# Computationally Efficient FLANN-based Intelligent Stock Price Prediction System

Jagdish C. Patra, Nguyen C. Thanh and Pramod K. Meher

*Abstract* — **We propose a computationally efficient and effective novel neural network for predicting the next-day's closing price of US stocks in different sectors: technology, energy and finance. In this paper we used a computationally efficient functional link artificial neural network (FLANN) in making stock price prediction. We modeled the trend in stock price movement as a dynamic system and apply FLANN to predict the stock price behavior. In addition to historical pricing data, we considered other financial indicators such as the industrial indices and technical indicators, for better accuracy. We showed its superior performance by comparing with a multilayer perceptron (MLP)-based model through several experiments based on different performance metrics, namely, computational complexity, root mean square error, average percentage error and hit rate.**

## I. Introduction

Stock trading is an attractive business nowadays due to its high profit. It provides a framework for investors to make profit by buying in "undervalued" stocks and selling "overvalued" stocks. However, stock market is highly volatile and thus, there is a need to develop a model to estimate the risk level and the profit gained in return.

Prior to the emergence of nonlinear models, most studies on stock price prediction tried to capture the relationship between the available financial variables and the stock price with linear assumptions. However, there is no proof that this relationship is perfectly linear and there is no guideline on choosing which variables as input to the prediction system. Linear methods are easy to implement and understand, however, it is unable to capture the non-linear components in the data. The relationship between inputs and output are relatively "static" for linear methods and thus, they are not flexible enough for cases where the input-output relationship changes dynamically.

In recent years, the improvement of computational capabilities has allowed the modeling of complex and multi-variable nonlinear systems, which employs neural network techniques as a powerful tool for time series prediction. There are many applications of neural network in financial forecasting, ranging from exchange rate forecasting, bankruptcy prediction, stock performance and financial portfolio assessment [1]. Box-Jenkins linear method was compared with neural network model, and the result showed that neural network yielded better forecasting performance [2]. Feed forward neural network was able to make better prediction for index options compared to parametric models such as stochastic volatility and stochastic volatility random jump [3].

Preliminary researches on stock market prediction have been done on predicting stock indices of Madrid Stock Exchange, Kuala Lumpur Stock Exchange or Taiwan Stock Index [4]-[6]. Neural network method was then improved with genetic algorithm, in which the networks are trained indirectly using genetic algorithm based weight optimization. The return from trading based on this model is shown to be better than buy-and-hold strategy and the random walk model [7]-[8]. Technical analysis assisted with neural network methods were proposed in [9]. The paper compared the performance of three models, namely multi-layer perceptrons (MLP), adaptive neuro-fuzzy inference system (ANFIS) and the general growing and pruning radial basis function (GGAP-RBF). It has been shown that GGAP-RBF has huge time complexity as compared to MLP and ANFIS but does not outperform MLP and ANFIS in accuracy measures. The false alarm rate in predicting stocks in technology, banking, consumer and cyclical sectors was successfully reduced to less than 10% in experiments with time delay, recurrent and probabilistic neural networks [10]. Popular technical indicators were utilized in combination with neural network models to predict Intraday Foreign Exchange trading rate. The model was able to make positive out-of-sample profits after transaction costs [11].

However, in order to improve the prediction capability, the complexity of the neural network models is increased in general, resulting in longer training time and higher computational resources. Our system tries to overcome this problem and increase the prediction quality by using a computationally efficient and reliable functional link artificial neural network (FLANN), which was originally proposed by Pao [12]. The comparison with the well-studied MLP network shows that FLANN is able to outperform MLP model in various aspects.

## II. Proposed Solution

In this paper, we propose a neural network-based prediction system that is used to predict the closing price of different stocks in the next trading day based on historical closing prices and other financial indicators. The FLANN, which has been proved to be more computational efficient in other researches was used to implement our prediction system.

School of Computer Engineering, Nanyang Technological University, Singapore. E.mail: {aspatra, NGUY0039, aspkmeher}@ntu.edu.sg

978-1-4244-3553-1/09/$25.00 ©2009 IEEE

To capture the changes in economic environment as well as the company's fundamental values, besides the closing prices in the past periods, we have added 13 more variables, namely:

- Dow Jones Industrial Average (DJIA)
- NASDAQ Bank index (IXBK)
- NASDAQ Computer index (IXK)
- 7-day moving average, for past 4 periods ($ma7_1$, $ma7_2$, $ma7_3$, $ma7_4$)
- 30-day moving average, for the past 4 periods ($ma30_1$, $ma30_2$, $ma30_3$, $ma30_4$)
- Standard deviation for the past 30 days (sd)
- Open price ($p_o$)

The data is collected on daily basis except for the open price, which is collected by the time the market is open in the next trading day. For example, the dataset for 30th September 2008 consists of DJIA, IXBK, IXK, $ma7_1$, $ma7_2$, $ma7_3$, $ma7_4$, $ma30_1$, $ma30_2$, $ma30_3$, $ma30_4$ and sd for the date of 30th September 2008, however, the open price $p_0$ is later collected when the market opens on 1st October 2008.

The N-day moving average is calculated as follow:

$$maN = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad (1)$$

where $x_i$ is the actual closing price of day $i$. Thus, in our research, $N$ is set to 7 and 30 for ma7 and ma30, respectively. The standard deviation for the past N days is calculated by the following formula:

$$sd = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2} \qquad (2)$$

where $\bar{x}$ is the average closing price of the N-day period in the analysis.

Fig. 1 illustrates the moving averages to be used in our prediction system. The 7-day moving verages are taken before the historical price points and the 30-day moving averages are taken for the periods before the 7-day moving averages. The basic idea here is that: for those data points that are closer to the day whose price is to be predicted, we will take the day-by-day prices as input, for days prior to those, we use the average values to capture the overall trend of the past few weeks and the past few months.

Fig. 2 shows the basic structure of our prediction system, which takes historical pricing data (closing prices of the past periods) and other financial variables as discussed previously as input. The inputs will be normalized into the same range and then fed into FLANN for making prediction of the closing price for next trading day.

## III. TRIGONOMETRIC FLANN

### A. Overview

Trigonometric FLANN is a single-layer neural network in which the original input pattern in low dimensional space is expanded into a higher dimensional space using trigonometric functions. The basic idea of this model is based on the Cover theorem, which stated that "The probability that classes are linearly separable increases when the features are nonlinearly mapped to a higher dimensional feature space" [13]. Based on this theorem, instead of using multiple neurons or multiple hidden layers, the expansion block in trigonometric FLANN tries to expand the inputs into a higher dimensional space, removing the hidden layers as in MLP. The structure of trigonometric FLANN consists of two parts: functional expansion component and the learning component which is essentially a single neuron [14]-[15]. There have been several reports on effectiveness of the FLANN in solving various nonlinear engineering problems, e.g., channel equalization [14], nonlinear system identification [15]-[17] and financial prediction [18] and temperature control [19]. A novel preprocessing scheme, termed "ensemble encoding" to enhance the MLP capabilities for classification and prediction tasks has been reported [20].

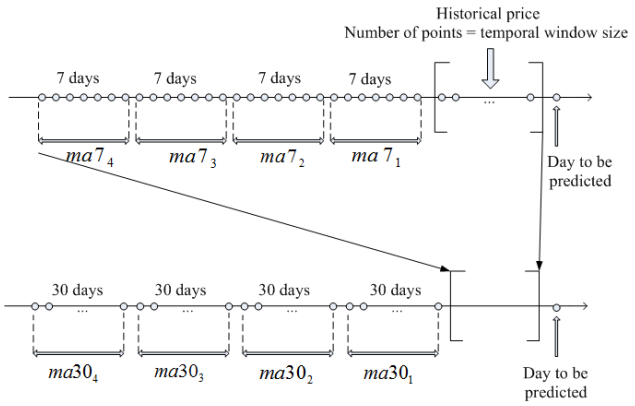


Fig. 1 Moving averages to be used in experiments.

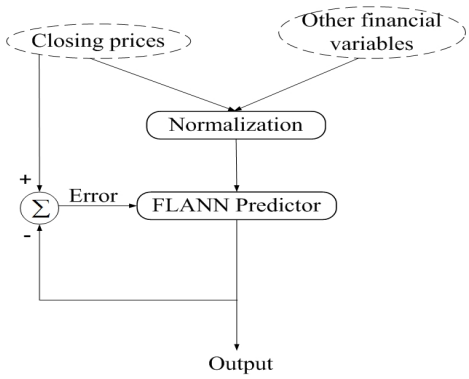


Fig. 2. General model of stock price prediction system.

The functional expansion component will generate a set of linearly independent functions with each element in the input vector taken as the argument of the function [14]-[15]. For example, in Fig. 3, an element $x_i$ (i = 1 and 2) of the input vector can be expanded into several terms: $x_i$, $sin(\pi x_i)$, $cos(\pi x_i)$, $sin(2\pi x_i)$, $cos(2\pi x_i)$, …. Let $X$ be the $n$-dimensional input vector given by

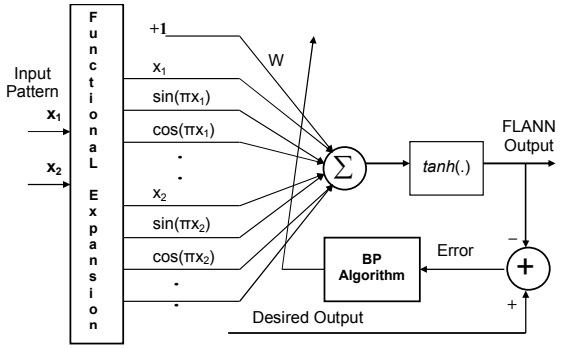


Fig. 3. Trigonometric FLANN.

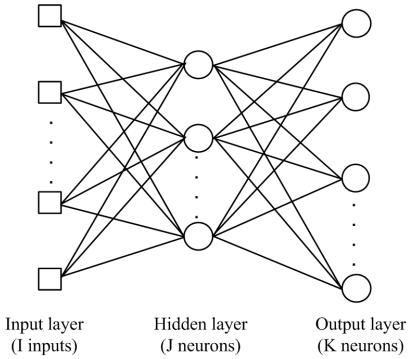


Fig. 4. MLP network with architecture {I-J-K}.

$$X = [x_1 \; x_2 \; \ldots \; x_n] \tag{3}$$

Next, each element $x_i$ ($i = 1 \ldots n$) in the input vector X will be expanded into *2k+1* terms as follow:

$$x_i' = [x_i \; \sin(\pi x_i) \; \cos(\pi x_i) \; \sin(2\pi x_i) \; \cos(2\pi x_i) \; \ldots \; \sin(k\pi x_i) \; \cos(k\pi x_i)] \tag{4}$$

Let Φ be the output of the expansion block with *n(2k+1)* elements:

$$\Phi = [x_1'; x_2'; \ldots; x_n'] \tag{5}$$

where $x_i'$ ($i = 1, 2, \ldots n$) is the *(2k+1)*-dimensional vector on the right hand side of equation (4). Let *W* be the weight vector with *n(2k+1)* elements and $b_0$ be the weight corresponding to the unit bias in Fig. **3**.

$$W = [w_1 \; w_2 \; \ldots \; w_{n(2k+1)-1} \; w_{n(2k+1)}] \tag{6}$$

Finally, the output *y* of trigonometric FLANN is calculated by $y = \delta(W\Phi^T + b_0)$ where δ is the transfer function of the neuron in the learning component. In our experiments, δ is a hyperbolic tangent function, which is defined as:

$$\delta(\theta) = \frac{e^{\theta} - e^{-\theta}}{e^{\theta} + e^{-\theta}} \tag{7}$$

The target values that are used in the training phase are normalized to the range of [-0.9; 0.9] while the value of *y* (predicted value) is in the range of [-1; 1], thus the predicted value can grow larger than the highest closing price in the past or shrink lower than the lowest closing price.

Our experiments showed that learning rate is a very critical parameter which strongly affects the training and prediction quality. For trigonometric FLANN, learning rate must be set to small values like 0.005 to 0.01 (as compared to larger values such as 0.2-0.6 for MLP).

The number of terms generated by expansion block corresponding to an element in the input vector affects the prediction system in two factors: performance and accuracy. Smaller value for the number of terms will result in a neural network structure which is same as a network with only one input layer and one perceptron that is used to combine these inputs, then apply to the transfer function and generate output. In this case, the structure of the network is so simple that it cannot handle the prediction accurately. On the other hand, using too many terms in the expansion will affect the overall performance of the prediction system without necessarily improve the prediction quality. Moreover, higher dimension at the expansion output will also increase the dimension of weight vectors, resulting in "over-fit" problem. The training algorithm to be used for trigonometric FLANN is the same as the case for MLP, which is the gradient descent learning algorithm.

### *B. Complexity of FLANN compared to MLP*

As shown in [18], using the same back-propagation algorithm, trigonometric FLANN has lower computational complexity as compared to MLP, and thus, FLANN is able to converge faster. Each iteration consists of three phases:

- Forward calculations to find the output given current weight values.
- Back error propagation due to back-propagation training algorithm.
- Calculation of adjustment in weights.

Consider the MLP network in Fig. **4** with architecture {I-J-K} and a FLANN with architecture {D-K}. For example, the FLANN architecture in Fig. **3** has D = 3n and K = 1 (only one output). The computational complexity is measured in terms of addition, multiplication and hyperbolic tangent *tanh*(.) operations. This complexity analysis has been shown in [18] and the result is shown in Table I.

TABLE I
COMPUTATIONAL COMPLEXITY OF MLP AND FLANN

| Operator | Number of operations | |
|---|---|---|
| | MLP | FLANN |
| Addition | 2IJ + 3JK + 3K | 2K(D+1) + K |
| Multiplication | 3IJ + 4JK + 3J + 5K | 3K(D+1) + 2K |
| tanh(.) | J + K | K |

Since there is no hidden layer in FLANN, its computational complexity is significantly lower than MLP, especially for hyperbolic tangent calculation. Even though extra computations need to be done during the functional expansion phase, this phase only needs to be processed once during initialization.

## IV. Simulation Studies

### A. Input dataset

In order to evaluate the performance of FLANN, we selected three stocks from three different sectors: IBM representing technology sector, Exxon Mobil (XOM) representing energy sector and Citigroup Inc (C) for the banking sector. Thus, the robustness of different networks can be evaluated for different sectors, which have different behaviors even for the same period.

The historical price data of these stocks as well as the industrial indices (DJIA, IXBK and IXK) was collected from Yahoo Finance website at http://finance.yahoo.com/ for the period from January 1998 to April 2008, which consists of 2478 sets of data. After having collected the historical closing price of the three stocks, the moving average and standard deviations were calculated based on (1) and (2).

### B. Performance metrics

There are basically two measures for the quality of a prediction system, firstly, the degree of closeness between the predicted value and the actual one, secondly, the directional correctness of the predicted value, which sometimes is more important for investors than the numerical value.

Let $\hat{x}_i$ ($i$ = 1, 2, …, $n$) be the predicted value (after denormalization) of the closing price for trading day $i$ in the period of $n$ days to be analyzed and $x_i$ be the actual closing price for that day. We use three performance metrics to compare different network architectures: Root mean square error, average percentage error and hit rate. The first two are meant to measure the numerical accuracy while hit rate is used to measure the directional correctness.

#### 1) Root Mean Squared Error (RMSE)

RMSE shows the deviation between the predicted value and the actual value over $n$ trading days and is defined as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2} \tag{8}$$

#### 2) Average percentage error (APE)

RMSE is a good measure of the prediction accuracy; however, its value is dependent on the actual range of share price, and thus, cannot be used to compare performance of neural network architecture across different stocks. For example, Citigroup's stock usually lies in the range of US$ 15 to 25 while IBM's stock is higher than US$ 100 for most of the time. As a result, the RMSE calculated from experiments with Citigroup will be lower than IBM, which does not reflect the actual performance of the networks.

To overcome this problem, we have used average percentage error (APE) in order to have a comparable measure across experiments with different stocks. APE is calculated as:

$$APE = \frac{1}{n}\sum_{i=1}^{n}\frac{|x_i - \hat{x}_i|}{x_i} \times 100\% \tag{9}$$

#### 3) Hit rate

Hit rate was proposed as an additional performance metric which assesses whether the neural network can accurately predict the trend of movement of tomorrow's closing price as compared to today, *i.e.*, whether the network can correctly predict the tomorrow's closing price as higher or lower than today. Hit rate R is defined as:

$$R = \frac{1}{n}\sum_{i=1}^{n} u\left[(\hat{x}_i - x_{i-1})(x_i - x_{i-1})\right], \tag{10}$$

where $u[x]$ is a unit step function given by:

$$u[x] = \begin{cases} 1 \ for\ x \geq 0 \\ 0 \ otherwise \end{cases} \tag{11}$$

Hit rate lies in the range between 0 and 1, with the value of 1 resulted when all the trends are predicted correctly in $n$ days and 0 if all the trends are predicted incorrectly.

### C. Temporal window size

Temporal window size refers to the number of days that the day-by-day historical closing prices are taken as input to feed into the network. For example, consider a prediction system using neural network with training period of 20 days and temporal window size of 3 days. Let $X_i$ and $d_i$ ($i$ = 1, 2, ..., 20) be one training pattern with input vector $X_i$ and target output $d_i$. Input $X_i$ is a vector which includes historical prices and other financial indicators such as industrial indices, moving averages. For a training period of 20 days, we will have 20 such pairs of input $X_i$ and target output $d_i$. A temporal window size of 3 days means that the closing prices of the past 3 trading days will be included as input to the prediction system. For a particular stock, $X_i$ has the form as follow:

$$\begin{aligned} X_i = [&(historical\ price)_w\ ma7_{i1}\ ma7_{i2}\ ma7_{i3}\ ma7_{i4} \\ &ma30_{i1}\ ma30_{i2}\ ma30_{i3}\ ma30_{i4}\ sd_i\ djia_i\ p_{oi}], \end{aligned} \tag{12}$$

where $w$ is the temporal window size. For a temporal window size of value $w$ = 3, we have:

$$(historical\ price)_w = (x_{i1}\ x_{i2}\ x_{i3}) \tag{13}$$

where $x_{i1}$, $x_{i2}$, $x_{i3}$ are closing prices of 1, 2 and 3 days ago, respectively.

### D. Decaying learning rate

In our experiments, it is observed that the training of trigonometric FLANN is much dependent on the value of learning rate. During the training period, learning rate should be reduced when the training error is reaching the optimal

value in order to avoid fluctuation. The decaying learning rate can be implemented as follow:

- For the first 500 iterations, the learning rate is kept constant at the value initialized by users.
- For the remaining iterations, the learning rate is reduced based on the following formula:

$$\alpha_t = \alpha_0 (1 - \frac{n_t - 500}{n_0}) \quad (14)$$

In the above formula, $\alpha_0$ is the initial learning rate set by the user, $n_t$ is the current iteration and $n_0$ is the total number of iterations set by the user.

## V. Experiment Results

### A. Description

The performance of trigonometric FLANN is compared with MLP based on the three performance metrics discussed in Section IV. The number of neurons in the hidden layer of MLP is set to 15. This parameter is resulted from extensive experiments with MLP in order to derive the comparatively optimal structure among all the MLP networks experimented.

In order to evaluate the performance of each network structure, we have developed a Java application in which the user is able to vary the parameters such as number of neuron in hidden layers for MLP, the number of terms in the trigonometric expansion of FLANN or the temporal window size. Fig. **5** shows a screenshot of our application in which the user can perform the experiments with IBM, XOM and C stocks in any period from January 1998 to April 2008.

The testing period is selected to be from 2nd January 2008 to 31st March 2008, which consists of 60 trading days. In order to make prediction for a particular day, the network is trained for a certain training period (set by user) counted from the day before that day. For example, in order to make prediction for 2nd January 2008 and the training period is set to 20 days; the neural network will be trained using the data from 3rd December 2007 to 31st December 2007, which consists of 20 trading days. The dataset for a particular day contains the historical closing prices, industrial indices and financial indicators as discussed in section II. The network is trained again to make prediction for the next day, thus, the latest trend of price movement is always captured.

### B. Experiment with trigonometric FLANN

In this experiment, the number of terms corresponding to a single element in the input vector is varied from 3 to 15 and the resulting performance metrics are recorded. The experiment was done on IBM stock with the settings as in Table II and its results are shown in Table III.

From Table III, we noticed that the RMSE and APE are approximately the same for the number of terms from 5 to 11. Thus, we decided to choose the number of expansion terms to be 5, which result in highest hit rate and less complex structure. From Table II, the number of elements in the input vector is $n$ = 15; each element will be expanded into 5 terms (thus, $2k+1$ = 5). Based on equation (5), the output Φ of the expansion block is a vector with 15×5 = 75 elements. Thus, the dimension of the input to the neuron in learning component in Fig. **3** is 75. A close agreement between the predicted and actual price of IBM stock can be seen in Fig. **6**.

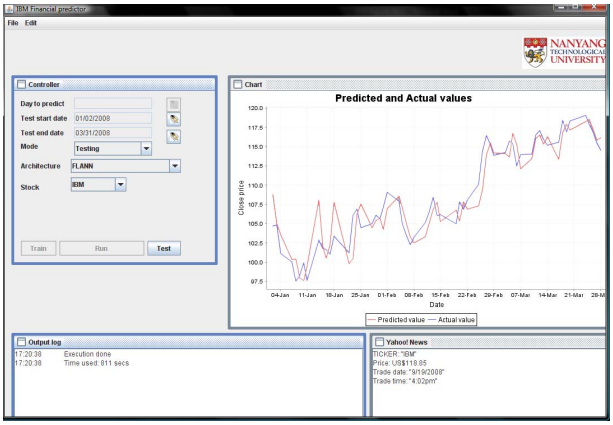


Fig. 5. Graphical user interface.

TABLE II
FLANN CONFIGURATIONS FOR EXPERIMENTS WITH DIFFERENT NUMBER OF EXPANSION TERMS

| Stock | IBM |
|---|---|
| Learning rate | 0.01 – Decaying |
| Momentum | 0.7 |
| No. of iterations | 4000 |
| Training period | 30 days |
| Temporal window size | 3 |
| Inputs | ☒ DJIA ☒ IXK<br>☒ $ma7_i$ (i = 1 … 4)<br>☒ $ma30_i$ (i = 1 … 4)<br>☒ sd (30 days)<br>☒ $p_o$ |
| No. of expansion terms | Varying from 3 to 15 |

TABLE III
RESULT OF EXPERIMENT WITH NUMBER OF EXPANSION TERMS FOR EACH ELEMENT

| Number of expansion terms ($2k+1$) | RMSE | APE | Hit rate |
|---|---|---|---|
| 3 | 3.85 | 2.68 | 0.49 |
| 5 | 2.02 | 1.46 | 0.62 |
| 7 | 2.18 | 1.64 | 0.58 |
| 9 | 2.00 | 1.41 | 0.6 |
| 11 | 2.14 | 1.62 | 0.53 |
| 13 | 2.31 | 1.77 | 0.55 |
| 15 | 2.42 | 1.91 | 0.5 |

### C. Convergence rate of FLANN

Using the configuration for FLANN and MLP in Table II and Table V, it was observed that FLANN is able to converge faster than MLP. After the first 800 iterations, the training

error (mean squared error) of FLANN can drop to 0.001, as shown in Fig. 7. The faster convergence rate of FLANN implies that the amount of time spent training FLANN is shorter than MLP with the same set of input dataset.

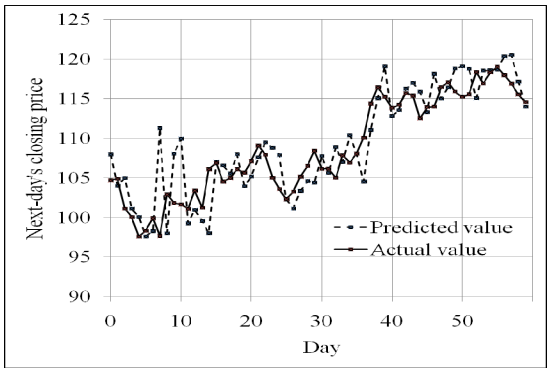


Fig. 6. Result of experiment with FLANN (5 expansion terms) for IBM stock.

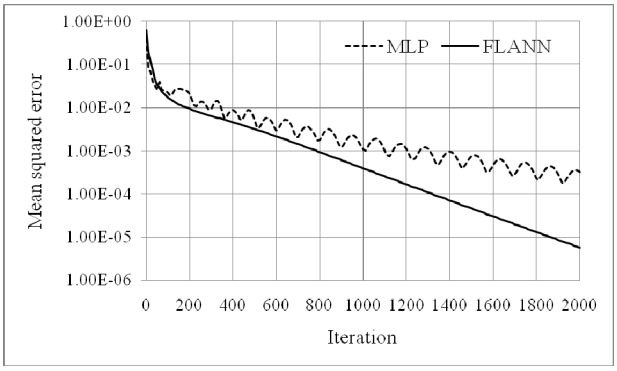


Fig. 7. Training error for MLP and FLANN.

In this experiment, the architecture of MLP is {27-15-1} and the architecture of FLANN is {75-1} (15 elements in the input vector an each element is expanded into 5 terms). Using the result from Table I, the computational complexity of FLANN and MLP networks in this experiment is shown in Table IV. The complexity of FLANN is much lower than MLP, which explains why FLANN is able to achieve shorter training time and faster convergence rate. Our experiments have been done on a computer with specification of Intel Core 2 Duo 1.83 GHz processor and 3 GB of RAM, on average, MLP requires 30 – 35 sec to predict next day's price while FLANN requires approximately 10 sec.

### *D. Comparison between trigonometric FLANN and MLP*

In this experiment, both trigonometric FLANN and MLP are fed with the same set of input and the corresponding performance metrics are compared. The temporal window size is varied from 1 (only yesterday's closing price included) to 15 days (closing prices of the past 15 days are included). The settings for trigonometric FLANN are shown in Table II and the settings for MLP are shown in Table V. The experiment is done on all 3 stocks. DJIA is included as input in the prediction of all 3 stocks, however, IXK (Computer index) is included for IBM only and IXBK (Banking index) is included for Citigroup only.

The APE and Hit rate of MLP and FLANN corresponding to each value of temporal window size (varied from 1 to 15) are shown in Fig. 8 and Fig. 9, respectively. From Fig. 8, we notice that in order to achieve APE less than 3% for the cases of IBM and XOM stocks or 6% for C stock, FLANN requires a temporal window with size at most 5 days while MLP requires a temporal window size of 7 days or more. This shows that the functional expansion block together with moving averages and standard deviations are able to compensate for the lack of huge amount of day-by-day historical prices in FLANN.

TABLE IV.
COMPUTATIONAL COMPLEXITY OF FLANN AND MLP IN EXPERIMENT

| Operator | Number of operations | |
|---|---|---|
| | MLP | FLANN |
| Addition | 858 | 153 |
| Multiplication | 1325 | 230 |
| tanh(.) | 16 | 1 |

TABLE V
OPTIMAL MLP CONFIGURATION

| Type | MLP | FLANN |
|---|---|---|
| Stock | IBM, XOM, C | |
| Learning rate | 0.5 - Decaying | 0.01 - Decaying |
| Momentum | 0.7 | 0.7 |
| No. of iterations | 8000 | 4000 |
| Training period | 30 days | 30 days |
| Temporal window size (days) | 15 | 3 |
| Inputs | ☒ DJIA ☒ IXK (IBM)<br>☒ IXBK (Citigroup)<br>☒ $ma7_i$ (i = 1 … 4)<br>☒ $ma30_i$ (i = 1 … 4)<br>☒ sd (30 days)<br>☒ $p_o$ | |
| Architecture | {27-15-1} | {75-1} |
| Avg. time taken | 30 – 35 sec | 10 sec |

Moreover, Fig. 9 shows that FLANN is able to predict the trend of price movement more correctly with a temporal window size of less than 9 days. An important point is that the hit rate of FLANN for these cases are all higher than 0.5, as compared to MLP where hit rate sometimes drops to 0.4 even with temporal window size larger than 7 days.

The best performance metrics for experiments with IBM, XOM and C stocks are shown in Table VI. Note that the optimal result for MLP is collected when the temporal window size is 15 days while the optimal result for FLANN only requires a temporal window size of one day.

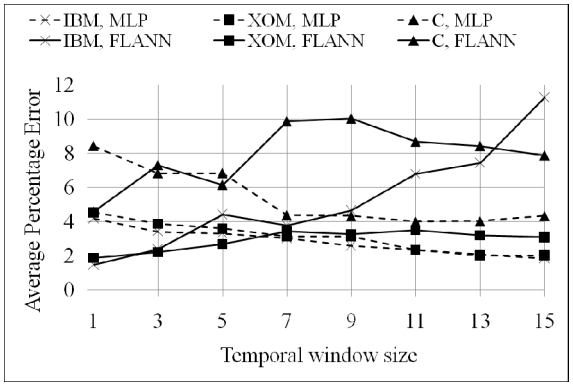


Fig. 8. APE of MLP and FLANN with different temporal window size.

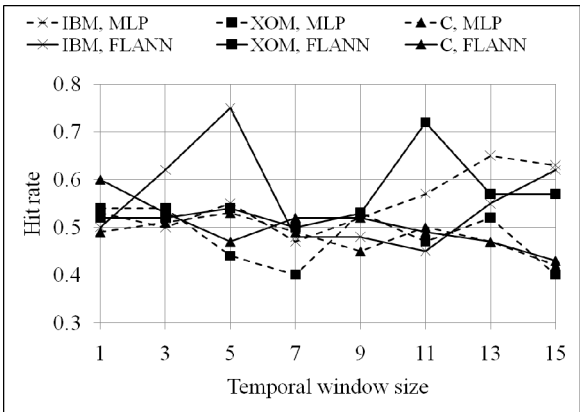


Fig. 9 Hit rate of MLP and FLANN with different temporal window size.

TABLE VI
MLP AND FLANN PERFORMANCE ON DIFFERENT STOCKS

| Performance Metric | IBM | | XOM | | C | |
|---|---|---|---|---|---|---|
| | MLP | FLANN | MLP | FLANN | MLP | FLANN |
| RMSE | 2.38 | 2.02 | 2.08 | 1.94 | 1.31 | 1.46 |
| APE | 1.83 | 1.46 | 2.01 | 1.86 | 4.01 | 4.58 |
| Hit rate | 0.53 | 0.62 | 0.54 | 0.57 | 0.51 | 0.43 |

The predicted and actual closing prices obtained from the above experiments (which results in optimal performance metrics for MLP and FLANN) for IBM, XOM, C are shown in Fig. **10**, Fig. **11** and Fig. **12**, respectively. We notice from these results that FLANN is able to follow the trend of movement more closely than MLP for all three cases. MLP usually generates a lot of spikes that move much higher or lower than the actual price movement, although the direction of movement may be correct. On the other hand, the spikes generated by FLANN are generally smaller than MLP, which reflects in smaller APE values, as shown in Table VI.

In the prediction with IBM and ExxonMobil, we noticed that the prediction error (in term of RMSE and APE) is lower than the case of Citigroup stock; this can be explained due to the lower value of Citigroup price compared to IBM and Exxon- Mobil. Moreover, the testing period from 2nd January 2008 to 31st March 2008 when there is a lot of unexpected instability in the banking sector. The result obtained from this experiment is comparable to the earlier research on FLANN for financial indices prediction [18] where the average percentage error of FLANN is from 0.96% (for STI) to 2.09% (for NASDAQ). This shows that FLANN is able to outperform MLP not only in the researches with stock indices but also individual stocks, where the price is much lower compared to the indices and thus, requires higher level of accuracy.

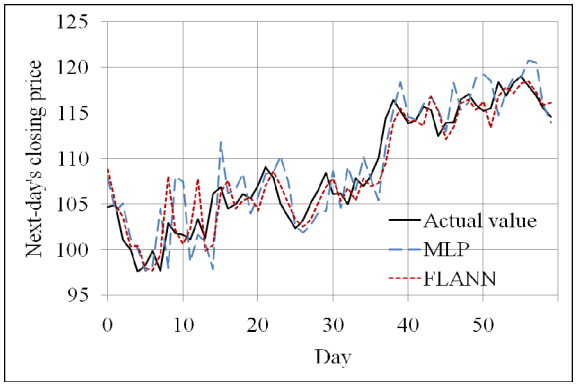


Fig. 10. IBM stock prediction using both network types.

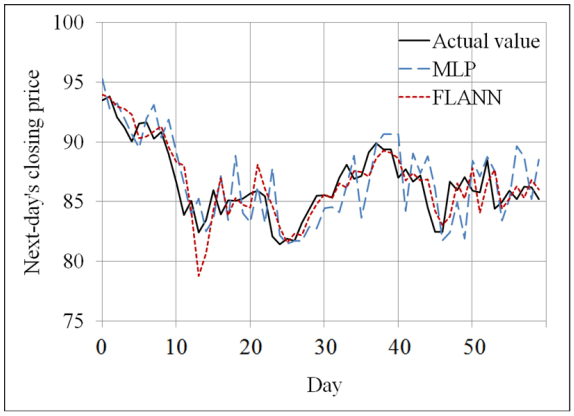


Fig. 11. Exxon Mobil stock prediction using both network types.

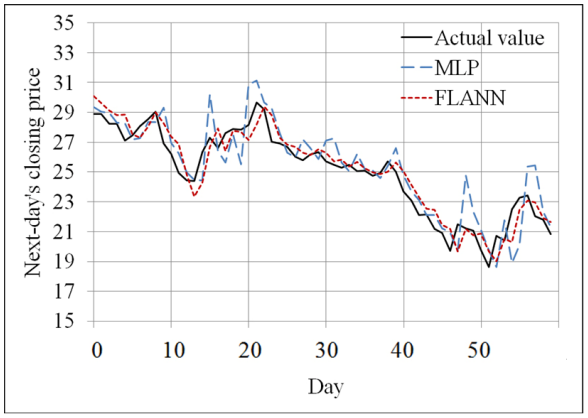


Fig. 12. Citigroup stock prediction using both network types.

## VI. CONCLUSION

In this his paper we developed stock price prediction system based on a novel trigonometric FLANN and has shown its advantages over traditional MLP networks. Besides utilizing historical data, we have added several new attributes such as industrial indices and technical indicators. These features, in combination with the functional expansion of FLANN allowed the prediction system to make more accurate, trend-following predictions with a smaller number of day-by-day historical data required as compared to MLP.

The application of FLANN has been shown to be effective in the prediction of stock prices in the next trading day. By making use of financial indicators, besides historical price, FLANN is able to make better predictions (APE lies in the range of 1.4 % to 4.5 % and higher hit rate) as compared to MLP (APE lies in the range of 1.8 % to 4 % but with lower hit rate). Moreover, the lower complexity of FLANN allows it to achieve shorter training time and faster convergence rate. The FLANN takes only 10 sec to train where as the MLP takes about 30-35 sec. More in-depth study on other financial applications of FLANN can be covered in the future with different algorithms in order to further improve the prediction quality.

## REFERENCES


[1] J. Kamruzzaman, R. Begg, R.A. Sarker, "Artificial Neural Networks in Finance and Manufacturing", Idea Group Inc, Hershey, PA, USA, 2006.

[2] G. Abdelmouez, S.R. Hashem, A.F. Atiya, M.A. El-Gamal, "Neural Network vs. Linear Models for Stock Market Sectors Forecasting", IJCNN'07, Florida, USA, pp. 1365 – 1369, Aug. 2007.

[3] R. Gençay, R. Gibson, "Model Risk for European-Style Stock Index Options", IEEE Trans. Neural Networks, vol. 18, no. 10, pp. 193 – 202, Jan. 2007.

[4] F. Ferna´ndez-Rodrı´guez, C. Gonza´lez-Martel, S. Sosviall-Rivero, "On the profitability of technical trading rules based on artificial neural networks: evidence from the Madrid Stock Market", Economics Letters, vol. 69, no. 1, pp. 89-94, Oct. 2000.

[5] J. Yao, C.J. Tan, H.L. Poh, "Neural networks for technical analysis: a study on KLCI", Int. J. Theoretical and Applied Finance, vol. 2, no. 2, pp. 221-241, 1999.

[6] A.S. Chen, M.T. Leung, H. Daouk, "Application of neural networks to an emerging financial market: forecasting and trading the Taiwan Stock Index", Computers and Operations Research, vol. 30, no. 6, pp. 901-923, May 2003.

[7] G. Dreyfus, "Neural Networks: Methodology and Applications", Birkhäuser, Germany, 2005.

[8] Y.K. Kwon, B. R. Moon, "A Hybrid Neurogenetic Approach for Stock Forecasting", IEEE Trans..Neural Networks, vol. 18, no. 3, pp. 851-864, May 2007.

[9] T.S. Quah, "DJIA stock selection assisted by neural network", Expert Systems with Applications, vol. 35, no. 1-2, pp. 50-58, Jul. 2008.

[10] E.W. Saad, D.V. Prokhorov, D.C. Wunsch II, "Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks", IEEE Trans. Neural Networks, vol. 9, no. 6, pp. 1456 – 1470, Nov. 1998.

[11] M. A. H. Dempster, T. W. Payne, Y. Romahi, G. W. P. Thompson, "Computational learning techniques for intraday FX trading using popular technical indicators", IEEE Trans. Neural Networks, vol. 12, no. 4, pp. 744 – 754, Jul. 2001.

[12] Y.H. Pao, "Adaptive Pattern Recognition and Neural Networks", Reading, MA: Addison-Wesley, 1989.

[13] K.Mehrotra, C.K. Mohan, S. Ranka, "Elements of Artificial Neural Networks", MIT Press, USA, pp. 88-89, 1997.

[14] J.C. Patra, R.N. Pal, R. Baliarsingh and G. Panda, "Nonlinear channel equalization for QAM signal constellation using artificial neural networks", IEEE Trans. Systems Man and Cybernetics, part B, vol. 29, no. 2, pp. 262-271, Apr. 1999.

[15] J.C. Patra, R.N. Pal, B. N.Chatterji and G. Panda, "Identification of nonlinear dynamic systems using functional link artificial neural networks," IEEE Trans. Systems Man and Cybernetics, part B, vol. 29, no. 2, pp. 254-262, Apr. 1999.

[16] Y.-H . Pao, S.M. Philips and D. J. Sobajic, "Neural-net computing and intelligent control," Int. J. Control, vol. 56, no. 2, pp. 263-289, 1992.

[17] S. Chen and S. A. Billings, "Neural networks for nonlinear dynamic system modeling and identification," Int. J. Control, vol. 56, no. 2, pp. 319-346, 1992.

[18] J.C. Patra, W. Kim, P.K. Meher, E.L. Ang, "Financial prediction of major indices using computational efficient artificial neural networks," IJCNN '06, Vancouver, Canada, pp.2114 – 2120, Jul. 2006.

[19] S. Narayan, G. A. Tagliarini and E. W. Page, "Enhancing MLP network using a distributed data representation," IEEE Trans. Systems Man and Cybernetics, art B, vol. 26, no. 1, pp. 143-149, Feb. 1996.

[20] C-H. Chen, C-T. Lin, C-J. Lin, "A Functional-Link-Based Fuzzy Neural Network for Temperature Control", FOCI'07, Honolulu, USA, pp. 53 – 58, Apr. 2007.