ISSN: 2321-7782 (Online)

Volume 3, Issue 2, February 2015

International Journal of Advance Research in Computer Science and Management Studies

Research Article / Survey Paper / Case Study

Available online at: www.ijarcsms.com

Optimizing LS-SVM using Modified Cuckoo Search Algorithm (MCS) for Stock Price Prediction

Osman Hegazy[1]
Faculty of Computers and Informatics
Cairo University
Egypt

Omar S. Soliman[2]
Faculty of Computers and Informatics
Cairo University
Egypt

Mustafa Abdul Salam[3]
Higher Technological Institute (H.T.I)
10th of Ramadan City
Egypt

Abstract: In this Paper, Modified Cuckoo Search algorithm (MCS), which is improved version of Cuckoo Search (CS) algorithm, has been used. MCS algorithm modifies CS algorithm which is inspired from the reproduction strategy of cuckoo birds. MCS algorithm exchange information between the top eggs, or best solutions which not found in standard CS algorithm. This modification ensures convergence to global minimum. MCS algorithm is proposed to optimize least square support vector machine (LS-SVM) model to be used in daily stock price predition. MCS is proposed to select best free parameters combination for LS-SVM. Six financial technical indicators derived from stock historical data were used as inputs to proposed model. Standard LS-SVM and ANN trained with scaled conjugate gradient algorithm (SCG) were used as benchmarks for comparison with proposed model. Proposed model tested with fifteen datasets representing different sectors in S&P 500 stock market. Results presented in this paper showed that the proposed MCS-LS-SVM model has a fast convergence speed. It achieved better accuracy than compared algorithm. It also overcame overfitting and local minima problems found in ANN and standard LS-SVM especially in fluctuated datasets.

Keywords: Modified cuckoo search; least square support vector machine; scaled conjugate gradient; financial technical indicators; and stock price prediction.

I. Introduction

Mining data streams has been at focus since last few years. It concerned with discovering valuable and hidden information and knowledge from continuous streams of data. The research in the area of data stream mining has grown due to the significance of its applications. Applications of data stream analysis can vary from critical scientific and astronomical applications to important business and financial ones [1]. Stock market data is considered one of the most commonly data streams.

Financial technical indicators play an important role in field of stock market. These were from the first methods used to forecast stock market trend and price. The indicators are classified in two classes, oscillators or leading indicators, and lagging indicators [2]. Leading indicators are designed to lead price movements. The lagging indicators follow the price action and are referred to as trend-following indicators.

Artificial Neural Network (ANN) is considered one of the most commonly machine learning techniques used in stock market prediction. In most cases ANNs suffer from over-fitting problem due to the large number of parameters to fix, and the little prior user knowledge about the relevance of the inputs in the analyzed problem [3].

Support vector machines (SVMs) have been developed as an alternative that avoids ANN limitations. SVM computes globally optimal solutions, unlike those obtained with ANN, which tend to fall into local minima [4]. Least squares support vector machine (LS-SVM) method which is presented in [5], is a reformulation of the traditional SVM algorithm. Although LS-SVM simplifies the SVM procedure, the regularization parameter and the kernel parameters play an important role in the regression system. Therefore, it is necessary to establish a methodology for properly selecting the LS-SVM free parameters. The perceived advantages of evolutionary strategies as optimization methods motivated the authors to consider such stochastic methods in the context of optimizing SVM. A survey and overview of evolutionary algorithms (EAs) is found in [6].

In 2009, Yang and Deb proposed Cuckoo Search (CS) Algorithm [7], which is a nature-inspired metaheuristic algorithm for continuous optimization. CS is based on the brood parasitism of some cuckoo species. CS is enhanced by the Levy flights [8], rather than by simple isotropic random walks. CS algorithm was applied to engineering design applications; it has superior performance over other algorithms for a range of continuous optimization problems such as spring design and welded beam design problems [9, 10, and 11]. Vazquez [12] used cuckoo search to train spiking neural network models. Chifu et al. [13] optimized semantic web service composition processes using cuckoo search. Kumar and Chakarverty [14] achieved optimal design for reliable embedded system using cuckoo search. Kaveh and Bakhshpoori [15] used CS to successfully design steel frames. Yildiz [16] has used CS to select optimal machine parameters in milling operation with enhanced results. Zheng and Zhou [17] provided a variant of cuckoo search using Gaussian process.

In 2011 Walton proposed Modified Cuckoo Search (MCS) algorithm [18]. MCS improved standard CS algorithm especially in terms of convergence to global minimum in real world applications.

This paper proposes a hybrid MCS-LS-SVM model which combines MCS algorithm, financial technical indicators, and LS-SVM model in one framework. The performance of LS-SVM is based on the selection of hyper parameters C (cost penalty), ⍰ (insensitive-loss function) and γ (kernel parameter). MCS will be used to find the best parameter combination for LS-SVM.

The rest of paper is organized as follows: Section II presents the Modified Cuckoo Search (MCS) algorithm; Section III presents the Least square support vector machine (LS-SVM) model; Section IV is devoted for the proposed model and its implementation in daily stock price and trend prediction; In Section V the results are discussed. The main conclusions of the work are presented in Section VI.

II. MODIFIED CUCKOO SEARCH ALGORITHM (MCS)

Researchers after enough computations proved that Cuckoo search (CS) algorithm is always find the optimum [19] but, as the search relies entirely on random walks, a fast convergence cannot be guaranteed. Modified Cuckoo search algorithm (MCS) made two modifications to the original CS with the aim of increasing the convergence rate. These modifications make the CS more practical for a wider range of applications but without losing the attractive features of the original method [18].

The first modification is made to the size of the Lévy flight step sizeα. In CS, α is constant and the value $\alpha = 1$ is employed [7]. In the MCS, the value of α decreases as the number of generations increases. This is done for the same reasons that the inertia constant is reduced in the PSO [20], i.e. to encourage more localized searching as the individuals, or the eggs, get closer to the solution. An initial value of the Lévy flight step size $A = 1$ is chosen and, at each generation, a new Lévy flight step is calculated using $\alpha = A/\sqrt{G}$, where G is the generation number. This exploratory search is only performed on the fraction of nests to be abandoned.

The second modification is to add information exchange between the eggs in an attempt to speed up convergence to a minimum. In the CS, there is no information exchange between individuals and, essentially, the searches are performed independently. In the MCS, a fraction of the eggs with the best fitness are put into a group of top eggs. For each of the top eggs, a second egg in this group is picked at random and a new egg is then generated on the line connecting these two top eggs. The distance along this line at which the new egg is located is calculated, using the inverse of the golden ratio $\varphi = (1 + \sqrt{5})/2$, such that it is closer to the egg with the best fitness. In the case that both eggs have the same fitness, the new egg is generated at the midpoint. Whilst developing the method a random fraction was used in place of the golden ratio, it was found that the golden ratio showed significantly greater performance than a random fraction. There is a possibility that, in this step, the same egg is picked twice. In this case, a local Lévy flight search is performed from the randomly picked nest with step size $\alpha = A/G^2$. The steps involved in the modified cuckoo search are shown in detail in Algorithm 1. There are two parameters, the fraction of nests to be abandoned and the fraction of nests to make up the top nests, which need to be adjusted in the MCS. Through testing on benchmark problems, it was found that setting the fraction of nests to be abandoned to 0.75 and the fraction of nests placed in the top nests group to 0.25 yielded the best results over a variety of functions.

Algorithm 1. Modified Cuckoo Search (MCS) [18]

```
$A \leftarrow MaxLévyStepSize$
$\varphi \leftarrow GoldenRatio$
Initialize a population of $n$ nests $x_i$ $(i = 1,2,\ldots,n)$
for all $x_i$ do
    Calculate fitness $F_i = f(x_i)$
end for
Generation number $G \leftarrow 1$
while $NumberObjectiveEvaluations$
  $< MaxNumberEvaluations$ do
  $G \leftarrow G + 1$
Sort nests by order of fitness
for all nests to be abandoned do
    Current position $x_i$
    Calculate Lévy flight step size $\alpha \leftarrow A/\sqrt{G}$
    Perform Lévy flight from $x_i$ to generate new egg $x_k$
    $x_i \leftarrow x_k$
   $F_i \leftarrow f(x_i)$
end for
for all of the top nests do
   Current position $x_i$
   Pick another nest from the top nests at random $x_i$
   if $x_i = x_j$ then
      Calculate Lévy flight step size $\alpha \leftarrow A/G^2$
      Perform Lévy flight from $x_i$ to generate new egg $x_k$
      $F_k = f(x_k)$
     Choose a random nest $l$ from all nests
     if $(F_k > F_l)$ then
      $x_l \leftarrow x_k$
     $F_l \leftarrow f_k$
     end if
else
   $dx = |x_i - x_j|/\varphi$
   Move distance $dx$ from the worst nest to the
   best nest to find $x_k$
  $F_k = f(x_k)$
  Choose a random nest l from all nests
     if $(F_k > F_l)$ then
      $x_l \leftarrow x_k$
     $F_l \leftarrow f_k$
    end if
  end if
end for
end while
```

III. Least Square Support Vector Machine (LSs-SVM)

Least squares support vector machines (LS-SVMs) are least squares versions of support vector machines (SVMs), which are a set of related supervised learning methods that analyze data and recognize patterns, and which are used for classification and regression analysis. In this version one can find the solution by solving a set of linear equations instead of a convex quadratic programming (QP) problem for classical SVMs. LS-SVMs classifiers, were proposed by Suykens and Vandewalle [21].LS-SVM is described as follows.

Let X is $n\times p$ input data matrix and y is $n\times 1$ output vector. Given the $\{x_i, y_i\}_{i=1}^{n}$ training data set, where $x_i \in R^p$ and $y_i \in R$, the LS-SVM goal is to construct the function $f(x)=y$, which represents the dependence of the output y_i on the input x_i . This function is formulated as

$$f(x) = W^T\varphi(x) + b \qquad (1)$$

Where W and $\varphi(x)$: $R^p \rightarrow R^n$ are $n\times 1$ column vectors, and $b \in R$. LS-SVM algorithm [22] computes the function (1) from a similar minimization problem found in the SVM method [4]. However the main difference is that LS-SVM involves equality constraints instead of inequalities, and it is based on a least square cost function. Furthermore, the LS-SVM method solves a linear problem while conventional SVM solves a quadratic one. The optimization problem and the equality constraints of LS-SVM are defined as follows:

$$\min_{w,e,b} j(w,e,b)=\frac{1}{2}w^T w+C\frac{1}{2}e^T \qquad (2)$$

$$y_i = w^T\varphi(x_i) + b + e_i \qquad (3)$$

Where e is the $n\times 1$ error vector, 1 is a $n\times 1$ vector with all entries 1, and $C \in R^{+}$ is the tradeoff parameter between the solution size and training errors. From (2) a Lagrangian is formed, and differentiating with respect to w,b,e,a (a is Lagrangian multipliers), we obtain

$$\begin{bmatrix} I & 0 & 0 & -Z^T \\ 0 & 0 & 0 & -1^T \\ 0 & 0 & CI & -I \\ Z & 1 & I & 0 \end{bmatrix} \begin{bmatrix} W \\ b \\ e \\ a \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ y \end{bmatrix} \qquad (4)$$

Where I represents the identity matrix and $Z = [\varphi(x1), \varphi(x2),..., \varphi(x_n)]^T$.

From equation (4) $w = Z^T a$ and $Ce = a$.

Then, by defining the kernel matrix $K = ZZ^T$, and the parameter $\lambda = C^{-1}$, the conditions for optimality lead to the following overall solution

$$\begin{bmatrix} 0 & 1^T \\ 1 & K+\lambda I \end{bmatrix} \begin{bmatrix} b \\ a \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix} \qquad (5)$$

Kernel function K types are as follows:

- Linear kernel :

$$K(x, x_i) = x_i^T x \tag{6}$$

- Polynomial kernel of degree d :

$$K(x, x_i) = (1 + x_i^T x / c)^d \tag{7}$$

- Radial basis function RBF kernel :

$$K(x, x_i) = \exp(-\|x - x_i\|^2 / \sigma^2) \tag{8}$$

- MLP kernel :

$$K(x, x_i) = \tanh(k x_i^T x + \theta) \tag{9}$$

IV. The Proposed Model

The proposed model is based on the study of stock historical data (High, Low, Open, Close, and Vol.). Then technical indicators are calculated from these historical data to be used as inputs to proposed model. After that LS-SVM is optimized by MCS algorithm to be used in the prediction of daily stock prices. Standard LS-SVM, and ANN trained with Scaled Conjugate gradient (SCG) algorithm, which is one of the best back-propagation derivatives, are used as benchmarks for comparison with proposed model. The proposed model architecture contains seven inputs vectors represent the historical data and six derived technical indicators from raw datasets, and one output represents next price. The proposed model phases are summarized in Fig. 1.

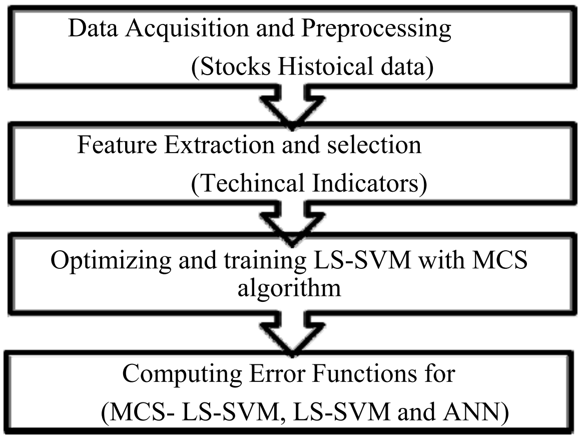
Data Acquisition and Preprocessing
(Stocks Histoical data)
Feature Extraction and selection
(Techincal Indicators)
Optimizing and training LS-SVM with MCS algorithm
Computing Error Functions for
(MCS- LS-SVM, LS-SVM and ANN)

Fig.1 the proposed model phases.

The financial technical indicators, which are calculated from the raw datasets, are as follows:

- **Price Momentum Oscillator (PMO) :**

  PMO is an oscillator based on a Rate of Change (ROC) calculation that is exponentially smoothed twice. Because the PMO is normalized, it can also be used as a relative strength tool. Stocks can thus be ranked by their PMO value as an expression of relative strength.

  TC = today's close price

  TDAC = close price ten days ago

  The following was used to calculate PMO:

$$\mathrm{PMO} = \mathrm{TC} - \mathrm{TDAC} \qquad (10)$$

- **Relative Strength Index (RSI)**:

A technical momentum indicator that compares the magnitude of recent gains to recent losses in an attempt to determine overbought and oversold conditions of an asset. The formula for computing the Relative Strength Index is as follows.

$$\mathrm{RSI} = 100 - [100 / (1 + \mathrm{RS})] \qquad (11)$$

Where RS = Avg. of x days' up closes divided by average of x days' down closes.

- **Money Flow Index (MFI):**

This one measures the strength of money in and out of a security. The formula for MFI is as follows.

$$\text{Money Flow (MF)} = \mathrm{TP} * \mathrm{V} \qquad (12)$$

Where, TP is typical price, and V is money Vol.

Money Ratio (MR) is calculated as:

$$\mathrm{MR} = (\text{Positive MF} / \text{Negative MF}) \quad (13)$$

$$\mathrm{MFI} = 100 - (100/ (1 + \mathrm{MR})) \qquad (14)$$

- **Exponential Moving Average (EMA):**

This indicator returns the exponential moving average of a field over a given period of time. EMA formula is as follows.

$$\mathrm{EMA} = [\alpha * \mathrm{T\ Close}] + [1 - \alpha * \mathrm{Y\ EMA}] \quad (15)$$

Where T is Today's close and Y is Yesterday's close.

- **Stochastic Oscillator (SO):**

The stochastic oscillator defined as a measure of the difference between the current closing price of a security and its lowest low price, relative to its highest high price for a given period of time. The formula for this computation is as follows.

$$\%K = [(CP - LP) / (HP - LP)] * 100 \qquad (16)$$

Where, CP is Close price, LP is Lowest price, HP is Highest Price, and LP is Lowest Price.

- **Moving Average Convergence/Divergence (MACD):**

This function calculates difference between a short and a long term moving average for a field. The formulas for calculating MACD and its signal are as follows.

$$\mathrm{MACD} = [0.075 * \mathrm{E}] - [0.15 * \mathrm{E}] \qquad (17)$$

Where, E is EMA (CP)

$$\text{Signal Line} = 0.2\text{*EMA of MACD} \qquad (18)$$

V. Results and Discussions

The proposed MCS-LS-SVM and compared models were trained and tested with daily datasets for twelve companies cover all sectors in S&P 500 stock market. Datasets period are from Feb. 2011 to Feb. 2014. All datasets are available in [23]. Datasets are divided into training part (70%) and testing part (30%). All results done by matlab 2012b.

MCS-LS-SVM algorithm parameters are shown in table 1.

TABLE 1
MCS-LS-SVM algorithm parameters.

No. of nests	epochs	Search
25	100	Lévy flight

The ANN parameters are found in table 2.

TABLE 2
The ANN model parameters.

Training algorithm	Input layer	Hidden layer	epochs	Output layer
SCG	7 nodes	15 nodes	1000	1 nodes

Fig. 2 outlines the ANN structure used in this paper. The ANN structure has seven nodes in input layer representing the six technical indicators and daily close price. It has also one node in output layer representing the next week close price.

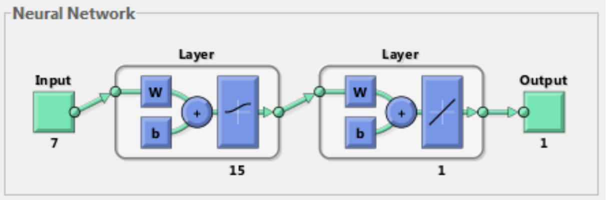

Fig. 2 ANN structure.

Table 3 outlines the performance evaluations criteria used in this paper to evaluate proposed and compared model according to error value and trend or direction prediction accuracy.

TABLE 3
Performance evaluations criteria used.

Performance criteria	Symbol	Formula
Root Mean Square Error	$RMSE$	$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(A_i - F_i)}$
Mean Absolute Error	MAE	$\frac{1}{n}\sum_{i=1}^{n}\lvert A_i - F_i\rvert$
Symmetric Mean Absolute Percentage Error	$SMAPE$	$\frac{\sum_{i=1}^{n}\lvert F_i - A_i\rvert}{\sum_{i=1}^{n} A_i + F_i}$
Percent Mean Relative Error	$PMRE$	$\frac{100}{n}\sum_{i=1}^{n}\left\lvert\frac{A_i - F_i}{F_i}\right\rvert$

Where A_i is the Actual Value, and F_i is the Forecasted Value.

Fig.s from Fig. 3 to Fig. 22 outline the application of Proposed MCS-LS-SVM model on test datasets period for different daily datasets representing different stock market sectors.

In Fig.s (3, 9, and 17) whose represent results of application proposed model to American Express, boing, and raptor companies which in. Results show that proposed MCS-LS-SVM model is achieving lowest error value since the test datasets differ from training datasets which found in Fig.s (4, 10, 18) respectively.

In Fig.s (6, and 15) whose represent two different companies AT&T, and HP, proposed model achieved little advance since datasets is normal and testing data near in change to training datasets which found in Fig.s(7, and 16) respectively.

In Fig.s (5, 8, 11,13, 14, 19, 21 and 22), whose represent results of eight companies in different sectors (Apple, Bank of America, Coca-Cola, Devon, General Motors, Toyota, Visa, and Western Digital). Results show that ANN is fallen in overfitting problem, since the datasets have fluctuations. MCS-LS-SVM algorithm is the best one with lowest error value and could easily overcome local minima and overfitting problems, while LS-SVM is better than ANN.

In Fig. 12 which represents results of Cisco Company, we can remark that test dataset is semi fluctuated, so the predicted curve using the proposed MCS-LS-SVM achieves best accuracy, followed by LS-SVM, while ANN-SCG is the worst one.

In Fig. 20, which represents result of United Health Company, one can notice that the LS-SVM has fallen in overfitting problem, while proposed model is the best one.

Tables (4, 5, 6, and 7) show RMSE, MAE, PMRE, and SMAPE performances functions for proposed model and compared algorithms for test data. Proposed MCS-LS_SVM model achieves best error value in all cases, and can easily overcome LS-SVM, and ANN problems.

Fig.s (23, 24, 25, and 26) represent test data results of RMSE, MAE, PMRE, and SMAPE performances functions.

Table 8 shows the trend (direction) prediction of test data, proposed model accuracy in all datasets near to 100%.

Tables (9, 10, 11, 12, and 13) show the four performance functions of training datasets.

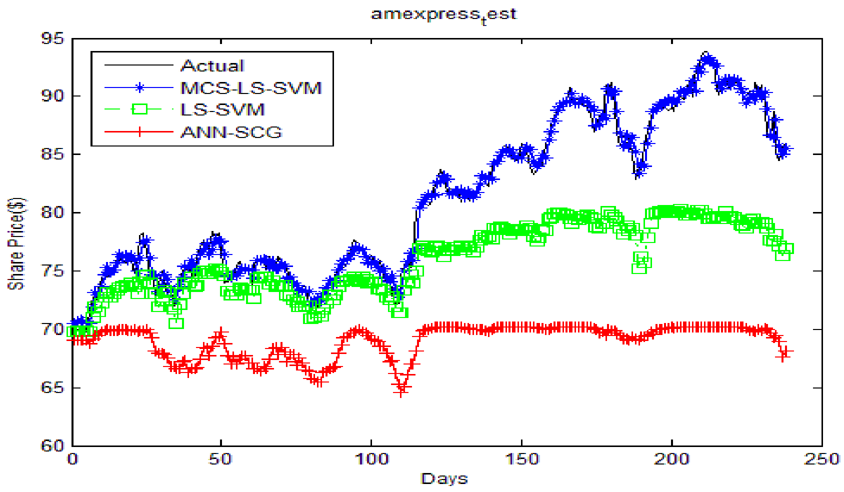

Fig. 3 Test results for American Express company

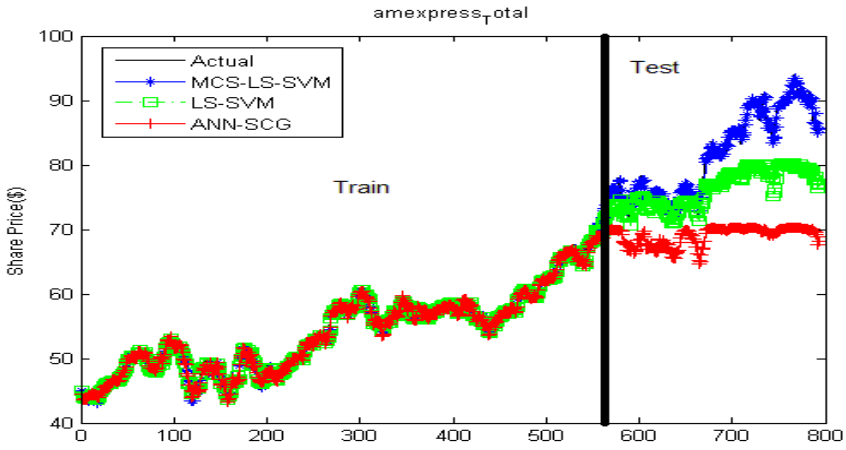

Fig. 4 Train andTest results for American Express company

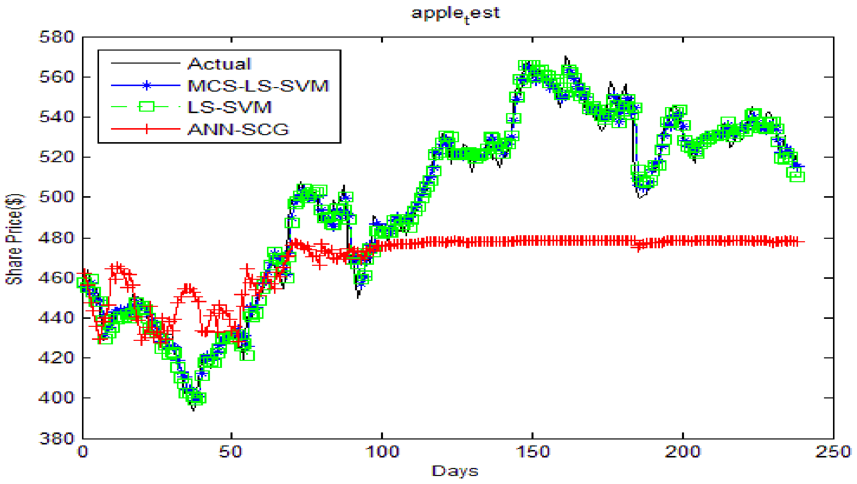

Fig. 5 Test results for Apple company

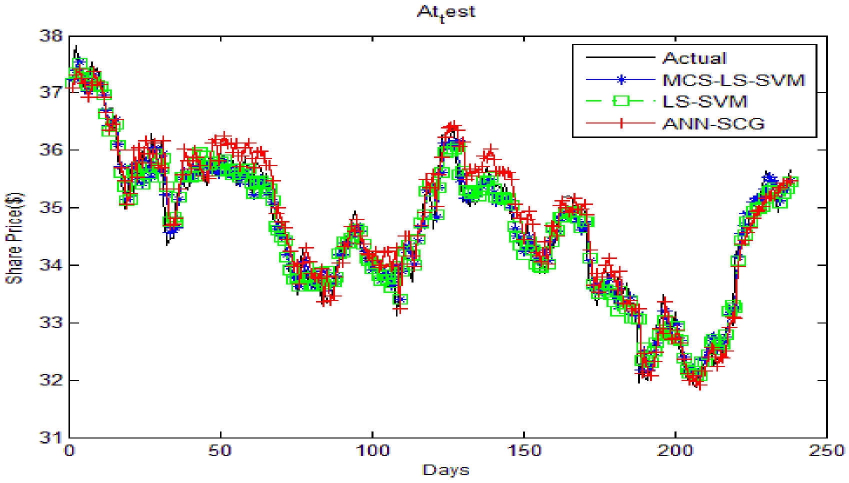

Fig. 6 Test results for AT&T company

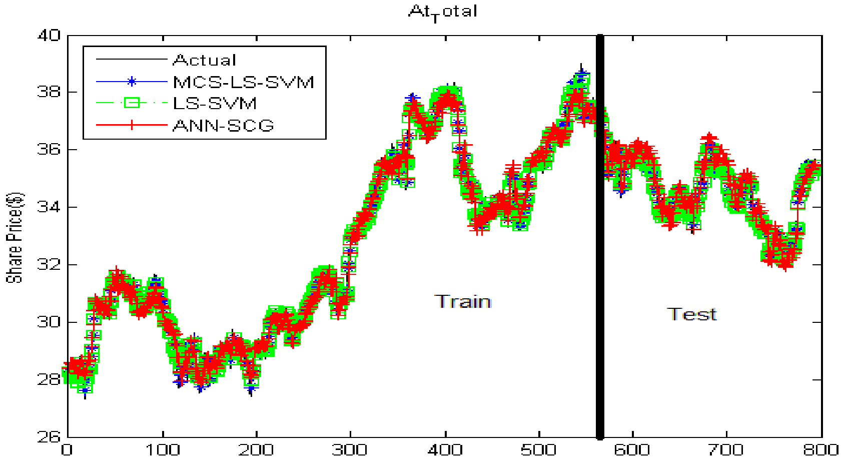

Fig. 7 Train and Test results for AT&T company

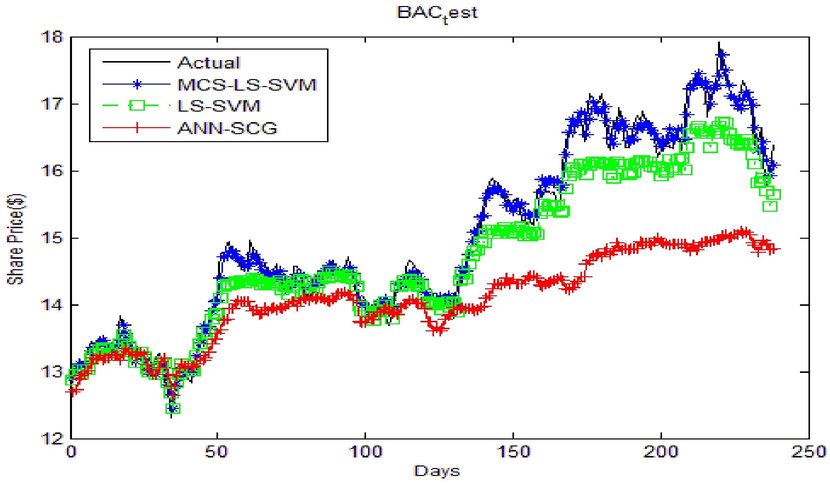

Fig. 8 Test results for Bank of America company

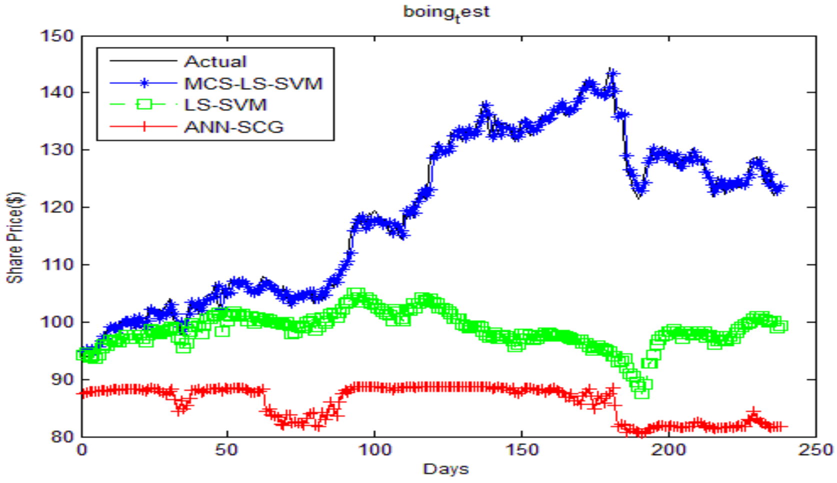

Fig. 9 Test results for Boing company

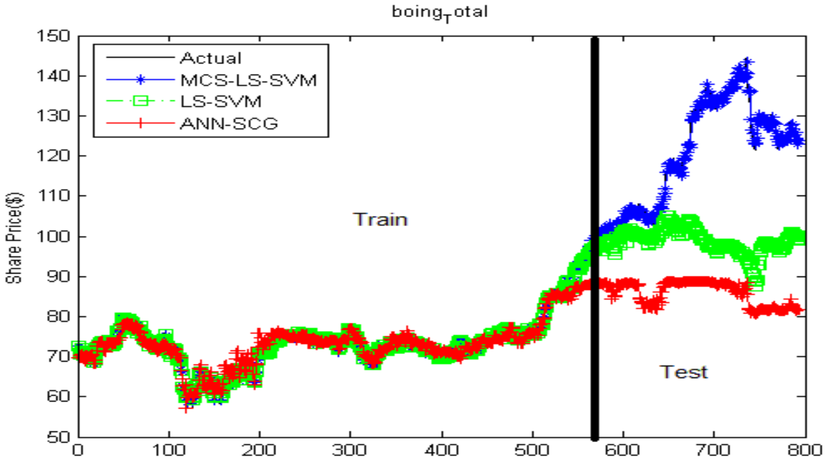

Fig. 10 Train andTest results for Boing company

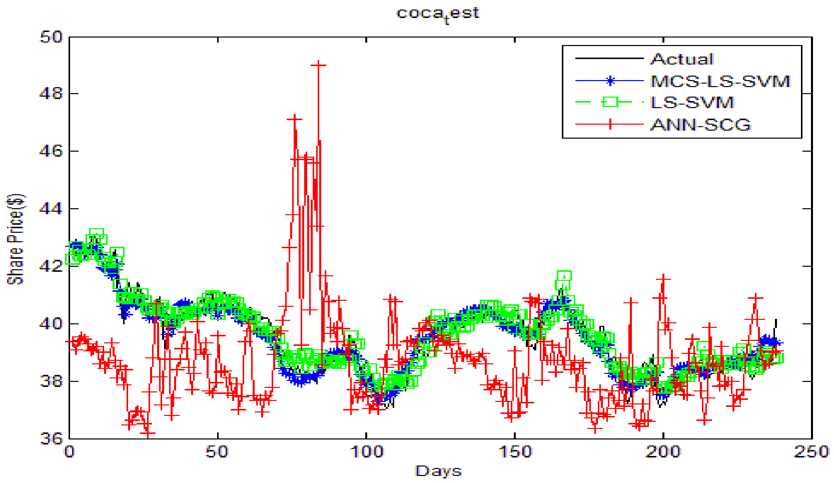

Fig. 11 Test results for Coca-Cola company

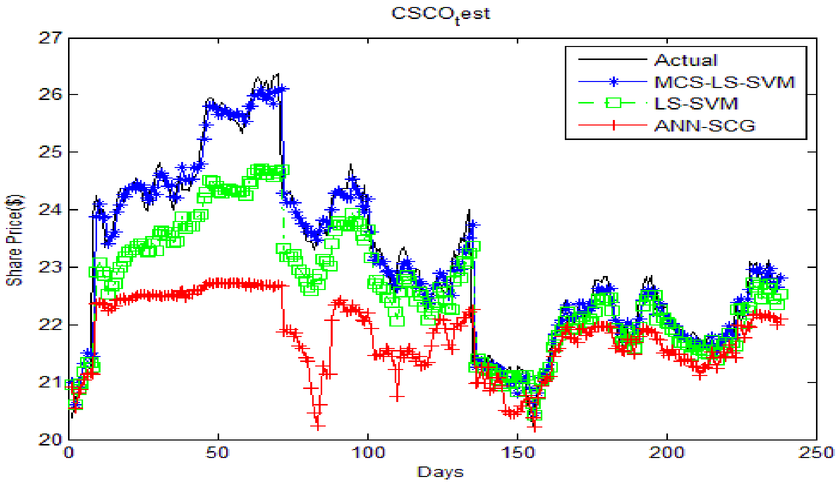

Fig. 12 Test results for CSCO company

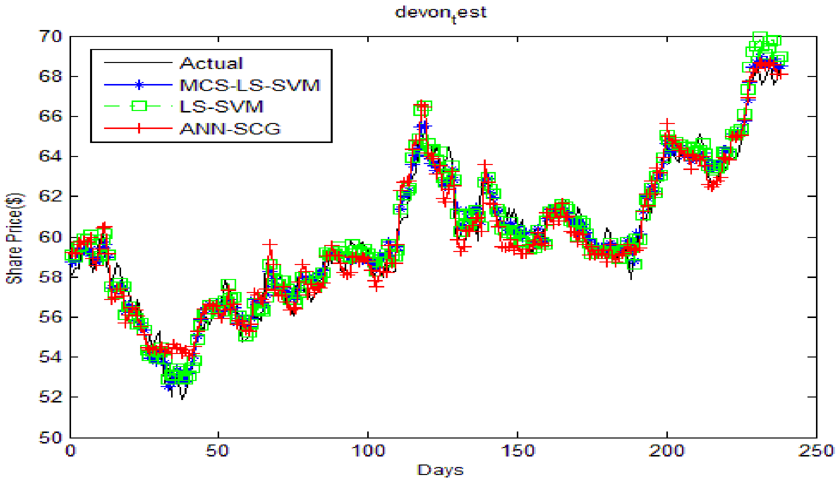

Fig. 13 Test results for Devon Energy company

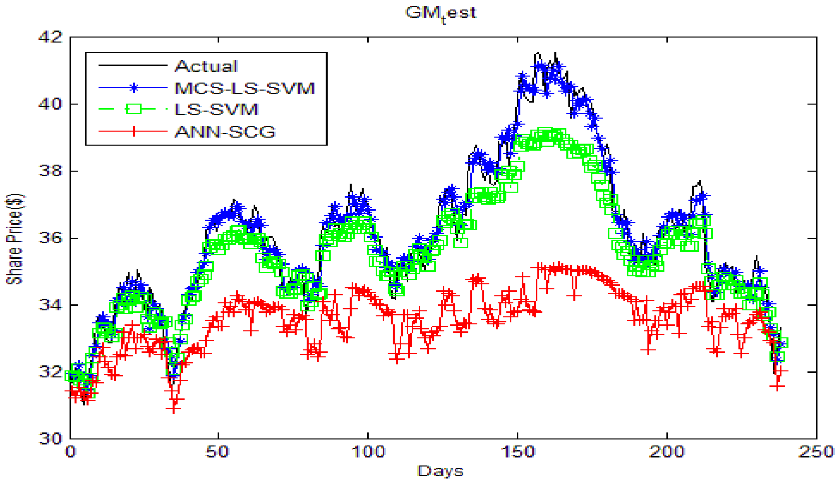

Fig. 14 Test results for General Motors company

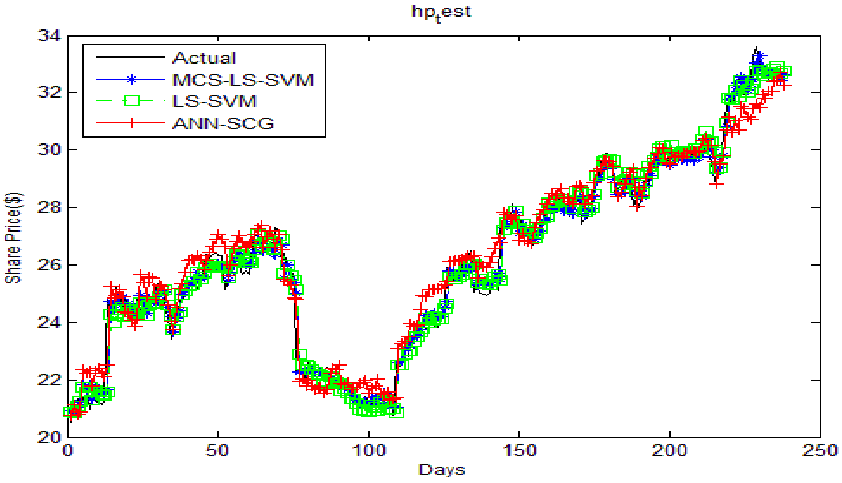

Fig. 15 Test results for HP company

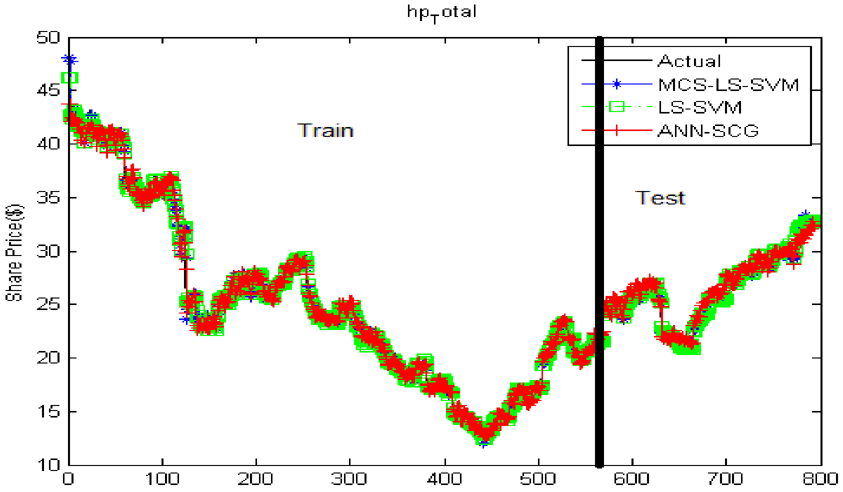

Fig. 16 Train andTest results for HP company

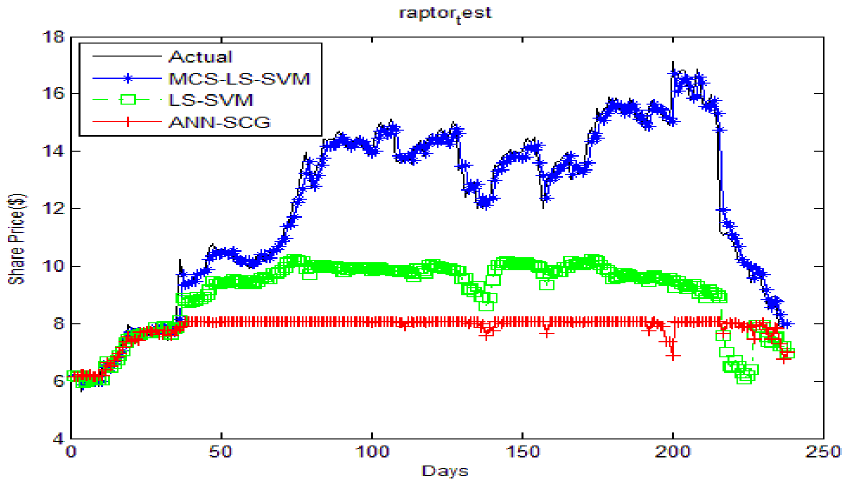

Fig. 17 Test results for Raptor company

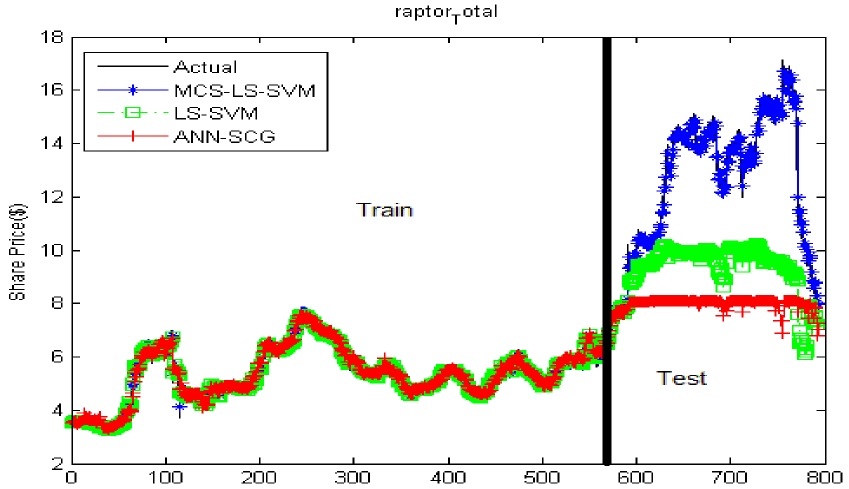

Fig. 18 Train andTest results for Raptor company

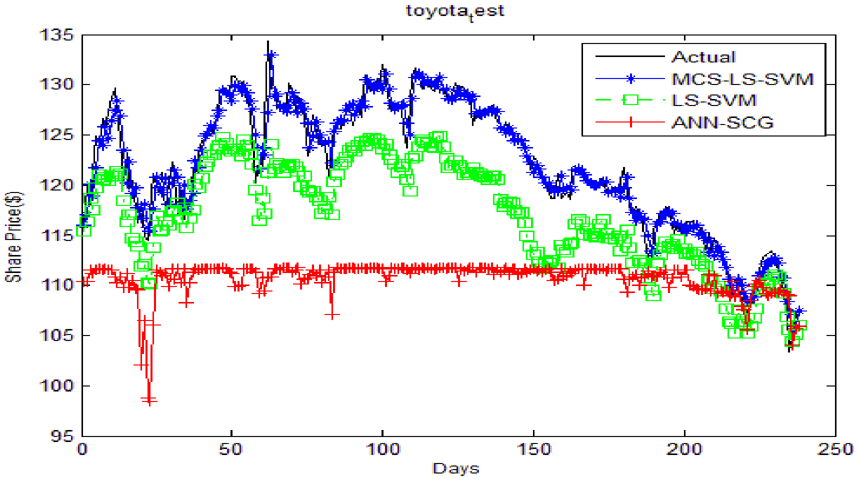

Fig. 19 Test results for Toyota company

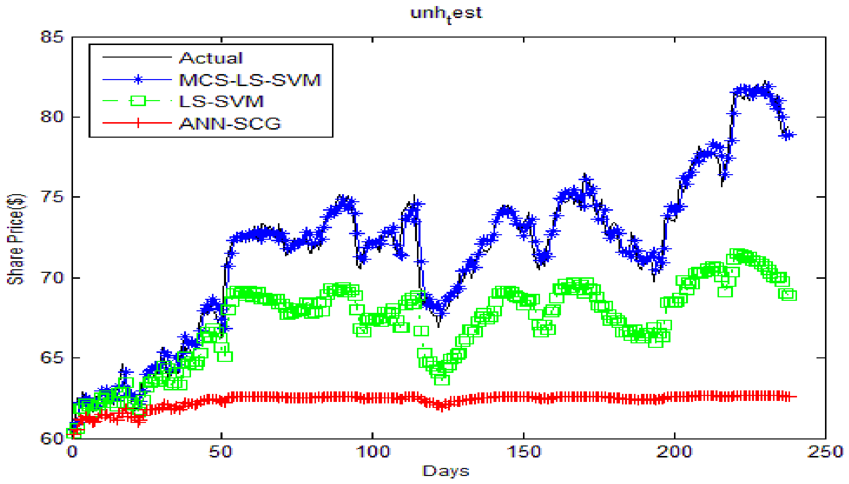

Fig. 20 Test results for United Health company

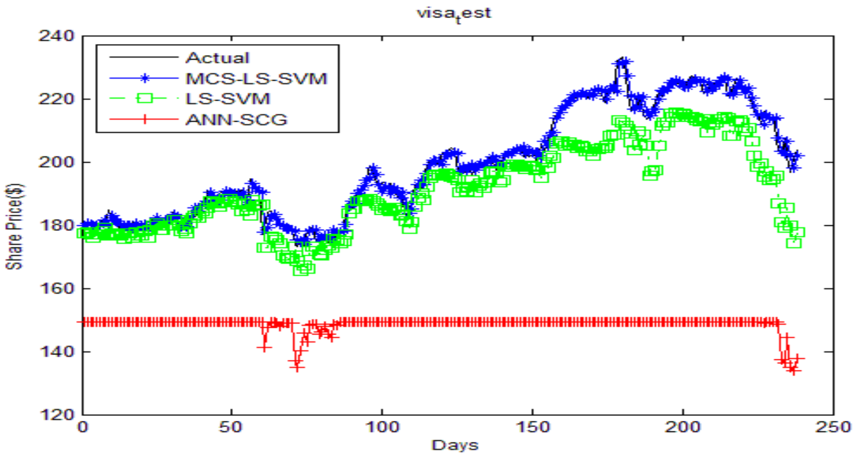

Fig. 21 Test results for Visa company

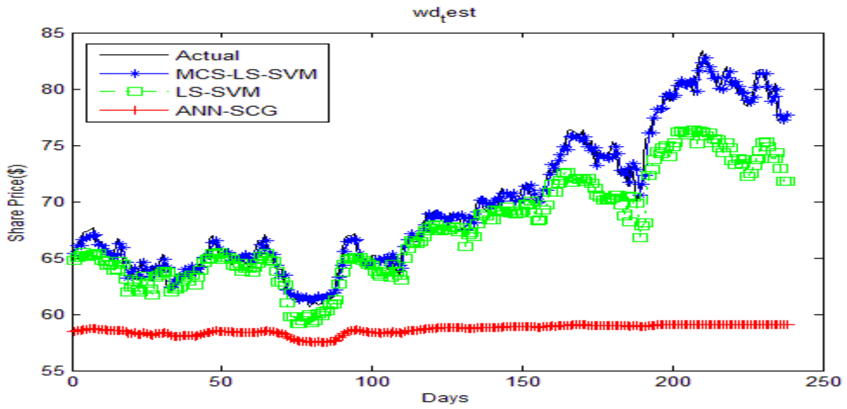

Fig. 22 Test results for Western Digital company

TABLE 4
RMSE for proposed model (test data)

Algorithm Company	MCS-LS-SVM	LS-SVM	ANN-SCG
AmExpress	0.880548	6.418072	13.422974
Apple	6.375421	6.611526	44.577136
AT & T	0.286645	0.295232	0.370319
BAC	0.173055	0.429745	1.234603
Boing	1.460992	25.179651	36.010404
CocaCola	0.403847	0.406244	2.499669
CSCO	0.331386	0.743207	1.657537
Devon	0.72682	0.77137	0.897103
GM	0.480869	0.893606	3.010547
Hp	0.485422	0.50987	0.703013
Raptor	0.470278	3.810533	5.090486
Toyota	1.520742	5.387445	12.654338
UNH	0.792719	5.181284	10.411913
Visa	2.462439	9.905196	53.280265

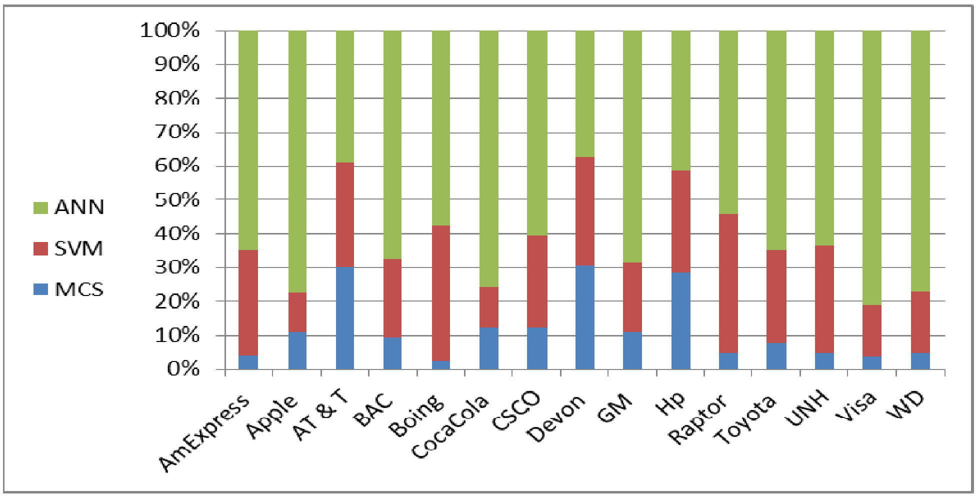

Fig. 23 RMSE for Test results

TABLE 5
MAE for proposed model (test data)

Algorithm Company	MCS-LS-SVM	LS-SVM	ANN-SCG
AmExpress	0.68042	5.123347	12.076106
Apple	4.639133	4.77267	37.049225
AT & T	0.221049	0.228984	0.28953
BAC	0.139494	0.331527	0.95271
Boing	1.110491	20.404714	32.936755
CocaCola	0.343324	0.310216	1.952098
CSCO	0.20502	0.568126	1.323726
Devon	0.595158	0.620055	0.705313
GM	0.382395	0.68977	2.562733
Hp	0.330151	0.341525	0.553495
Raptor	0.30588	3.168884	4.345035
Toyota	1.109046	4.89144	11.317913
UNH	0.590533	4.454317	9.326017
Visa	1.799991	7.96308	50.533963

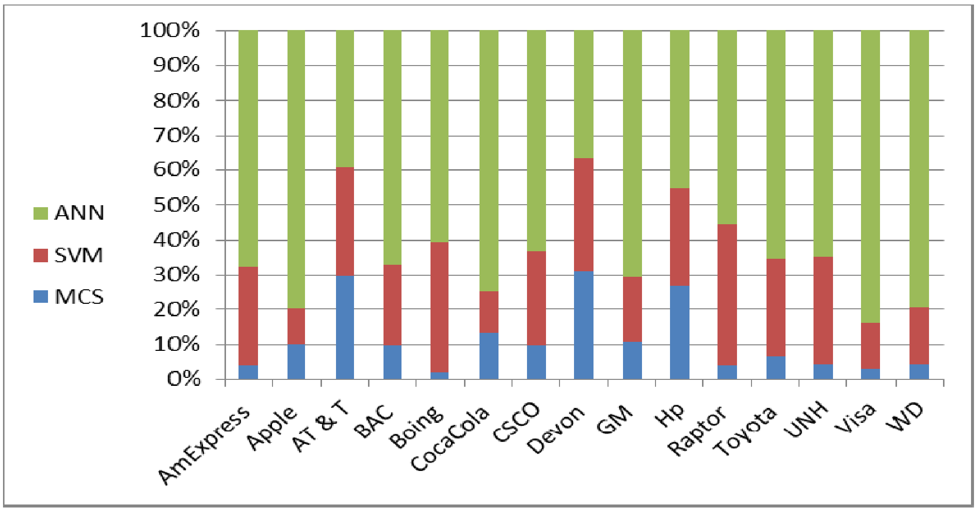

Fig. 24 MAE for Test results

TABLE 6
PMRE for proposed model (test data)

Algorithm / Company	MCS-LS-SVM	LS-SVM	ANN-SCG
AmExpress	0.841671	5.979947	14.407969
Apple	0.939361	0.966161	7.157149
AT & T	0.639885	0.662396	0.837282
BAC	0.924032	2.097881	5.951571
Boing	0.934145	15.927115	26.60704
CocaCola	0.869147	0.790209	4.913165
CSCO	0.888894	2.362508	5.499938
Devon	0.998531	1.031276	1.186563
GM	1.063365	1.849157	6.87684
Hp	1.280569	1.328086	2.170946
Raptor	2.54864	23.110541	31.774237
Toyota	0.912332	3.956713	9.064758
UNH	0.827668	5.997417	12.615981
Visa	0.906221	3.847784	24.834406

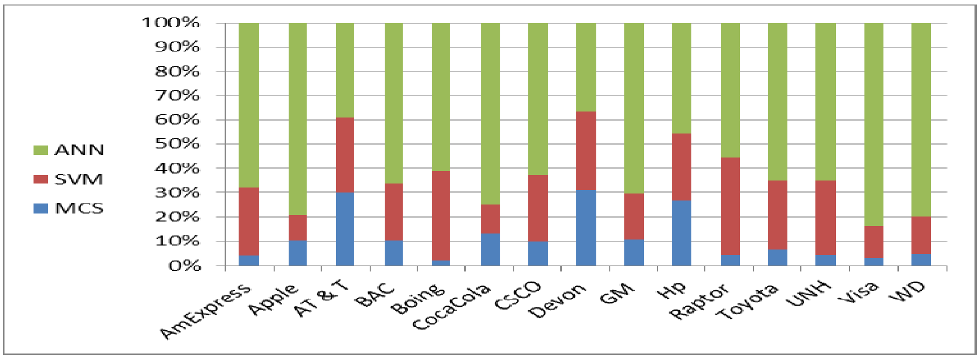

Fig. 25 PMRE for proposed model (test data)

TABLE 7
SMAPE for proposed model (test data)

Algorithm / Company	MCS-LS-SVM	LS-SVM	ANN-SCG
AmExpress	0.004194	0.032614	0.080439
Apple	0.004662	0.004799	0.03835
AT & T	0.003191	0.003307	0.004173
BAC	0.004649	0.011154	0.032751
Boing	0.004668	0.09378	0.160633
CocaCola	0.004345	0.00392	0.024938
CSCO	0.004437	0.012426	0.029465
Devon	0.004961	0.005165	0.005884
GM	0.0053	0.009632	0.03681
Hp	0.006302	0.006516	0.010522
Raptor	0.012575	0.149281	0.216618
Toyota	0.004548	0.020464	0.048648
UNH	0.004117	0.03206	0.069576
Visa	0.004515	0.020383	0.145225

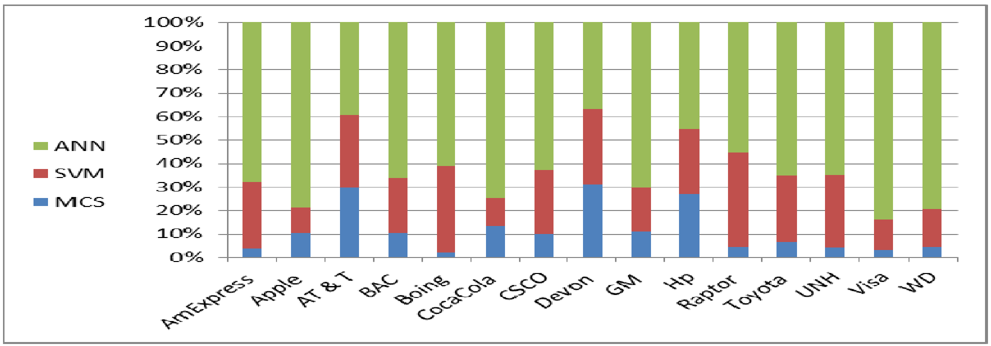

Fig. 26 SMAPE for Test results

TABLE 8
Trend (Direction) prediction accuracy (test data)

Algorithm / Company	MCS-LS-SVM	LS-SVM	ANN-SCG
AmExpress	99.6%	96.7%	92.0%
Apple	99.5%	99.5%	96.2%
AT & T	99.7%	99.7%	99.6%
BAC	99.5%	98.9%	96.7%
Boing	99.5%	90.6%	83.9%
CocaCola	99.6%	99.6%	97.5%
CSCO	99.6%	98.8%	97.1%
Devon	99.5%	99.5%	99.4%
GM	99.5%	99.0%	96.3%
Hp	99.4%	99.3%	98.9%
Raptor	98.7%	85.1%	78.3%
Toyota	99.5%	98.0%	95.1%
UNH	99.6%	96.8%	93.0%
Visa	99.5%	98.0%	85.5%

TABLE 9
RMSE for proposed model (train data)

Algorithm Company	MCS-LS-SVM	LS-SVM	ANN-SCG
AmExpress	0.686453	0.667902	0.68772
Apple	8.110449	8.095458	79.60534
AT & T	0.288142	0.286182	0.328294
BAC	0.214525	0.213852	0.230338
Boing	0.948338	0.922257	1.731691
CocaCola	1.734405	1.6765	2.315521
CSCO	0.273488	0.270263	0.276747
Devon	1.136228	1.128957	1.656125
GM	0.496876	0.494546	0.503192
Hp	0.568257	0.568328	0.610653
Raptor	0.179822	0.1732	0.212469
Toyota	1.022656	1.013559	1.805378
UNH	0.720064	0.703075	0.715621
Visa	1.535204	1.617205	7.351335

TABLE 10
MAE for proposed model (train data)

Algorithm Company	MCS-LS-SVM	LS-SVM	ANN-SCG
AmExpress	0.532364	0.528878	0.540365
Apple	5.909538	5.88038	52.704317
AT & T	0.217237	0.216296	0.253481
BAC	0.161293	0.161742	0.176203
Boing	0.717733	0.714312	1.295356
CocaCola	0.50964	0.55786	1.499443
CSCO	0.196057	0.195681	0.206574
Devon	0.872801	0.864696	1.274983
GM	0.381379	0.3782	0.392627
Hp	0.374993	0.384376	0.435647
Raptor	0.113104	0.109673	0.147678
Toyota	0.797089	0.793468	1.372389
UNH	0.544022	0.531853	0.547174
Visa	1.164535	1.269731	5.402679

TABLE 11
PMRE for proposed model (train data)

Algorithm Company	MCS-LS-SVM	LS-SVM	ANN-SCG
AmExpress	2.35669	2.334835	2.38238
Apple	2.907299	2.880496	21.414087
AT & T	1.57706	1.567389	1.833604
BAC	4.394257	4.402236	4.855174
Boing	2.351199	2.338804	4.160603
CocaCola	2.187756	2.31111	6.243868
CSCO	2.55674	2.557532	2.700418
Devon	3.12376	3.087058	4.510218
GM	3.561442	3.538229	3.638937
Hp	3.644546	3.734497	4.307377
Raptor	5.109913	4.908134	6.769907
Toyota	2.299384	2.279393	3.940889
UNH	2.469106	2.410094	2.478335
Visa	2.472728	2.697023	11.43382

TABLE 12
SMAPE for proposed model (train data)

Algorithm Company	MCS-LS-SVM	LS-SVM	ANN-SCG
AmExpress	0.004928	0.004896	0.005002
Apple	0.006224	0.006193	0.058391
AT & T	0.003343	0.003328	0.003901
BAC	0.00863	0.008654	0.00943
Boing	0.00491	0.004887	0.008863
CocaCola	0.004289	0.004694	0.012631
CSCO	0.005365	0.005355	0.005651
Devon	0.006638	0.006576	0.009701
GM	0.00744	0.007378	0.00766
Hp	0.007481	0.007668	0.008692
Raptor	0.010594	0.010273	0.0138
Toyota	0.004872	0.00485	0.008387
UNH	0.005201	0.005085	0.005231
Visa	0.005057	0.005514	0.023497

TABLE 13
Trend (Direction) prediction accuracy (train data)

Algorithm Company	MCS-LS-SVM	LS-SVM	ANN-SCG
AmExpress	99.5%	99.5%	99.5%
Apple	99.4%	99.4%	94.2%
AT & T	99.7%	99.7%	99.6%
BAC	99.1%	99.1%	99.1%
Boing	99.5%	99.5%	99.1%
CocaCola	99.6%	99.5%	98.7%
CSCO	99.5%	99.5%	99.4%
Devon	99.3%	99.3%	99.0%
GM	99.3%	99.3%	99.2%
Hp	99.3%	99.2%	99.1%
Raptor	98.9%	99.0%	98.6%
Toyota	99.5%	99.5%	99.2%
UNH	99.5%	99.5%	99.5%
Visa	99.5%	99.4%	97.7%

VI. Conclusions

In this paper, Modified Cuckoo Search algorithm (MCS) is used to optimize LS-SVM for daily stock price and trend prediction. Financial technical indicators were used with proposed model to enhance the price and trend prediction accuracy of the model. MCS improves the convergence speed of standard CS. MCS is used in selection of LS-SVM free parameters C (cost penalty), □ (insensitive-loss function) and γ (kernel parameter). The proposed BA-LS-SVM model convergence to a global minimum can be expected in little iterations while compared models have slow convergence speed. Also proposed model overcame the overfitting problem which found in ANN and LS-SVM, especially in case of fluctuations in stock sector. MCS-LS-SVM algorithm parameters are few and can be tuned easily. Optimum found by the proposed model is better than LS-SVM and ANN models. MCS-LS-SVM achieved the lowest error value for all compared evaluation criteria (RMSE, MAE, SMAPE, and PMRE) followed by standard LS-SVM, while ANN-SCG algorithm is the worst one. MCS is very promising in optimizing LS-SVM model and more research efforts should be applied to this new and amazing algorithm.

References

1. M. M. Gaber, A. Zaslavsky and S.Krishnaswamy, Mining Data Streams: A Review, SIGMOD Record, Vol. 34, No. 2, pp.18-26, June 2005.
2. P.Fernández-Blanco1, D.Bodas-Sagi1, F.Soltero1, J.I.Hidalgo1, Technical market indicators optimization using evolutionary algorithms, Proceedings of the 2008 GECCO conference companion on Genetic and evolutionary computation, ACM New York, NY, USA , pp. 1851-1858, 2008.
3. X. Leng, and H. Miller, Input dimension reduction for load forecasting based on support vector machines, IEEE International Conference on Electric Utility Deregulation, Restructuring and Power Technologies (DRPT2004), pp.510 – 514, 2004.
4. V. Cherkassky, and Y. Ma, Practical Selection of SVM Parameters and Noise Estimation for SVM regression, Neural Networks, 17(1), pp.113-126, 2004.
5. J. Suykens, V. Gestel, and J. Brabanter, Least squares support vector machines, World Scientific, 2002.
6. A. Carlos , B. Gary , and A. David, Evolutionary Algorithms for Solving Multi-Objective Problems Springer, 2007.
7. X. S., Yang, S.Deb, Cuckoo search via Lévy flights. Proceedings of World Congress on Nature & Biologically Inspired Computing (NaBIC 2009, India), IEEE Publications, USA, pp. 210–214, 2009.
8. I.Pavlyukevich, Le´vy flights, non-local search and simulated annealing. J Comput Phys, 2007, 226:1830–1844.
9. A.H. Gandomi, X.S Yang., A.H. Alavi, Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng Comput doi: 10.1007/s00366-011-0241-y, vol.29 (1), pp.17–35. 2013.
10. AH. Gandomi, X.S .Yang, Talatahari S, Deb S, Coupled eagle strategy and differential evolution for unconstrained and constrained global optimization. Comput Math Appl 63(1), pp.191–200, 2012.
11. X.S.Yang, S. Deb, Engineering optimization by cuckoo search. Int J Math Modell Num Opt 1(4):330–343(2010)

12. Vazquez RA (2011) Training spiking neural models using cuckoo search algorithm. 2011 IEEE congress on evolutionary computation (CEC’11), pp 679–686.

13. V.R. Chifu, C.B. Pop, I. Salomie, D.S. Suia, A.N. Niculici ,Optimizing the semantic web service composition process using cuckoo search. Intell Distributed Comput V Stud Computat Intell vol. 382, pp.93–102, 2012

14. A. Kumar, S. Chakarverty , Design optimization for reliable embedded system using Cuckoo search. In: Proceedings of 3rd international conference on electronics computer technology (ICECT2011), pp 564–568, 2011

15. A. Kaveh, T. Bakhshpoori, Optimum design of steel frames using cuckoo search algorithm with Levy flights. Structural design of tall and special buildings, vol 21, online first 28 Nov 2011.

16. A.R. Yildiz, Cuckoo search algorithm for the selection of optimal machine parameters in milling operations. Int J AdvManuf Technol. doi: 10.1007/s00170-012-4013-7, 2012.

17. H.Q. Zheng, Y Zhou, A novel cuckoo search optimization algorithm based on Gauss distribution. J Comput Inf Syst vol.8, pp.4193–4200, 2012.

18. S.Walton, O.Hassan, K.Morgan and M.R.Brown "Modified cuckoo search: A new gradient free optimisation algorithm" Chaos, Solitons & Fractals Vol 44 Issue 9, DOI:10.1016/j.chaos.2011.06.004, pp. 710-718, Sept 2011.

19. X.S. Yang, S. Deb, Engineering optimisation by cuckoo search. International Journal of Mathematical Modeling and Numerical Optimisation, vol.1, pp.330–43. 2010.

20. D. Bratton, J. Kennedy, Defining a standard for particle swarm optimization, Proceedings of the 2007 IEEE Swarm Intelligence Symposium, 2007.

21. J. Suykens, J. Vandewalle, Least squares support vector machine classifiers, Neural Processing Letters, Vol. 9 (3), 293-300, 1999.

22. M. ANDRÉS, D. CARLOS, and C. GERMÁN, Parameter Selection in Least Squares-Support Vector Machines Regression Oriented Using Generalized Cross-Validation, Dyna journal, 79(171), 23-30, 2012.

23. http://finance.yahoo.com, Last accessed 1 Jan. 2015.