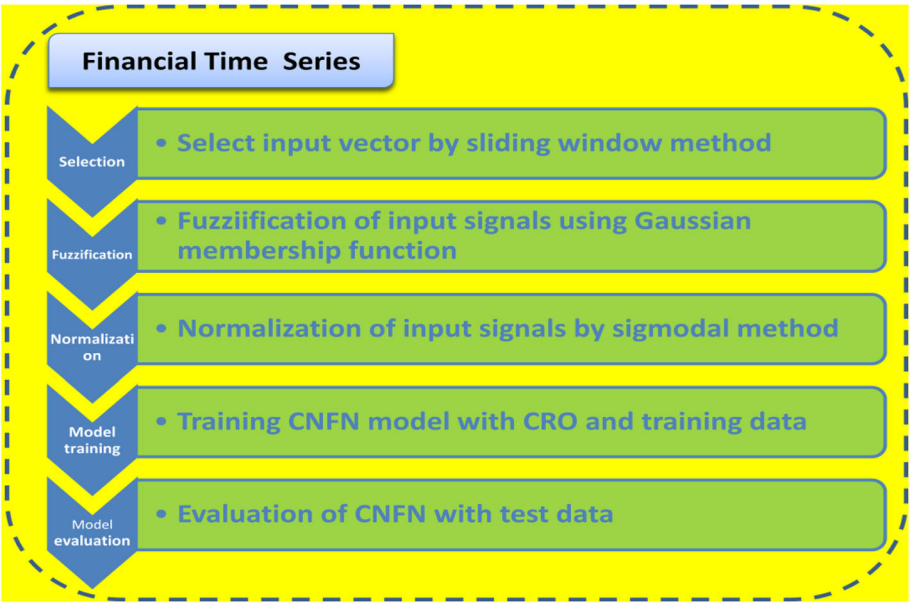

Fig. 2 Flow diagram for CNFN based forecasting

$$gaussian\,(x; c, \sigma) = e^{-\frac{1}{2}\left(\frac{x-2}{\sigma}\right)^2}, \tag{1}$$

where x is the input, c is the center of the pattern, and is the width or thickness of the input pattern. By considering the smallest, medium, and biggest values as the center of the input pattern, each unit in the input pattern has a triangular Gaussian membership function as its inner function, given by Eqs. 2–4.

$$O_{i,1}(x_i, sm_i, th_i) = e^{-\frac{1}{2}\left(\frac{x_i - sm_i}{th_i}\right)^2}. \tag{2}$$

$$O_{i,2}\,(x_i, me_i, th_i) = e^{-\frac{1}{2}\left(\frac{x_i - me_i}{th_i}\right)^2}. \tag{3}$$

$$O_{i,3}\,(x_i, bg_i, th_i) = e^{-\frac{1}{2}\left(\frac{x_i - bg_i}{th_i}\right)^2}, \tag{4}$$

where *xi* is the *ith* input signal, *smi* is the low value of the *ith* input vector, *bgi* is the high value of the *ith* input vector, and*mei* is the medium value of the *ith* input vector, and it is taken as $\frac{bg_i - sm_i}{size\ of\ input\ pattern}$.Each input data point is the input of these membership functions (low, medium, high), and the outputs of the units O_{ij} ($i = 1 \cdots N$, $j = 1, 2, 3$) are the grades of the membership functions. Let the input vector be represented as $X = [x_1, x_2, \cdots\cdots, x_N]\ T$, where N is the number of closing prices in the input vector and 'T' denotes the matrix transpose operation. Let us assume that there are M numbers of classes in the fuzzification process. After the fuzzification process, the output is an$N \times M$ membership matrix described as follows:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_N \end{bmatrix}^T = \begin{bmatrix} mf_{1,1}(x_1) & mf_{1,2}(x_1) & mf_{1,3}(x_1) \\ mf_{2,1}(x_2) & mf_{2,2}(x_2) & mf_{2,3}(x_2) \\ mf_{3,1}(x_3) & mf_{3,2}(x_3) & mf_{3,3}(x_3) \\ \cdots & \cdots & \cdots \\ mf_{N,1}(x_N) & mf_{N,2}(x_N) & mf_{N,3}(x_N) \end{bmatrix}, \tag{5}$$

Where $mf_{i,j}(xi)$ is the membership of thei^{th} input xi (i = 1,2, ⋯, N) and(j = 1,2,3, ⋯, M).

Now the membership matrix is converted into a vector of length $N \times M$, and this vector is supplied as input to the MLP. The MLP considered here contains a single hidden layer, and the MLP-based predictor is presented in Fig. 3.

The MLP can approximate any arbitrary function in terms of its mapping abilities. The error correction learning, in this case, is supervised learning, i.e., the desired response for the system is presented at the output neuron. This model consists of a single output unit to estimate one-day-ahead closing prices. The neurons in the input layer use a linear transfer function the neurons in the hidden and output layers use a sigmoidal function as follows:

$$Y_{out} = \frac{1}{1 + e^{-\lambda y_{in}}}, \tag{6}$$

where y_{out} is the output of the neuron, λ is the sigmoidal gain, and y_{in} is the input to the neuron. The first layer corresponds to the problem input variables, with one node for each input variable. The second layer is useful in capturing nonlinear relationships among the variables. At each neuron, j, in the hidden layer, the weighted output Z is calculated using Eq. 7.

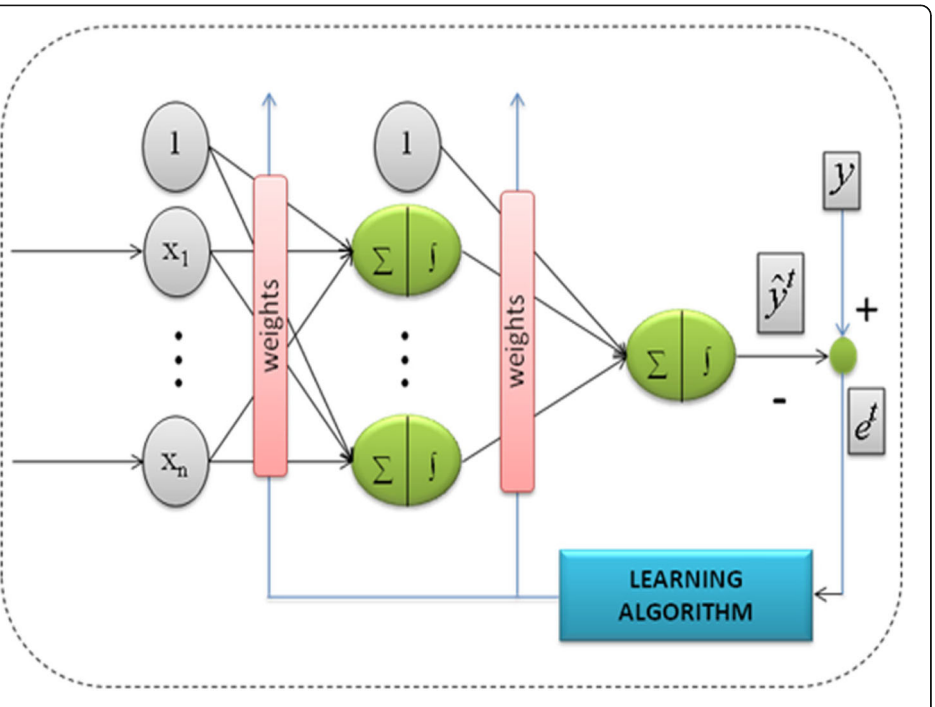

Fig. 3 MLP based predictor

$$Z = f\left(B_j + \sum_{i=1}^{n} V_{ij} * O_i\right), \tag{7}$$

where *Oi* is the *ith* component of the $N \times M$ input vector, *Vij* is the synaptic weight value between the *ith* input neuron and *ith* hidden neuron, *Bj* is the bias value, and *f* is a nonlinear activation function. The output y at the single output neuron is calculated using Eq. 8.

$$y = f\left(B_o + \sum_{j=1}^{m} W_j * Z\right), \tag{8}$$

where *Wj* is the synaptic weight value from the*jth* hidden neuron to the output neuron, Z is the weighted sum calculated in Eq. 7, and B_0 is the output bias. This output y is compared to the target output and the error is calculated by using Eq. 9.

$$Err_i = |T\ \text{arget}_i - Estimated_i|, \tag{9}$$

where Err_i is the error signal, $Target_i$ is the target signal for the i^{th} training pattern, and $Estimated_i$ is the calculated output for the i^{th} pattern. The error signal $Er(i)$ and the input vector are employed to the weight update algorithm to compute the optimal weight vector. During the training, the network is repeatedly presented with the training vector, and the weights and biases are adjusted through CRO till the desired input-output mapping occurs. The error is calculated by Eq. 9, and the objective is to minimize the error function, as shown in Eq. 10, with an optimal set of network parameters.

$$E(i) = \frac{1}{2} \sum_{i=1}^{N} Err\,(i)^2 \tag{10}$$

The overall architecture of the CNFN model is presented in Fig. 4. The number of neurons in the input layer is equal to the number of components in the membership matrix, i.e., $N \times M$. The number of hidden neurons in the hidden layer is selected experimentally. Since the model forecasts one closing price, there is one neuron in the output layer.

The basic element of CRO is an atom, which can be visualized as a weight or bias value of the network. A set of such atoms (weight and biases) constitute a molecule. For simplicity, the number of input layer neurons is represented as $NOIN = N*M$. Let the number of neurons in the hidden layer, $NOHN =$, and a single output neuron form the CNFN. Hence, the total number of weight values is $(NOIN*NOHN) + (NOHN*\ 1) = (N*M*H) + (H*\ 1)$. Therefore, the length of a molecule is equal to $[(N*M*H) + (H*\ 1) + 2\ biasvalues)$. Fig. 4 represents a molecule of CRO. As shown in the figure, let the weight values between the input layer and hidden layer be represented as $[V11,12, \cdots, VNM*H]$. The weight values between the hidden layer and output layer are represented as $[W1, W2, \cdots, WH]$. The two bias values to the hidden and output neurons are $[B0, B1]$.

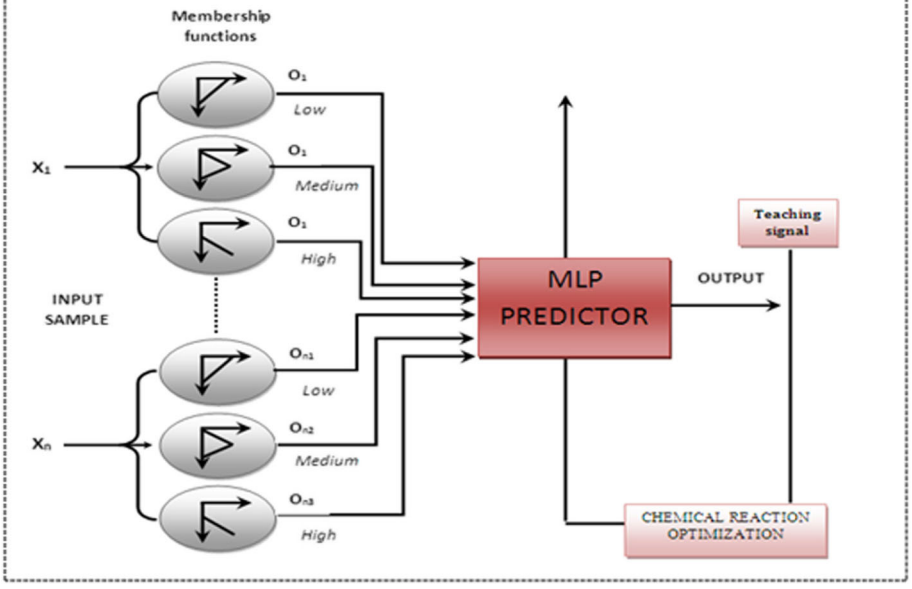

Fig. 4 Architecture of CNFN based forecasting model

A reactant is associated with some enthalpy (minimum prediction error signal in this case), which can be considered as the fitness value of that reactant. The mean absolute percentage of error (MAPE) value is treated as the enthalpy of a reactant and calculated in Eq. 11.

$$MAPE = \frac{\sum_{i=1}^{N} |Actual_i - Estimated_i|}{N}, \tag{11}$$

where N is the number of input patterns in the training set.

A reactant, as shown in Fig. 5, can be seen as a potential solution for the proposed model. A set of such reactants form a reactant pool, which can be visualized as a search space consisting of possible weight and bias vectors for the model. The search process in CRO starts with an initial reactant pool of random values. Each molecule of the pool, along with the training data, is supplied to the model one by one. The model estimates on output. The corresponding enthalpies are calculated by comparing the model estimation with the desired or target output using Eq. 9. Different chemical reactions (Synthesis, Displacement, Decomposition, Redox1, and Redox2) are applied to the molecules for exploration as well as exploitation of the search space. The fitness of the molecules is evaluated, and the reactant pool is updated with the molecules that have lower enthalpy values. The lower the enthalpy value of a molecule/reactant, the better its chance of being included in the reactant pool. These steps constitute an iteration of

Weight Values								Bias Values	
Input and Hidden Layer				Hidden and Output Layer				Hidden	Output
V_{11}	V_{12}	...	V_{NM*H}	W_1	W_2	...	W_H	B_0	B_1

Fig. 5 Reactant representations for CNFN model

CRO. A finite number of such iterations are carried out, and the CRO approaches the global minima. At the end, the reactant pool becomes inert (equilibrium state), and the search process is assumed to be converged. The molecule with the best enthalpy value is selected as the optimal parameter of the model. This molecule along with the test data is supplied to the same model, and the model estimates an output. The difference between the actual signal at the output neuron and the model output is calculated and preserved as the error signal for the respective test pattern. In this way, the error signals for all the train and test patterns of the data set are evaluated. The average of these error signals is calculated by Eq. 11 and considered as the prediction performance of the model. The lower the value of the average error, the more accurate the model is. The high-level training algorithm for the CNFN model is described by Algorithm 3.

Algorithm 3: CNFN Training

1. *Initialize reactants*
2. *Set training and test data*
   / choosing number of closing prices as input vector for the network*/*
3. *Fuzzification of training and test data*
   */*Fuzzification of each closing price to form a membership matrix applying Gaussian membership functions using Eq. 2 – 4 */*
4. *Normalization of fuzzified training and test data*
5. *Supply training data and reactant pool to the MLP*
6. *Apply Chemical Reactions to optimize reactant pool*
   / apply Algorithm2 for training phase*/*
7. *Supply test data and the best reactant to the MLP and preserve error signal*
8. *Repeat the steps 2-7 for all training and testing pattern and calculate the total error signals*

Experimental results and discussion

This section presents the forecasting results obtained by employing the forecasting models discussed above. These models are validated on five real stock indices: the DJIA, BSE, NASDAQ, TAIEX, and FTSE. All the experiments are carried out in a MATLAB-2015 environment, with Intel® core TM i3 CPU, 2.27 GHz processing and 2.42 GB memory size.

Data description, input selection, and preprocessing

The model is validated on five real stock indices: DJIA, BSE, NASDAQ, TAIEX, and FTSE. The closing prices are collected for each transaction day of the stock exchanges from January 1, 2003 to September 12, 2016 (source: https://in.finance.yahoo.com/). These data are publicly available. The historical daily closing prices that form the financial time series are considered for experimentation. The datasets and total number of data points in each set are summarized in Table 2. The closing prices are plotted in Fig. 6.

For more clarity on the data analysis, the descriptive statistics of daily closing prices are summarized in Table 3. As can be observed, the positive skewness values of the

Table 2 Datasets considered for experiment

Short name	Long Name	Total number of data points
BSE	Bombay Stock Exchange	3738
DJIA	Dow Jones Industrial Average	3449
NASDAQ	National Association of Securities Dealers Automated Quotation System	3447
FTSE	Financial Times Stock Exchange	3570
TAIEX	Taiwan Capitalization Weighted Stock Index	3708

closing price result in the BSE, DJIA, and NASDAQ datasets being spread out toward right, which is a good symptom of investment opportunities. The kurtosis analysis implies that the stock prices of the DJIA and NASDAQ more prone to outliers, whereas all the other financial time series are less outlier prone. Also, from the Jarque-Bera test statistics, it can be observed that all the stock price datasets are non-normally distributed. From the histogram of daily returns for the TAIEX and BSE, as presented in Fig. 7, it can be observed that the peaks of the histograms are much higher than those corresponding to the normal distribution. In the case the BSE stock data, it is slightly skewed to the right, and in the case of the TAIEX stock data, it is slightly skewed to the left.

Stationarity in financial time series

Next, we check the stationarity of all financial time series as this is important for obtaining valid forecasts. Therefore, we first investigate the financial time series for the presence of a unit root to determine whether the analyzed time series are stationary or not. We conducted the stationarity test using two well-known unit root tests: The

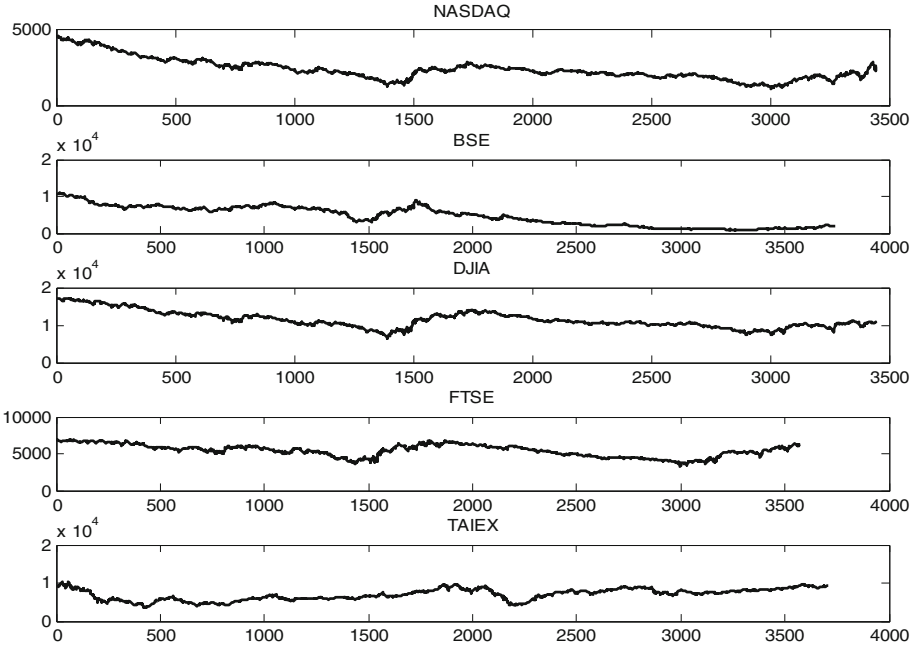

Fig. 6 Daily closing indices of NASDAQ, BSE, DJIA, FTSE, and TAIEX stock market during 01 January 2003 to 12 September 2016

Table 3 Descriptive statistics from BSE, DJIA, NASDAQ, FTSE, and TAIEX financial time series

Stock Index	Descriptive statistics						
	Minimum	Maximum	Mean	Standard deviation	Skewness	Kurtosis	Jarque-Bera test statistics
BSE	792.1800	1.1024e + 004	4.6235e + 003	2.6947e+ 003	0.1154	1.7908	236.0430(h = 1)
DJIA	6.5471e + 003	1.7138e + 004	1.1400e + 004	2.1801e+ 003	0.6644	3.0512	253.8134(h = 1)
NASDAQ	1.1141e + 003	4.5982e + 003	2.3858e + 003	709.7888	1.0392	4.0027	764.3663(h = 1)
FTSE	3287	6.8785e + 003	5.4165e + 003	836.2381	−0.2837	2.1378	158.4568(h = 1)
TAIEX	3.4463e + 003	1.0202e + 004	6.9835e + 003	1.4846e+ 003	−0.1776	2.0465	159.9786(h = 1)

Augmented Dickey-Fuller (ADF Test) and Phillips-Perron (PP Test). The null hypothesis and alternative for the standard Dickey-Fuller test can be written as follows:

H_0: $\alpha = 0 = >$ the series has a unit root, so it is non-stationary and.

H_{alt}: $\alpha < 0 = >$ the series does not have a unit root, so the time series is stationary. These are evaluated using the t_α statistic. The t_α statistic is calculated as $t\alpha = \hat{a} / s(\hat{a})$, where α represents the estimate of α and $(se(\hat{a}))$ represents the coefficient of standard error. The value of the t_α statistics are calculated and then compared with the critical values of the ADF test. If the value of the t_α statistic is less than the critical value, then we reject the null hypothesis, meaning there is no unit root and the time series are stationary.

An alternative to the ADF test is the Phillips-Perron test (PP test), which is a non-parametric method that is very similar to the ADF test. However, the PP test allows the residue to be auto correlated by introducing an automatic correction into the testing procedure. The mathematical details of these tests are beyond the scope of this article.

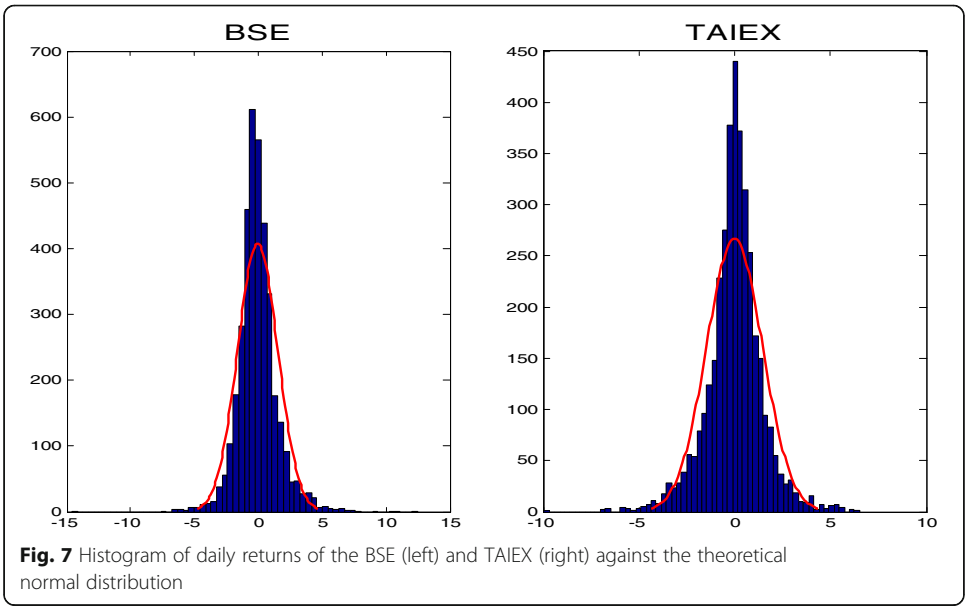

Fig. 7 Histogram of daily returns of the BSE (left) and TAIEX (right) against the theoretical normal distribution

We used the ADF test and PP test available in the MATLAB software to check for time series stationarity. The outputs obtained from these tests are summarized in Table 4.

It can be seen from Table 4 that there is non-stationarity in levels and stationarity in the first difference for all five time series analyzed by applying the Augmented Dickey-Fuller and Phillips-Perron tests. This conclusion is in accordance with the Box-Jenkins approach to modeling time series, which states that financial time series are non-stationary.

Input selection

We use a sliding window of fixed size for selecting the input for the forecasting model. In this method, rather than selecting all the data seen so far, or some sample therein, we make decisions based only on recent closing prices. On each sliding of the window, a new closing price is incorporated and the oldest one is discarded. The window moves through the whole financial time series, and the selection of window size is a matter of experimentation. For an example, the indices used for one-day-ahead and one-month-ahead are shown in Figs. 8 and 9, respectively. The total number of windows generated for each category of prediction is presented below in Table 5.

Normalization

In order to normalize the raw closing prices data, several data normalization methods have been tested, and out of these, the sigmoid normalization method was found to be superior (Nayak et al., 2014). The benefit of this normalization technique is that it does not depend on the distribution of data, which may be unknown at the time the model is trained. The normalization method is shown in Eq.12.

$$x_{norm} = \frac{1}{1 + e^{-\left(\frac{x_i - x_{min}}{x_{max} - x_{min}}\right)}}, \tag{12}$$

Table 4 The results from Augmented Dickey-Fuller test and Phillips-Perron test for five financial time series

Stock Data	Indicators	ADF-Test		PP-Test	
		Level	1st Diff.	Level	1st Diff.
BSE	t-Stat (Prob.)	−2.5460 (0.3223)	−41.5448 (0.0001)	− 2.4068 (0.3913)	−54.0093 (0.0000)
	t-critical (5%)	−3.4138	− 3.4138	− 3.4138	− 3.4138
DJIA	t-Stat (Prob.)	−2.3788 (0.4051)	−44.7766 (0.0000)	−2.4392 (0.3752)	−63.6215 (0.0000)
	t-critical (5%)	−3.4139	− 3.4139	− 3.4139	− 3.4139
NASDAQ	t-Stat (Prob.)	−2.5920 (0.2995)	−48.0225 (0.0001)	−2.5823 (0.3043)	−64.9935 (0.0001)
	t-critical (5%)	−3.4139	−3.4138	− 3.4139	−3.4138
FTSE	t-Stat (Prob.)	−2.0381 (0.5738)	−47.2026 (0.0000)	−2.1130 (0.5367)	−65.5486 (0.0000)
	t-critical (5%)	−3.4139	−3.4138	− 3.4139	−3.4138
TAIEX	t-Stat (Prob.)	−3.2628 (0.0730)	−40.8769 (0.0000)	−3.1798 (0.0889)	−58.4660 (0.0000)
	t-critical (5%)	−3.4138	−3.4138	− 3.4138	−3.4138

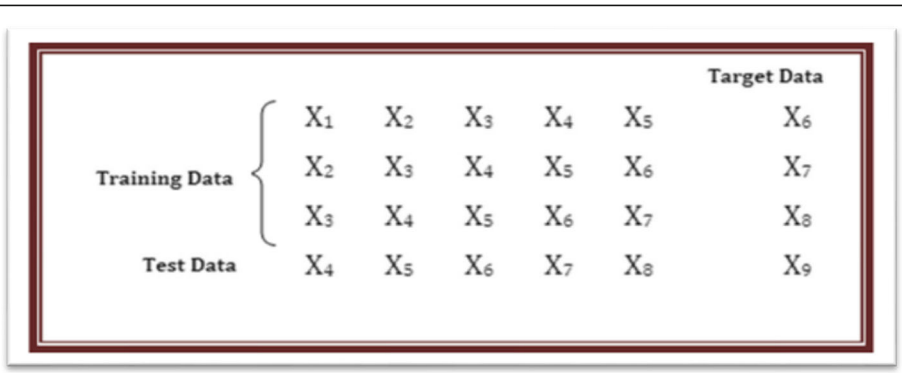

Fig. 8 Training window generation for one-day-ahead forecasting

where x_{norm} is the normalized price, x_i is the current day closing price, x_{max} and x_{min} are the maximum and minimum prices of the window, respectively. The data within the current training set are normalized within [0, 1]. Therefore, instead of using the maximum and minimum of whole data set, which may be unavailable at the time of prediction, we use the maximum and minimum of the current training data set. The record to be tested is also normalized using Eq. 12, but its value is not used for deriving the x_{max} and x_{min} values, i.e., the target value may reside outside $[x_{max}, x_{min}]$. The normalized values are now considered as the input vector to the model. Since each time the sliding window moves one step ahead, only one new closing price data point has been added to the training set.

Performance metrics

The four performance metrics used for evaluating the forecasting models are as follows: The mean absolute percentage error (MAPE) is the first performance metric to establish a comparable measure across experiments with different stock data sets. The closer the value of MAPE is to zero, the better the prediction ability of the model. The formula for MAPE is represented by Eq.13.

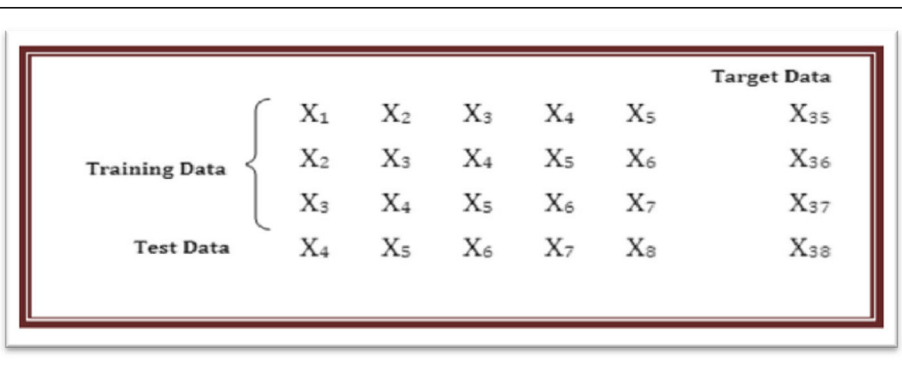

Fig. 9 Training window generation for one-month-ahead forecasting

Table 5 Number of windows generated for each category of prediction

Stock Index	One-day-ahead	One-month-ahead
BSE	3733	3708
DJIA	3444	3419
NASDAQ	3442	3417
FTSE	3565	3540
TAIEX	3703	3678

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|x_i - \hat{x}_i|}{xi} \times 100\% \tag{13}$$

The second performance metric is the mean of squared error calculated on the normalized data sets, and it is known as the normalized mean squared error (NMSE). The closer its value is to zero, the better the prediction ability of the model. The NMSE can be calculated as in Eq.14.

$$NMSE = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{x}_i)^2 \tag{14}$$

The next metric considered is the median absolute percentage error (MDAPE). It is the middle value of all the percentage errors for a data set when the absolute values of the errors are ordered by size. Like the above two metrics, an efficient model is the one in which the MDAPE value is closer to zero.

The fourth performance metric considered here is known as the coefficient of determination, or the coefficient of multiple determinations for multiple regressions represented as R2. R-squared is a statistical measure of how close the data are to the fitted regression line. The definition of R-squared is presented by Eq. 15. The ideal value of R2 is 1(one). 0 values indicate that the model explain none of the variability of the response data around its mean, whereas 1 indicates that the model explains all the variability of the response data around its mean.

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}, \tag{15}$$

where SSres is the sum of the squares of the residuals, or residual sum of squares, and is represented by Eq. 16.

$$SS_{res} = \sum_{i=1}^{N}(x_i - \hat{x}_i)^2 \tag{16}$$

Similarly, the SStot is the total sum of the squares that is proportional to the variance of the data, and it is calculated in Eq. 17.

$$SS_{tot} = \sum_{i=1}^{N} \left(x_i - \overline{X}\right)^2 \tag{17}$$

The fifth evaluation measure is the average relative variance (ARV). The ARV can be calculated as in 18.

$$\frac{\sum_{i=1}^{N} \left(\hat{x}_i - x_i\right)^2}{\sum_{i=1}^{N} \left(\hat{x}_i - \overline{X}\right)^2} \tag{18}$$

If the ARV value of the forecasting model is equal to 1, then it is considered to be the same as the mean of the financial time series. The model is considered to be performing worst, compared to the mean, if the ARV value is greater than 1. However, the model is considered to be performing better than simply calculating the mean if its ARV value is less than 1. Hence, the closer the value to 0, the more accurate the forecasting model tends to be.

For all the above calculations, x_i is the observed data, $\hat{x}_i$ is the estimated data, $\overline{X}$is the mean of the observed data, and N is the total number of observations.

Results and analysis

Extensive simulation studies are conducted to observe the performance of the proposed CNFN-based forecasting model. Separate experiments are conducted for each data set. In order to avoid the biases of the neural-based models, 20 simulations are conducted, and the averaged results are collected for comparison purposes. The performance of the proposed approach is compared with four other state-of-the-art forecasting models. The comparative models are CRO-based MLP (MLP-CRO), backpropagation-based MLP (MLP-BP), the radial basis functional neural network (RBFNN), and an adaptive neuro-fuzzy inference system (ANFIS) based forecasting model. The training and testing data sets for all the models are the same. Since only one new closing price data point is included into the training set through each move of the sliding window over the financial time series, there may not be a significant change in the nonlinear behavior of the training data set. Therefore, we have used the previously optimized weight set for the successive training instead of considering another random weight set. In this fashion, after the first training set, the number of iterations has been fixed to a small value, significantly reducing training time. During experimentation, different possible values for the model parameters were tested and the best values are recorded. Suitable parameter values obtained during simulation process are called the simulated parameters, and they are presented in Table 6.

The MAPE, NMSE, MDAPE, R2, and ARV values generated by all the forecasting models from all five stock indices are presented in Table 7. It can be observed from Table 7 that for all five data sets considered, four models (apart from MLP-BP) have quite good R2 values, very close to the ideal value. This property indicates that the models explain all the variability of the response data around its mean. Also, there is not much significant difference between the R2 values of the CNFN and MLP-CRO models. However, considering the other four metrics, a more comparative performance

Table 6 Simulated parameters for the forecasting models

Parameter	CNFN	MLP-CRO	MLP-BP	RBFNN
Learning Rate (α)	NA	NA	0.35	0.2
Momentum Factor (μ)	NA	NA	0.4	0.5
No. of iteration for 1st training set	100	150	500	500
No. of iteration for subsequent training	5	5	20	20
Reactant Number	50	50	NA	NA
Weight updated/found	CRO	CRO	Gradient descent	Gradient descent

analysis has been carried out for the models. For all data sets, it can be observed that the proposed CNFN model produces superior MAPE, NMSE, MDAPE, and ARV values compared to other models. For more clarity, the MAPE and ARV values are shown in Figs. 10 and 11, respectively. These graphs provide a clear picture of the proficiency of the CNFN model. The average MAPE generated by the CNFN model over all five stock data sets is 0.004436, whereas it is 0.006701, 0.042947, 0.008261, and 0.006849 for the MLP-CRO, MLP-BP, ANFIS, and RBFNN models, respectively.

Similarly, the simulated results from the one-month-ahead forecasting are summarized in Table 8. Here also, the CNFN model exhibits superior performance compared

Table 7 One-day-ahead forecasting comparison

Stock Index	Model	MAPE	NMSE	MDAPE	R2	ARV
DJIA	CNFN	0.003921	0.002586	0.001371	0.967062	0.003922
	MLP-CRO	0.005877	0.007463	0.002105	0.953123	0.006825
	MLP-BP	0.026483	0.028210	0.100735	0.890663	0.063320
	ANFIS	0.009267	0.005935	0.008326	0.936653	0.017438
	RBFNN	0.008309	0.008837	0.005733	0.954482	0.008492
FTSE	CNFN	0.006739	0.000215	0.002723	0.984023	0.005749
	MLP-CRO	0.008334	0.000461	0.003659	0.979107	0.008350
	MLP-BP	0.058463	0.008563	0.038740	0.923005	0.017305
	ANFIS	0.009438	0.005582	0.006637	0.958270	0.008677
	RBFNN	0.008306	0.009452	0.008920	0.910035	0.006774
BSE	CNFN	0.001739	0.000138	0.000997	0.998460	0.003575
	MLP-CRO	0.002683	0.000105	0.001179	0.986293	0.005125
	MLP-BP	0.052281	0.008567	0.008879	0.886585	0.030067
	ANFIS	0.006653	0.002256	0.004653	0.920875	0.007549
	RBFNN	0.002310	0.020644	0.004655	0.921167	0.008407
TAIEX	CNFN	0.003748	0.000206	0.004235	0.996919	0.003988
	MLP-CRO	0.008875	0.000236	0.006847	0.993565	0.007448
	MLP-BP	0.020044	0.007432	0.060552	0.894052	0.065505
	ANFIS	0.006840	0.003054	0.008804	0.960054	0.004490
	RBFNN	0.006822	0.003205	0.008825	0.962205	0.005628
NASDAQ	CNFN	0.006035	0.000886	0.002926	0.984520	0.005965
	MLP-CRO	0.007734	0.001465	0.004650	0.969155	0.008555
	MLP-BP	0.057465	0.008706	0.029874	0.926300	0.010375
	ANFIS	0.009108	0.005985	0.007635	0.950827	0.004967
	RBFNN	0.008500	0.009655	0.008966	0.940535	0.004764

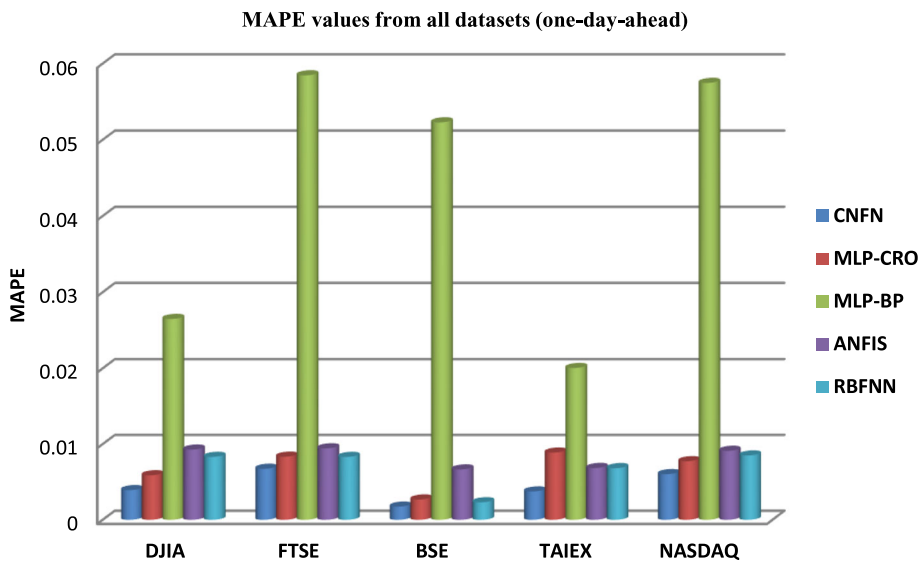

Fig. 10 MAPE generated by all models from all datasets (One-day-ahead)

to other models. The average MAPE generated by the CNFN model over all five stock data sets is 0.02656, whereas it is 0.06041, 0.08183, 0.05881, and 0.064045 for the MLP-CRO, MLP-BP, ANFIS, and RBFNN models, respectively. This also demonstrates the superiority of the proposed approach. For more clarity, the MAPE and ARV values are shown in Figs. 12 and 13 respectively. These graphs provide a clear picture of the proficiency of the CNFN model.

The proposed CNFN model differs from the ANFIS model in the following ways. The conventional ANFIS model has five layers of architecture. However, the CNFN has only three layers. A gradient descent backpropagation learning algorithm is used in the

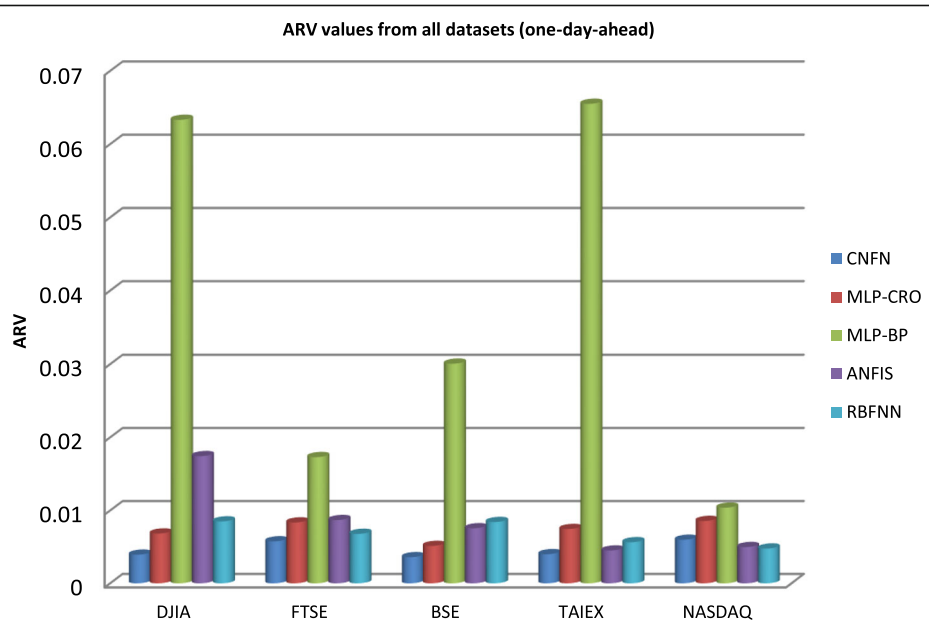

Fig. 11 ARV generated by all models from all datasets (One-day-ahead)

Table 8 One-month-ahead forecasting comparison

Stock Index	Model	MAPE	NMSE	MDAPE	R2	ARV
DJIA	CNFN	0.006825	0.005558	0.010375	0.937644	0.016392
	MLP-CRO	0.007547	0.008746	0.043172	0.903120	0.025682
	MLP-BP	0.073648	0.092821	0.371735	0.811963	0.088332
	ANFIS	0.070926	0.045935	0.025832	0.906651	0.050740
	RBFNN	0.078300	0.083835	0.033573	0.904785	0.073840
FTSE	CNFN	0.044673	0.013021	0.030725	0.925402	0.047745
	MLP-CRO	0.065837	0.016465	0.050652	0.910000	0.060352
	MLP-BP	0.075963	0.066863	0.087974	0.843025	0.065573
	ANFIS	0.049439	0.050584	0.056630	0.900274	0.078683
	RBFNN	0.050351	0.061453	0.050892	0.890000	0.058675
BSE	CNFN	0.045736	0.080013	0.002799	0.935400	0.026357
	MLP-CRO	0.085268	0.200125	0.043870	0.910062	0.038552
	MLP-BP	0.097285	0.578567	0.075872	0.806582	0.084606
	ANFIS	0.068852	0.088225	0.060465	0.900015	0.040754
	RBFNN	0.062315	0.090662	0.062465	0.905165	0.050840
TAIEX	CNFN	0.008845	0.038820	0.040023	0.906555	0.048395
	MLP-CRO	0.075872	0.042806	0.049684	0.883064	0.069744
	MLP-BP	0.073500	0.278432	0.099054	0.800455	0.446558
	ANFIS	0.065845	0.040305	0.087800	0.875351	0.140480
	RBFNN	0.070685	0.039920	0.080827	0.872735	0.058062
NASDAQ	CNFN	0.026733	0.020825	0.020925	0.920045	0.039596
	MLP-CRO	0.067535	0.050146	0.024657	0.909350	0.048577
	MLP-BP	0.088746	0.089570	0.075987	0.815630	0.090037
	ANFIS	0.039010	0.052986	0.070063	0.858825	0.040896
	RBFNN	0.058574	0.060965	0.071089	0.840500	0.042576

ANFIS for weight and bias optimization. On the other hand, CRO is characterized by a faster convergence rate, better performance, and less tunable control parameters. Hence, the CNFN is computationally and performance-wise better than the ANFIS.

Further, the performance of the CNFN can be evaluated by plotting the estimated closing prices against the actual prices of the individual stock data sets. Figs. 14, 15, 16, 17 and 18 show the plots for all five data sets for the one-day-ahead forecasting. From these plots it can be observed that the estimated prices from the CNFN model are quite close to the actual prices, which establishes the accuracy of the model.

Deibold-Mariano test

To find out the precise advantage of the proposed model, we chose the Deibold-Mariano (DM) test of statistical significance (Diebold & Mariano, 2002; Harvey et al., 1997). The DM Test is a pair-wise comparison of two or more time series models and is available for forecasting a particular variable of interest. Let the actual time series be $\{y_t; t = 1, \cdots, T\}$, and the two forecasts are $\{\hat{y}1t; t = 1, \cdots, T\}$ and $\{\hat{y}2t; t = 1, \cdots, T\}$, respectively. The objective is to test whether the forecasts are equally good. The DM statistic is defined as:

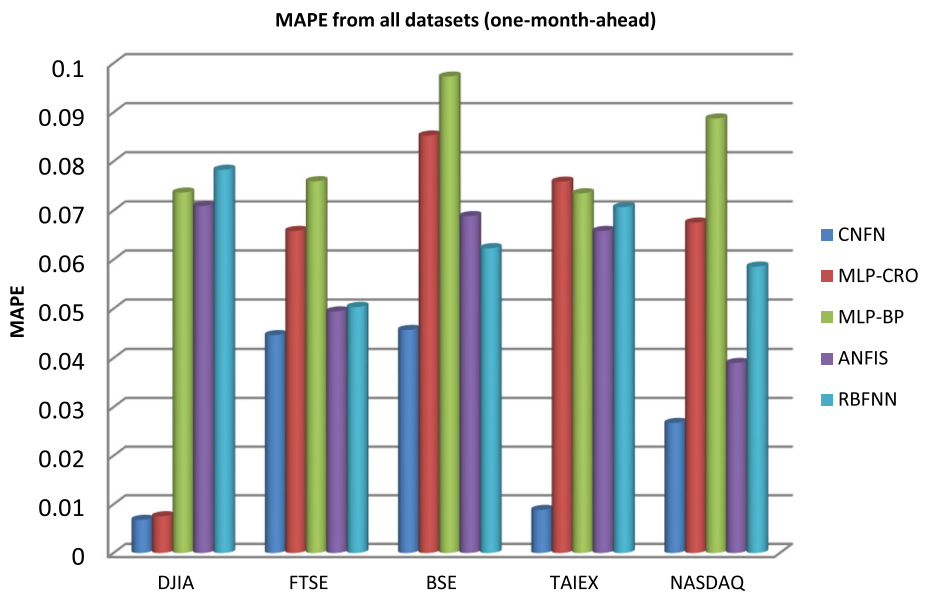

Fig. 12 MAPE generated by all models from all datasets (One-month-ahead)

$$\mathrm{DM} = \frac{\overline{\mathrm{d}}}{\sqrt{\frac{\hat{\gamma}_{\mathrm{d}}(0) + 2\sum_{\mathrm{k}=1}^{\mathrm{h}-1}\hat{\gamma}_{\mathrm{d}}(\mathrm{k})}{\mathrm{T}}}}, \tag{19}$$

where $\overline{\mathrm{d}}$ is the sample mean of the loss differential, h is the forecast horizon, and $\hat{\gamma}$ d (k) is an estimate of the auto-covariance of the loss differential γ d (k) at lag k. The null hypothesis of no difference will be rejected if the DM statistic value falls outside the range of $-z\propto/2$ to$z\propto/2$, i.e., $|\mathrm{DM}| > z\propto/2$, where $z\propto/2$is the upper z-value from the standard normal table corresponding to half of the desired ∝ level of the test. Consider the

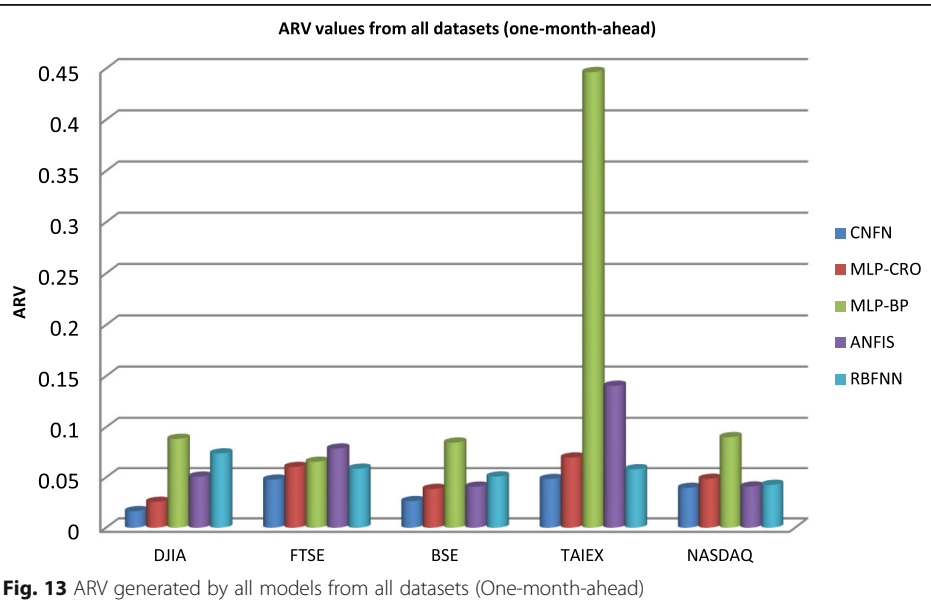

Fig. 13 ARV generated by all models from all datasets (One-month-ahead)

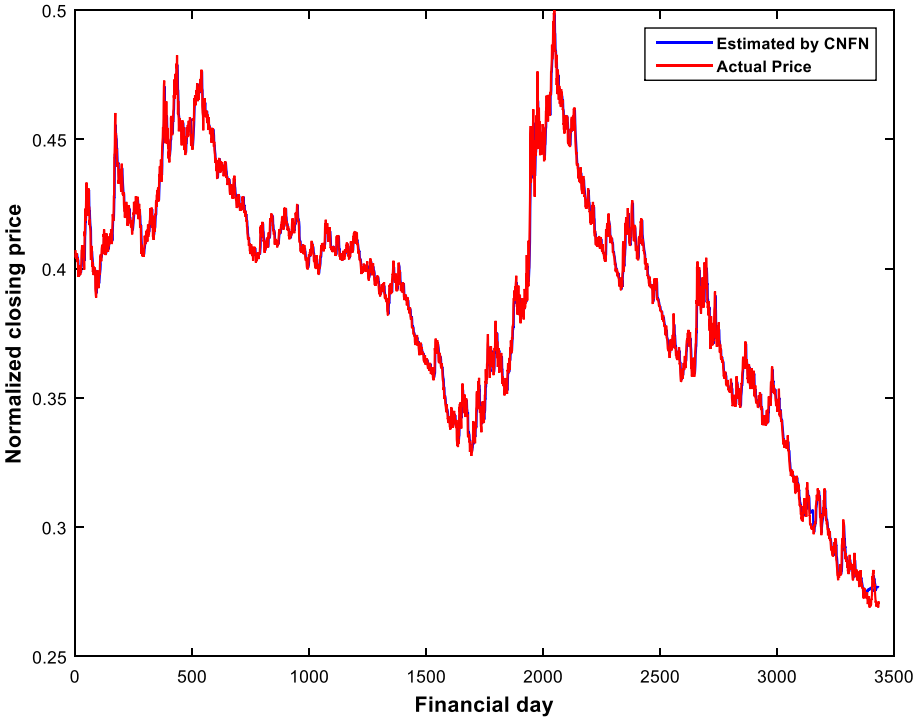

Fig. 14 Estimated v/s actual closing prices by CNFN forecasting model for DJIA stock market data

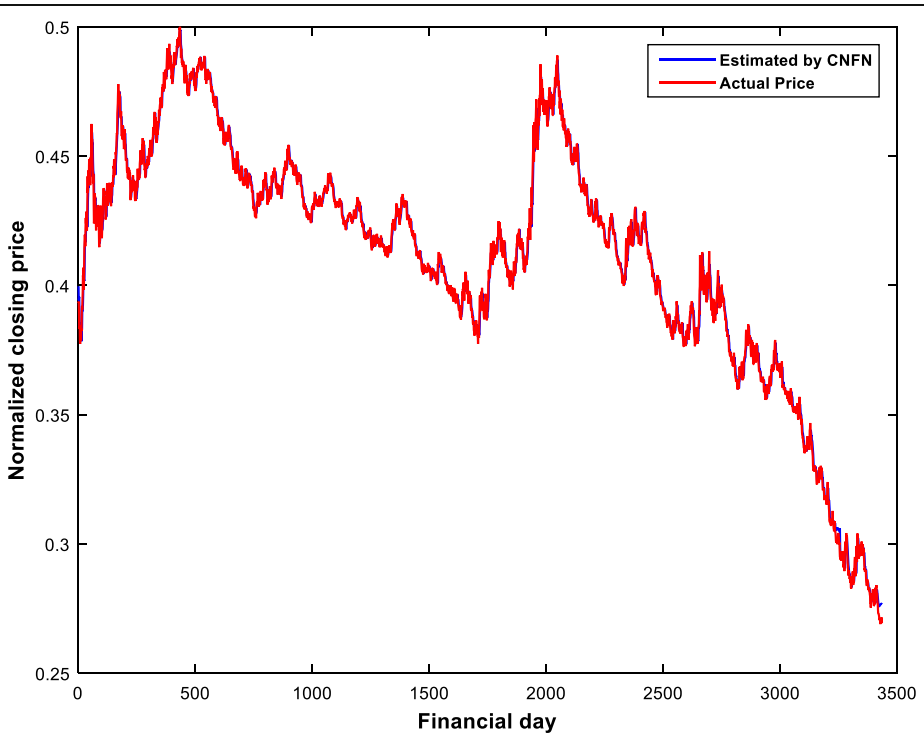

Fig. 15 Estimated v/s actual closing prices by CNFN forecasting model for NASDAQ stock market data

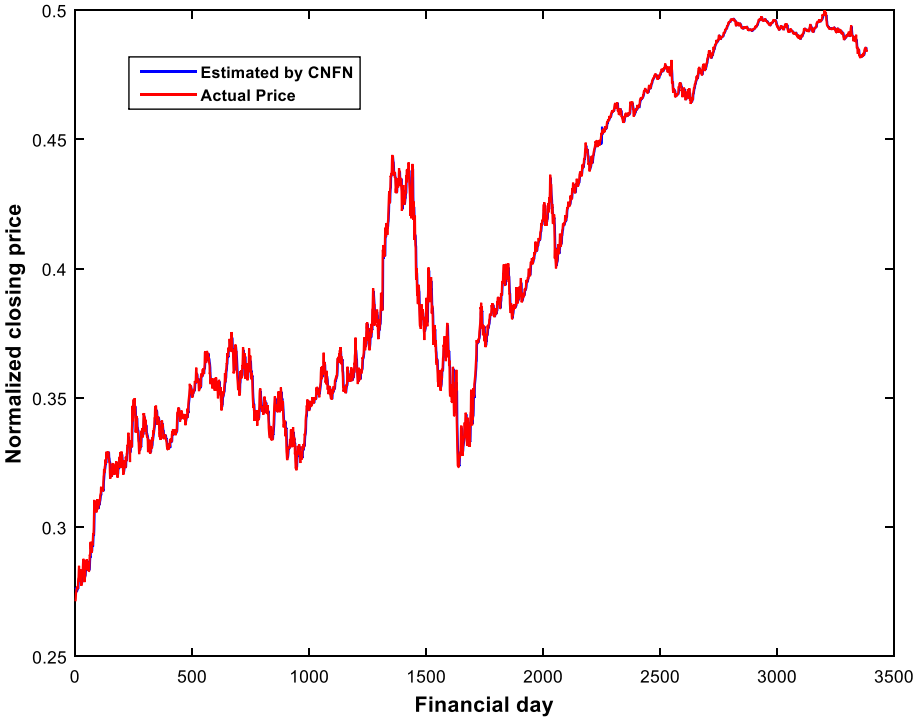

Fig. 16 Estimated v/s actual closing prices by CNFN forecasting model for BSE stock market data

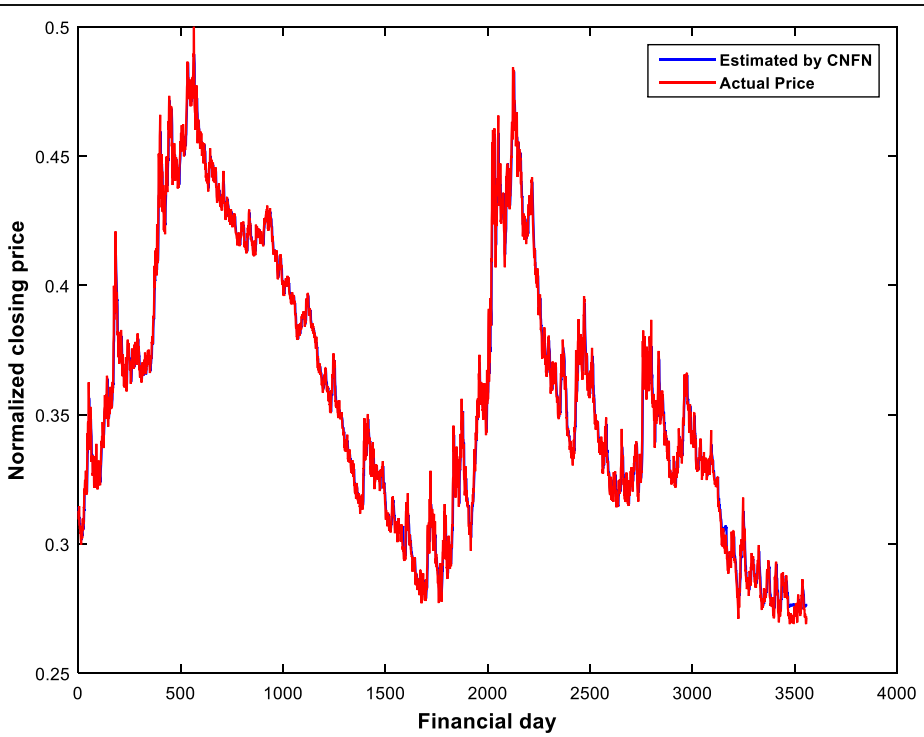

Fig. 17 Estimated v/s actual closing prices by CNFN forecasting model for FTSE stock market data

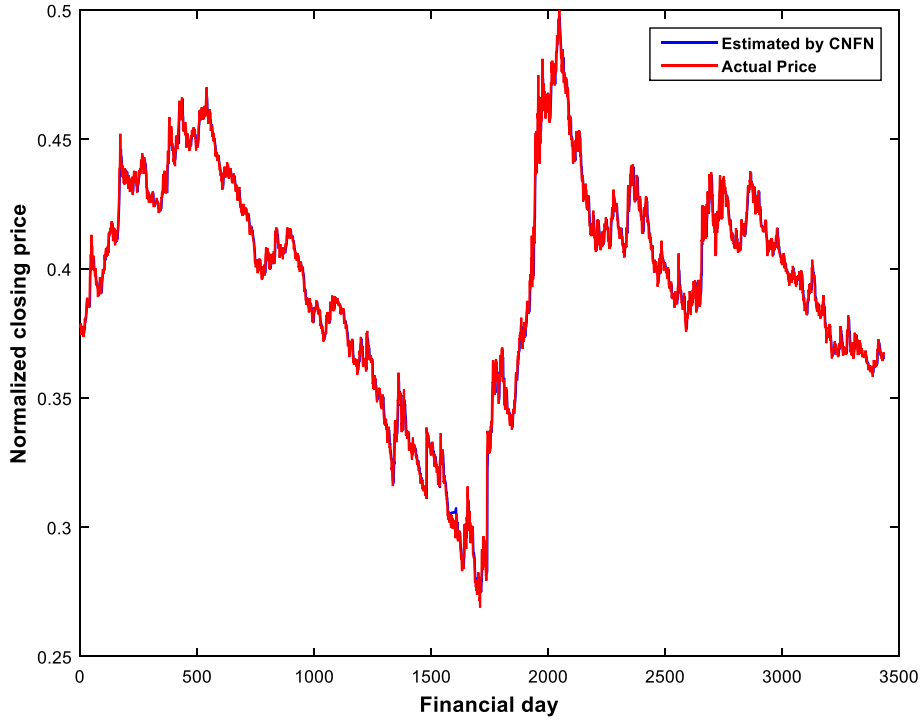

Fig. 18 Estimated v/s actual closing prices by CNFN forecasting model for TAIEX stock market data

significance level of the test as $\propto = 0.05$. Since this is a two-tailed test, the lower critical z-value corresponds to − 0.025 is − 1.96, and the upper critical z-value corresponds to 0.975 is 1.96. If the computed DM statistic falls outside the range of − 1.96 to 1.96 the null hypothesis of no difference will be rejected. The computed DM statistics are summarized in the Table 9. It can be observed that the computed DM statistic values lie outside the critical range, which supports rejection of the null hypothesis.

From the above table, it can be seen that the DM statistics obtained always lie outside of the critical range. Hence, the null hypothesis of no difference between the CNFN and the other models is rejected.

Conclusions

This paper proposed an intelligent chemical-reaction-optimization-based neuro-fuzzy network (CNFN) model to capture the high market volatility, nonlinearity, complex dynamism, and time-varying nature of stock market data. To increase the dimensionality of the input pattern space for better generalization, the input signals to the model are fuzzified. Different fuzzification methods are tested and the Gaussian membership

Table 9 Computed DM statistic values (one-day-ahead forecasting)

Stock Data		RBFNN	MLP-CRO	ANFIS	MLP-BP
BSE	CNFN	1.9647	−2.2055	2.3300	2.7545
DJIA		2.2031	3.0505	−4.1585	−5.3562
NASDAQ		−2.4165	2.1653	2.1006	3.6581
FTSE		2.0265	1.9772	2.0515	−3.2835
TAIEX		2.2274	−2.2565	2.5627	−4.5468

function is found to be better. The Gaussian membership function enables smoother transitions between members and non-members in comparison to triangular and trapezoidal membership functions, and it has fewer parameters than the bell membership function. Each input pattern generated after fuzzification is associated with a degree of membership to different classes. The optimal search space of this model is explored through CRO, which requires fewer tunable parameters. The proposed CNFN model has been employed for short-term and long-term predictions of closing prices for five real stock indices over a period of 13 years and 8 months. The model is adaptive in nature and uses the least number of input closing prices, which reduces computation time. The underlying motivations for using CRO in this study are to overcome the issues of convergence, parameter setting, and over fitting as well as to accurately forecast financial time series data even when the underlying system processes are typically nonlinear. Five performance metrics were used to evaluate the performance of the model. The performance of the model was also compared to that of four other models: the MLP-CRO, MLP-BP, ANFIS, and RBFNN models, and was found to be significantly better. The Deibold-Mariano test also established the superiority of the proposed model. The use of MLP as base model in CNFN may increase the computation and is a limitation of this model. The future research may include the use of other fuzzy membership methods and testing the applicability of the proposed model in other domains.

Abbreviations

ACO: Ant Colony Optimization; ANFIS: Adaptive Network based Fuzzy Inference System; ANN: Artificial Neural Network; ARV: Average Relative Variance; BPNN: Back Propagation Neural Network; BSE: Bombay Stock Exchange; CNFN: CRO based Neuro-Fuzzy Network; CRO: Chemical Reaction Optimization; DE: Differential Evolution; DJIA: Dow Jones Industrial Average; FL: Fuzzy Logic; GA: Genetic Algorithm; GD: Gradient Descent; GSA: Gravitational Search Algorithm; HS: Harmony Search; KE: Kinetic Energy; MA: Moving Average; MAPE: Mean Absolute Percentage of Error; MDAPE: Median Absolute Percentage Error; MLP: Multilayer Perceptron; MLP-BP: Backpropagation based MLP; MLP-CRO: CRO based Multilayer Perceptron; NMSE: Normalized Mean Squared Error; PE: Potential Energy; PSNN: Pi-Signa Neural Network; PSO: Particle Swarm Optimization; RBFNN: Radial Basis Functional Neural Network; TEPIX: Tehran Stock Exchange Index; TSK: Takagi-Sugeno-Kang

Acknowledgements

The authors are grateful to the editor-in-chief and the anonymous reviewers for their valuable suggestions which helped in improving the quality of this paper. The authors are also thankful to the Southwestern University of Finance and Economics for providing open access to our article. The first author would like to thank to the management, CMR College of Engineering & Technology, Hyderabad, India, for their continuous encouragement and support.

Authors' contributions

SCN (first author) designed the forecasting model, analyzed and interpreted data, conducted experiments, discussed the results and wrote the article. BBM (second author) explored the research area and was a major contributor in writing the manuscript. All authors read and approved the final manuscript.

Funding

Not applicable, No funding available.

Availability of data and materials

The datasets analyzed and experimented during the current study are available at https://in.finance.yahoo.com/, which openly available. The source of datasets is highlighted in subsection 5.1.

Competing interests

The authors declare that they have no competing interests.

Author details

[1]Department of Computer Science and Engineering, CMR College of Engineering & Technology, Hyderabad 501401, India. [2]Department of Information Technology, Silicon Institute of Technology, Bhubaneswar 751024, India.

Received: 26 December 2018 Accepted: 4 September 2019
Published online: 01 November 2019

References

Abbasi E, Abouec A (2008) Stock price forecast by using neuro-fuzzy inference system. Proceedings of World Academy of Science. Eng Technol 36:320–323
Abraham A, Nath B, Mahanti PK (2001) Hybrid intelligent systems for stock market analysis. In: International Conference on Computational Science. Springer, Berlin, pp 337–345
Addo P, Guegan D, Hassani B (2018) Credit risk analysis using machine and deep learning models. Risks 6(2):38
Adhikari R, Agrawal RK (2014) A combination of artificial neural network and random walk models for financial time series forecasting. Neural Comput Appl 24(6):1441–1449
Alalaya MM, Al Rawashdeh HA, Alkhateb A (2018) Combination Method between Fuzzy Logic and Neural Network Models to Predict Amman Stock Exchange. Open J Bus Manag 6(03):632
Alatas B (2011) ACROA: artificial chemical reaction optimization algorithm for global optimization. Expert Syst Appl 38(10): 13,170–13,180
Alatas B (2012) A novel chemistry based metaheuristic optimization method for mining of classification rules. Expert Syst Appl 39(12):11,080–11,088
Aminian F, Suarez ED, Aminian M, Walz DT (2006) Forecasting economic data with neural networks. Comput Econ 28(1):71–88
Atsalakis GS, Valavanis KP (2009) Forecasting stock market short-term trends using a neuro-fuzzy based methodology. Expert Syst Appl 36(3):10,696–10,707
Blinova TO (2007) Analysis of possibility of using neural network to forecast passenger traffic flows in Russia. Aviation 11(1):28–34
Board, F. S. (2017). Artificial intelligence and machine learning in financial services. November, available at: http://www.fsb.org/2017/11/artificialintelligence-and-machine-learning-in-financialservice/(Accessed 30 Jan 2018).
Boyacioglu MA, Avci D (2010) An Adaptive Network-Based Fuzzy Inference System (ANFIS) for the prediction of stock market return: The case of Istanbul Stock Exchange. Expert Syst Appl 37:7908–7912
Calderon TG, Cheh JJ (2002) A roadmap for future neural networks research in auditing and risk assessment. Int J Account Inf Syst 3(4):203–236
Castellano G, Castiello C, Fanelli AM, Jain L (2007) Evolutionary neuro-fuzzy systems and applications. Advances in evolutionary computing for system design, studies in computational intelligence, vol 66. Springer, Verlag, pp 11–45
Chandra DK, Ravi V, Bose I (2009) Failure prediction of dotcom companies using hybrid intelligent techniques. Expert Syst Appl 36(3):4830–4837
Chow JC (2018) Analysis of Financial Credit Risk Using Machine Learning. arXiv preprint arXiv 1802:05326
Darbellay GA, Slama M (2000) Forecasting the short-term demand for electricity: Do neural networks stand a better chance? Int J Forecasting 16(1):71–83
Daubie M, Meskens N (2002) Business failure prediction: a review and analysis of the literature. In: New trends in banking management. Physica, Heidelberg, pp 71–86
Diebold FX, Mariano RS (2002) Comparing predictive accuracy. J Bus Econ Stat 20(1):134–144
Dubois D, Prade H (1980) Fuzzy sets and systems: theory and applications. Academic Press, New York, pp 255–348
Ecer F (2013) Comparing the bank failure prediction performance of neural networks and support vector machines: The Turkish case. Economic Research-Ekonomska istraživanja 26(3):81–98
Enke D, Thawornwong S (2005) The use of data mining and neural networks for forecasting stock market returns. Expert Syst Appl 29:927–940
Esfahanipour A, Aghamiri W (2010) Adapted Neuro-Fuzzy Inference System on indirect approach TSK fuzzy rule base for stock analysis, Expert Systems with Applications, vol 37, pp 4742–4748
Fouladvand S, Salavati S, Masajedi P, Ghanbarzadeh A (2015) A modified neuro-evolutionary algorithm for mobile robot navigation: Using fuzzy systems and combination of artificial neural networks. Int J Knowl Based Intell Eng Syst 19(2):125–133
Ghosh A, Shankar BU, Meher SK (2009) A novel approach to Neuro-fuzzy classification. Neural Network 22(1):100–109
Gu S, Kelly B, Xiu D (2018) Empirical asset pricing via machine learning (No. w25398). National Bureau of Economic Research
Guan H, Dai Z, Zhao A, He J (2018) A novel stock forecasting model based on High-order-fuzzy-fluctuation Trends and Back Propagation Neural Network. PloS one 13(2):e0192366
Harvey D, Leybourne S, Newbold P (1997) Testing the equality of prediction mean squared errors. Int J Forecasting 13(2):281–291
Hsu MW, Lessmann S, Sung MC, Ma T, Johnson JE (2016) Bridging the divide in financial market forecasting: machine learners vs. financial economists. Expert Syst Appl 61:215–234
James JQ, Lam AY, Li VO (2011) Evolutionary artificial neural network based on chemical reaction optimization. In: 2011 IEEE Congress of Evolutionary Computation (CEC), pp 2083–2090 IEEE
Keles_ A, Keles_ A (2013) Extracting fuzzy rules for diagnosis of breast cancer. Turkish J Electrical Eng Comput Sci 21(1):1495–1503
Kotha KK, Sahu B (2016) Macroeconomic factors and the Indian stock market: Exploring long and short run relationships. Int J Econ Financ Issues 6(3):1081–1091
Kou G, Chao X, Peng Y, Alsaadi FE, Herrera-Viedma E (2019) Machine learning methods for systemic risk analysis in financial sectors. Technol Econ Dev Economy:1–27
Kou G, Peng Y, Wang G (2014) Evaluation of clustering algorithms for financial risk analysis using MCDM methods. Inf Sci 275:1–12
Kuo RJ, Chen CH, Hwang YC (2001) An intelligent stock trading decision support system through integration of genetic algorithm based fuzzy neural network and artificial neural network. Fuzzy Sets Systems, vol 118, pp 21–24
Lam AY, Li VO (2010) Chemical reaction optimization for cognitive radio spectrum allocation. In: 2010 IEEE Global Telecommunications Conference GLOBECOM 2010, pp 1–5 IEEE
Lam AY, Li VO (2012) Chemical reaction optimization: A tutorial. Memetic Computing 4(1):3–17
Lam AY, Li VO, James JQ (2012) Real-coded chemical reaction optimization. IEEE Trans Evol Comput 16(3):339–353
Lam AY, Xu J, Li VO (2010) Chemical reaction optimization for population transition in peer-to-peer live streaming. In: IEEE Congress on Evolutionary Computation, pp 1–8 IEEE
Li G, Kou G, Peng Y (2016) A group decision making model for integrating heterogeneous information. IEEE Transact Syst Man Cybern Syst 48(6):982–992

Liu B (2004) Uncertainty theory: an introduction to its axiomatic foundations. Springer, Berlin, pp 191–346
Mostafa MM (2004) Forecasting the Suez Canal traffic: a neural network analysis. Marit Policy Manag 31(2):139–156
Mostafa MM (2010) Forecasting stock exchange movements using neural networks: Empirical evidence from Kuwait. Expert Syst Appl 37(9):6302–6309
Najafzadeh M, Barani GA, Hessami-Kermani MR (2015) Evaluation of GMDH networks for prediction of local scour depth at bridge abutments in coarse sediments with thinly armored beds. Ocean Eng 104:387–396
Najafzadeh M, Bonakdari H (2016) Application of a neuro-fuzzy GMDH model for predicting the velocity at limit of deposition in storm sewers. Journal of Pipeline Systems Engineering and. Practice 8(1):06016003
Najafzadeh M, Saberi-Movahed F (2018) GMDH-GEP to predict free span expansion rates below pipelines under waves. Mar Georesources Geotechnol:1–18
Najafzadeh M, Saberi-Movahed F, Sarkamaryan S (2018) NF-GMDH-Based self-organized systems to predict bridge pier scour depth under debris flow effects. Mar Georesources Geotechnol 36(5):589–602
Nayak J, Naik B, Behera HS (2015) A novel chemical reaction optimization based higher order neural network (CRO-HONN) for nonlinear classification. Ain Shams Eng J 6(3):1069–1091
Nayak SC, Misra BB, Behera HS (2012) Stock index prediction with neuro-genetic hybrid techniques. Int J Comput Sci Inform 2:27–34
Nayak SC, Misra BB, Behera HS (2013) Hybridizing chemical reaction optimization and artificial neural network for stock future index forecasting. In: 2013 1st International Conference on Emerging Trends and Applications in Computer Science, pp 130–134 IEEE
Nayak SC, Misra BB, Behera HS (2014) Impact of data normalization on stock index forecasting. Int J Comp Inf Syst Ind Manag Appl 6:357–369
Nayak SC, Misra BB, Behera HS (2016) Efficient forecasting of financial time-series data with virtual adaptive neuro-fuzzy inference system. Int J Bus Forecasting Mark Intell 2(4):379–402
Nayak SC, Misra BB, Behera HS (2017a) Artificial chemical reaction optimization of neural networks for efficient prediction of stock market indices. Ain Shams Eng J 8(3):371–390
Nayak SC, Misra BB, Behera HS (2017b) Artificial chemical reaction optimization based neural net for virtual data position exploration for efficient financial time series forecasting. Ain Shams Eng J
Nayak SC, Misra BB, Behera HS (2018) ACFLN: artificial chemical functional link network for prediction of stock market index. Evolving Systems:1–26
Niaki STA, Hoseinzade S (2013) Forecasting S&P 500 index using artificial neural networks and design of experiments. J Ind Eng Int 9(1):1
Pan B, Lam AY, Li VO (2011) Network coding optimization based on chemical reaction optimization. In: 2011 IEEE Global Telecommunications Conference-GLOBECOM 2011, pp 1–5 IEEE
Quek C (2005) Predicting the impact of anticipator action on US stock market – An event study using ANFIS (a neural fuzzy model). Comput Intell 23:117–141
Rahman, P. A., Panchenko, A. A., & Safarov, A. M. (2017). Using neural networks for prediction of air pollution index in industrial city. In IOP Conference Series: Earth and Environmental Science (Vol. 87, No. 4, p. 042016). IOP Publishing.
Romahi Y, Shen Q (2000) Dynamic financial forecasting with automatically induced fuzzy associations. In: Ninth IEEE International Conference on Fuzzy Systems. FUZZ-IEEE 2000, vol 1, pp 493–498 (Cat. No. 00CH37063). IEEE
Shaverdi M, Fallahi S, Bashiri V (2012) Prediction of Stock Price of Iranian Petrochemical Industry using GMDH-Type Neural Network and Genetic Algorithm. Appl Math Sci 6(7):319–332
Singh VK, Kumar P, Singh BP, Malik A (2016) A comparative study of adaptive neuro fuzzy inference system (ANFIS) and multiple linear regression (MLR) for rainfall-runoff modelling. Int J Sci Nat 7(4):714–723
Tomczak JM, Zięba M (2015) Classification restricted Boltzmann machine for comprehensible credit scoring model. Expert Syst Appl 42(4):1789–1796
Truong TK, Li K, Xu Y (2013) Chemical reaction optimization with greedy strategy for the 0–1 knapsack problem. Applied Soft Comput 13(4):1774–1780
Turchenko, V., Beraldi, P., De Simone, F., & Grandinetti, L. (2011). Short-term stock price prediction using MLP in moving simulation mode. In Proceedings of the 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems (Vol. 2, pp. 666–671). IEEE.
Ture M, Kurt I (2006) Comparison of four different time series methods to forecast hepatitis A virus infection. Expert Syst Appl 31:41–46
Xu J, Lam AY, Li VO (2010) Chemical reaction optimization for the grid scheduling problem. In: 2010 IEEE International Conference on Communications, pp 1–5 IEEE
Xu J, Lam AY, Li VO (2011a) Stock portfolio selection using chemical reaction optimization. In: Proceedings of International Conference on Operations Research and Financial Engineering (ICORFE 2011), pp 458–463
Xu J, Lam AY, Li VO (2011b) Chemical reaction optimization for task scheduling in grid computing. IEEE Transact Parallel Distributed Syst 22(10):1624–1631
Yu L, Wang S, Lai KK (2009) A neural-network-based nonlinear metamodeling approach to financial time series forecasting. Appl Soft Comput 9(2):563–574
Yunos ZM, Shamsuddin SM, Sallehuddin R (2008) Data Modeling for Kuala Lumpur Composite Index with ANFIS. In: Second Asia international conference on modeling and simulation, AICMS 08, Kuala Lumpur, pp 609–614
Zadeh L (1965) A. Fuzzy sets. Inf Control 8(3):338–353
Zhang GP (2003) Time series forecasting using a hybrid ARIMA and neural network model. Neurocomputing 50(2003):159–175
Zhang H, Kou G, Peng Y (2019) Soft consensus cost models for group decision making and economic interpretations. Eur J Oper Res 277(3):964-980
Zhong X, Enke D (2017) Forecasting daily stock market return using dimensionality reduction. Expert Syst Appl 67:126–139
Zhuo W, Li-Min J, Yong Q, Yan-Hui W (2007) Railway passenger traffic volume prediction based on neural network. Appl Artif Intell 21(1):1–10

Publisher's Note