Application of multi-GRNN with a gating network in stock prices forecast

LU Jin-na(卢金娜), HU Hong-ping(胡红萍), BAI Yan-ping(白艳萍)
(*School of Science, North University of China, Taiyuan 030051, China*)

Abstract: This paper proposes the generalized regression neural network(GRNN) model and multi-GRNN model with a gating network by selecting the data of Shanghai index, the stocks of Shanghai Pudong Development Bank(SPDB), Dongfeng Automobile and Baotou Steel. We analyze the two models using Matlab software to predict the opening price respectively. Through building a softmax excitation function, the multi-GRNN model with a gating network can obtain the best weights. Using the data of the four groups, the average of forecasting errors of 4 groups by GRNN neural model is 0.012 208, while the average of the multi-GRNN models's with a gating network is 0.002 659. Compared with the real data, it is found that the both results predicted by the two models have small mean square prediction errors. So the two models are suitable to be adopted to process a large quantity of data, furthermore the multi-GRNN model with a gating network is better than the GRNN model.

Key words: stock forecasting; GRNN model; gating network; softmax incentive

CLD number: TP183 **Document code:** A
Article ID: 1674-8042(2012)04-0374-05 **doi:** 10.3969/j.issn.1674-8042.2012.04.015

The change trend of the stock market has been always one of the main concerns for stock investors. So, the stock market index and the stock price forecast become an important issue in both security field and academia. Because the stock market is a highly complex nonlinear dynamic system, factors of the stock, such as the recent opening price, the lowest price, the highest closed price and the related indexes and so on, have great influence on its change. At the same time, political, economic, psychological and many other factors also have certain effect on its change[1]. This article proposes the generalized regression neural network(GRNN) model and the multi-GRNN model with a gating network simulates and to forecast the stock data for getting better results.

The GRNN is a nonlinear regression based on the theory of feed forward neural network model. It approaches a nonlinear function through the activation neurons, uses local accept domain to execute the function mapping, and realizes that the input vector function values are approached by local accept region of the corresponding function value neurons vector mapping. The GRNN needs only simple smoothing parameters without circulation training process, and this results in much potential in the nonlinear intends, and can get better predicting results of a large number of data processing models[2].

The main idea of the gating network is to distribute the task to one or more GRNNs. If the actual output and the desired output values of GRNN are different, the network modifies the weights of these GRNNs and the gating network through an excitation function to localize every GRNN and adapt it to the assigned task. Finally, fewer errors are got by comparison with the expectation values.

1 Model analysis

This paper establishes two models to come to conclusions after the models′ forecasts, analysis and comparison based on the target data. The model Ⅰ builds a GRNN model with 15 input neurons. The model Ⅱ is multiple GRNNs with an adaptive gating network, which functions as a model-output selector. At first it builds 15 GRNN models each having a single neuron, then establishes a gating network according to the front 15 models' weights, and finally gets better result through the adjustment of the values of the weights.

1.1 GRNN (model Ⅰ)

The GRNN architecture was put forward by Specht[4], which is subsumed the radial-basis function

* **Received data:** 2012-08-18
Foundation item: Postdoctoral Granted Financial Support from China Postdoctoral Science Foundation(20100481307); Natural Science Foundation of Shanxi Province, China(No. 2009011018-3)
Corresponding author: LU Jin-na (Jinna4813@gmail.com)

method. It approximates any arbitrary function between the input and output vectors, drawing the function estimatation directly from be training data. Because GRNN is a one-pass learning algorithm with a highly parallel structure, even with lager amount of data in a multi-dimensional measurement space, the algorithm can provide smooth transition from one observed value to another. The algorithmic form can be used for any regression problem in which an assumption of linearity is not justified.

In GRNN, each training vector has a corresponding radial-basis neuron in the hidden layer, and the hidden neurons store all the training vectors. The network structure is shown in Fig. 1.

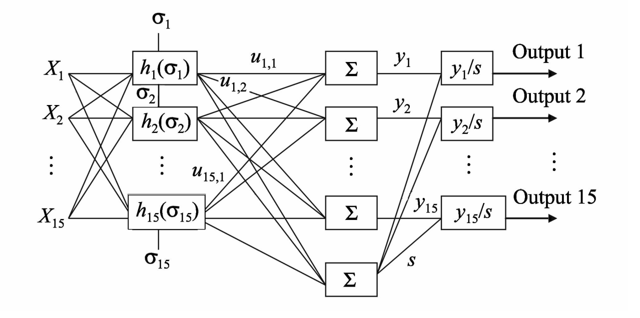

Fig. 1 GRNN architecture

The neurons of the input layer are simple distribution units that pass input variables to the model layer. The hidden layer uses the Gaussian transformation function to control the output of this layer, and so to restrain the output cell activation. The output layer uses joint density function to expect the output operations, which reduces the errors.

The input neurons of the first layer are product terms of the weighted input vector and the corresponding threshold values, and then the output neurons of the first layer are calculated by Gaussian function. Among them, the weighted input vector are denoted as the squared distances between the input vectors and the training vector. The influence on the output neurons from the input neurons has an exponential decay with the increase of the distance.

The base function of hidden layer nodes of GRNN is Gaussian function which has local approximation ability, and this results in fast network learning. In addition, because GRNN has only one threshold, network learning completely relies on data sample, which can possibly avoids the influence of the subjective factors on the predicted results.

1.2 Multi-GRNN model with a gating network (model Ⅱ)

This model is designed to use 15 GRNNs to learn the inputs individually with an adaptive gating network to select the best inputs, based on the error magnitude, out of the total inputs. The model then combines the results to produce the finally predicted value. The model is divided into two parts. The first part establishes 15 GRNN neural network models for 15 input neurons, respectively. The second part builds a gating network, where we consider the weights obtained in the first part as the gating network's inputs. Adjusting the size of the weights by calculating the errors between output values and the expectations, the output results can be optimized. The nodel structure is shown in Fig. 2.

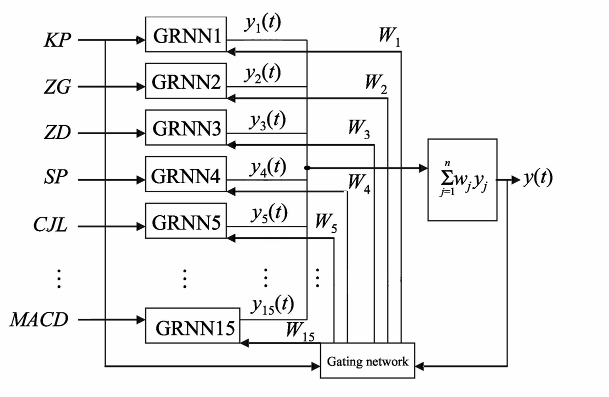

Fig. 2 Multi-GRNN architecture

The gating network assigns weights to each GRNN module. The error magnitude weights are sorted to create a GRNN module ranking based on the mean square percentage error(MSPE). Then the weight value is applied to the rank GRNN module. The final result is calculated by

$$Y = \sum_{j=1}^{n} g_j y_j ,$$

where

$$g_j = \frac{\exp(w_{ij})}{\sum_i \exp(w_{ij})},$$

and w_{ij} is normalized.

Compared with GRNN model, multi-GRNN model increases gating network adjustment in the end of 15 GRNNs, based on the error magnitude, and it can improve the forecast accuracy. The gating network functions as a model-output selector and assigns weights based on the error magnitude of multi-GRNN module before computing the final forecasting results. So it keeps the characteristics of GRNN model and emerges new characteristics. It estimates values for continuous dependent variables through the use of nonparametric estimators of probability density functions – $f(\boldsymbol{x}, \boldsymbol{y})$ just like GRNN. It allows the appropriate form to be expressed as a probability density function that is empirically determined from the observed data. Thus, the approach is not limited to any particular form and requires no prior knowledge of the appropriate form[4].

2 Learning rule of model Ⅰ

2.1 Learning rule of Model Ⅰ

1) Input layer calculates the distances between the input vectors and weights vectors. When a new vector is input to the network, the distances between the new vector and each unit weight vector in hidden layer are calculated again. Formula is as follows

$$d = \sqrt{(\boldsymbol{x}_i - \boldsymbol{u}_i)^{\mathrm{T}}(\boldsymbol{x}_i - \boldsymbol{u}_i)}, i = 1,2,\cdots,N, \quad (1)$$

where $\boldsymbol{x}$ is the estimator input vector, $\boldsymbol{u}$ is unit weight vector in hidden layer and N is sample size.

2) Gaussian function in hidden layer is based on the following formula

$$h_i = \exp(-d_i^2/(2\sigma^2)), \quad (2)$$

where d_i is the distance between the input vector $\boldsymbol{x}$ and the training vector $\boldsymbol{u}$; σ is the smooth factor, which decides the shape of basis functions in the hidden layer, $\sigma>0$.

Because the smooth factor has a major influence on the performance of the network, the bigger smooth factor is, the smoother approximation process of the network to the sample data is; the smaller the smooth factor is, the stronger approximation performance of the network to the sample is.

Therefore, it is needed to continue to try to get the best value.

3) The GRNN output layer is linear, and the output neurons are calculated by

$$s = \sum_{i=1}^{n} h_i, \quad (3)$$

$$y_i = \frac{\sum_{i=1}^{n} h_i u_{ij}}{s}, \quad (4)$$

where u_{ij} is the target output corresponding to input training vector $\boldsymbol{x}_i$ and output $\boldsymbol{y}_j$.

4) The output expectation based on the training sample $\boldsymbol{X}$ is expressed by

$$E[\boldsymbol{y}/\boldsymbol{x}] = \frac{\int_{-\infty}^{+\infty} \boldsymbol{y}f(\boldsymbol{x},\boldsymbol{y})d\boldsymbol{y}}{\int_{-\infty}^{+\infty} \mathrm{f}(\boldsymbol{x},\boldsymbol{y})\mathrm{d}\boldsymbol{y}}, \quad (5)$$

where $\boldsymbol{x}$ is the estimator input vector; $\boldsymbol{y}$ is the output of the estimator; $f(\boldsymbol{x},\boldsymbol{y})$ is the joint probability density function of $\boldsymbol{x}$ and $\boldsymbol{y}$[4].

2.2 Learning rule of Model Ⅱ

1) Extract target input vector, and do the normalized processing. Formula is

$$\boldsymbol{x}_j(i) = \boldsymbol{x}_j(i)/\max(\boldsymbol{x}_j(k)), k = 1,2,\cdots,N, \quad (6)$$

where $\boldsymbol{x}_j(i)$ expresses the input value $\boldsymbol{y}_i$ of neurons $\boldsymbol{y}_j$. The number of training samples is N.

2) Extract the number of input neurons, and build multi-GRNN model according to the number of neurons, and concludes the output results and initialized weights.

3) Mediate the result of step 2) through the gating network. The gating network uses softmax excitation function, just as that gating network uses softmax incentive g_j for the output neurons $\boldsymbol{y}_j$.

$$g_j = \frac{\exp(w_{ij})}{\sum_i \exp(w_{ij})}, \quad (7)$$

where w_{ij} is the target output corresponding to input training vector $\boldsymbol{x}_i$ and output $\boldsymbol{y}_j$.

4) The final output for the network is

$$Y = \sum_{j=1}^{n} g_j \boldsymbol{y}_j, \quad (8)$$

where n is the number of GRNN networks; $\boldsymbol{y}_j$ is the output of GRNN j; g_j is excitation function for output neurons $\boldsymbol{y}_j$ of the gating network.

3 Matlab training and prediction

This paper chooses Shanghai Pudong Development Band(SPDB), Dongfeng automobile, Baotou three stocks as well as the Shanghai index to predict the opening price of the last 5 days from 2011-08-15 to 2012-06-20 according to the selected data. The choice of the data has a great influence on the network result, so we adopt the 100 set of data of the above-mentioned stocks and use Matlab for programming.

In the two models, data is! normalized, and there are a lot of ways of dealing with data. Here, the data wiml be divided by the maximum of all of them, which is to transform all of the inqut and output training data to [0,1] inside. An accuracy measure is often defined in tools of the forecasting error, which is the difference between the actual (desired) and the predicted value. The MSPE is used to check error magnitude, which is calculated according to the 100-day historical results and is normalized to create error magnitude weights. Formula is

$$MSPE = 100 \times \frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2, \quad (9)$$

where N is the number of training samples, $(y_1, y_2, \cdots, y_N)$ expresses the actual value of the training sample, $(\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_N)$ show output values of the prediction model.

For Model Ⅰ, according to the training sample, this paper determines the dimension of input and output neuron network, and determines the 15 input neurons, establishes a GRNN model and predicts last results using statements[6]

$$net = newgrnn\ (Input, Output, \sigma).$$

For Model Ⅱ, according to the training sample, the paper determines the dimension of input and output neuron network, establishes 15 GRNN models according to 15 neurons respectively. And then it adjusts optimal weights using the gating network, according to the error comparison of the final results based on the front 15-GRNN model and weights. At last

$$y = softmax(W)$$

is chosen to minimize the error. The experimental results are shown in Table 1. The Table 2 shows the forecasting errors of the two models.

Table 1 Prediction results

Stock	Day	Actual value	Predicted values of model Ⅰ	Predicted values of model Ⅱ
the Shanghai index KP values	2012-06-14	2 306.78	2 289.694 494 286 537	2 308.951 994 291 741
	2012-06-15	2 299.78	2 303.290 562 385 176	2 304.469 127 354 488
	2012-06-18	2 313.53	2 300.730 423 370 073	2 314.567 397 396 998
	2012-06-19	2 313.459	2 313.285 436 851 229	2 314.087 505 083 577
	2012-06-20	2 300.41	2 313.455 609 018 794	2 306.432 795 417 948
the SPDB KP values	2012-06-14	8.3	8.310 070 692 215 767	8.311 569 406 187 003
	2012-06-15	8.31	8.300 354 635 457 575	8.304 168 488 839 046
	2012-06-18	8.47	8.310 000 000 000 001	8.478 558 038 461 586
	2012-06-19	8.47	8.470 000 000 000 001	8.486 007 757 039 065
	2012-06-20	8.54	8.470 000 000 000 002	8.548 819 361 847 079
the Dongfeng Automobile KP values	2012-06-14	3.41	3.362 698 116 743 164	3.414 045 483 310 596
	2012-06-15	3.41	3.410 262 106 148 517	3.411 271 492 020 246
	2012-06-18	3.39	3.417 685 283 591 788	3.397 273 035 674 043
	2012-06-19	3.38	3.399 529 992 680 672	3.388 310 601 256 602
	2012-06-20	3.35	3.389 917 066 404 872	3.368 702 903 185 198
the Baotou Steel KP values	2012-06-14	6.18	6.182 179 224 870 881	6.205 001 050 652 275
	2012-06-15	6.15	6.183 478 897 671 705	6.177 225 429 119 889
	2012-06-18	6.06	6.154 425 464 094 560	6.130 994 972 394 714
	2012-06-19	6.02	6.090 547 329 562 404	6.088 780 020 714 959
	2012-06-20	5.83	6.021 460 969 698 566	5.905 948 250 400 332

Table 2 Error analysis

MSPE	Shanghai index	SPDB	Dongfeng automobile	Baotou steel	Average of 4
Model Ⅰ	0.002 399	0.008 536	0.008 128	0.029 768	0.012 208
Model Ⅱ	0.000 243	0.000 162	0.000 763	0.009 469	0.002 659

For convenience of comparison, this paper gives four sets of prediction graphs of data with two kinds of models are shown in Figs. 3 – 6.

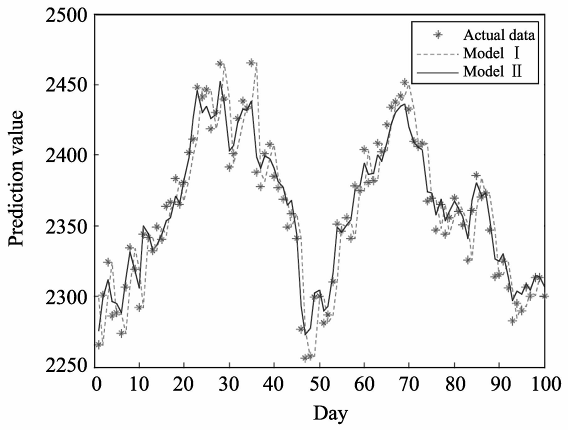

Fig. 3 Shanghai index prediction curve

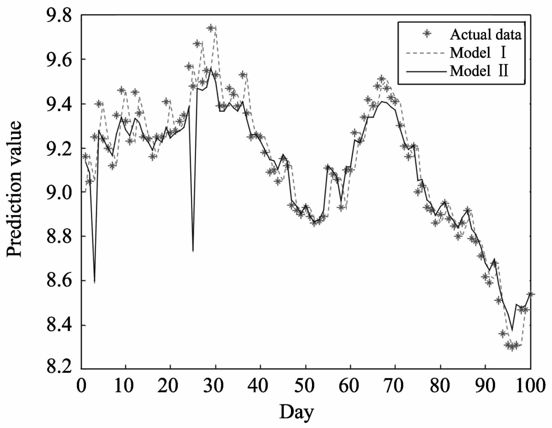

Fig. 4 SPDB prediction curve

The Table 1 is the last 5-day forecasting result for the four stocks and it shows that both the two models for the selection of four sets of data can get better forecasting consequences, and that the predicted values coincide with the actual values.

In Table 2, using the data of the four groups, the average of forecasting errors of 4 groups by GRNN model is 0.012 208, while the average of forecasting errors of 4 groups by the multi-GRNN model with a gating network is 0.002 659.

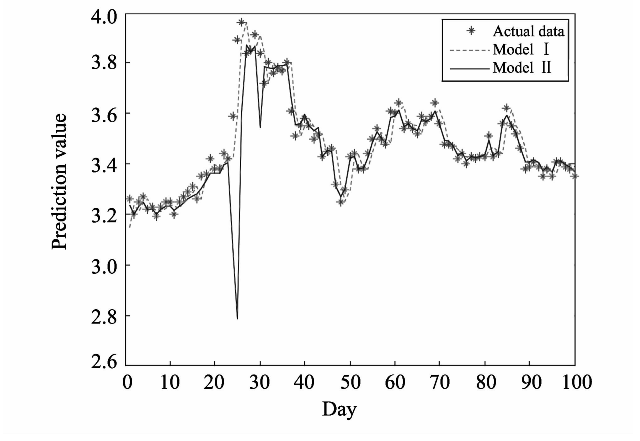

Fig. 5 The dongfeng automobile prediction curve

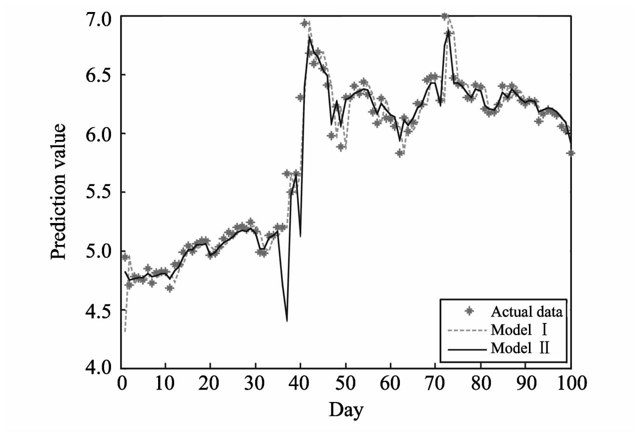

Fig. 6 The Baotou Steel prediction curve

According to the Figs. 3 – 6, GRNN model predicts for a short time, but the earlier prediction error is small and the later is big, so the final result is not very ideal.

On the contrary, the earlier prediction error of multi-GRNN model with a gating network looks bigger, but the later error is very small, so better expected results can be got. Comparing two model errors, we conclude that multi-GRNN model with a gating network has smaller error, and better results can be got.

4 Conclusion

This paper proposes GRNN model and multi-GRNN model with a gating network. Both models can achieve good effect in stock prediction. Gating network decreases error of only one GRNN neural network through softmax excitation function. Multi-GRNN with a gating network model works better in all four groups of data and has better accuracy, better stability and stronger generalization ability than the GRNN model.

References

[1] LIU Hai-yue, BAI Yan-ping. Analysis of ar model and neural network for forecasting stock prices. Journal of Mathematics in Practice and Theory, 2011, 41(4): 14-19.

[2] ZHOU Min, LI Shi-ling. Application of GRNN and uniform design to nonlinear system modeling. Computer Measurement & Control, 2007, 15(9): 1189-1191.

[3] LI Huan-qin, WAN Bai-wu. Application of modular wavelet neural networks in industries product quality control. Control and Decision, 2004, 19(3): 295-298.

[4] Specht D F. A general regression neural network. IEEE Trans. on Neural Network, 199(2): 568-576.

[5] Parinya D. Global stock index forecasting using multiple generalized regression neural networks with a gating network. Missouri: University of Missouri Rolla, 2001.

[6] Cong Shuang. Neural network theory and applications with Matlab toolboxes. China Science and Technology University Press, 1998.