Bulletin of Networking, Computing, Systems, and Software – www.bncss.org, ISSN 2186-5140
Volume 8, Number 1, pages 9–13, January 2019

A Multi-factor Approach for Stock Price Prediction by using Recurrent Neural Networks

Xu Zhang, Chen Li, Yasuhiko Morimoto
Department of Information Engineering
Hiroshima University
1-4-1 Kagamiyama, Higashi-Hiroshima, 739-8527, Japan

***Abstract*— Stock price prediction is a difficult type of time series predictive modeling problem. In time series forecasting, Autoregressive Integrated Moving Average(ARIMA) is one of the famous linear models. However real-world time series like the stock is rarely pure linearity. The conventional ARIMA method cannot model the non-linear function very well. Since the stock price forecast depends on many factors, generating a good prediction model is a huge challenge for the researchers. In recent years, Recurrent Neural network (RNN) has yielded immense success on time-series prediction which can learn arbitrary linear and non-linear function from the dataset. Long Short-Term Memory network (LSTM) is one of RNN which is usually used for solving gradient vanishing problem. In this paper, we propose an LSTM approach to predict stock price. Moreover, for better improving the accuracy of our model, we consider multi-factor which are relevant to stock price. In the experiments, we firstly compare our model with the conventional ARIMA method. Besides, we also demonstrate the performance of single-factor with the multi-factor model and different time periods.**

Keywords: Stock Price, Recurrent Neural Network, ARIMA.

I. INTRODUCTION

In the real world, a lot of tasks belong to time series forecasting tasks such as forecasting stock price, forecasting exchange rate, and forecasting weather. In general, researchers firstly analyze the past observations and get a model which could represent the relationship of time series. Finally, using this model for forecasting future data.

Stock price belongs to financial forecasting. In fact, there are lots of people care about of it since it related to our life. So in the past years, so many people are working on financial forecasting task. If the model is well, people can get a high profit. But forecasting tasks are not very easy because in real world there are linear features and nonlinear features influence on it. And in the real dataset, maybe it is not complete, or it has some unreal data [13].

Before neural network, there is forecasting method called Autoregressive Integrated Moving Average (ARIMA) [2] is very popular. The reason of why it can be wildly used is that it could capture the linear relationship in the dataset. In fact, ARIMA model involved three different types of model [11]. The first one is the Autoregressive model (AR), the second one is called Moving Average (MA) model, the last one is ARMA model which is combined with former two models. But the performance of ARIMA model for real-world time series is not very well. Because this model only could capture linear relationship. This is its major limitation. For our stock dataset, there are a lot of factors influence stock price. The single linear function cannot capture complex relationship. So using ARIMA model could solve some simple forecasting task which time series only include linear function.

To solve the limitation of ARIMA model, Artificial Neural Networks (ANNs) appear [18]. In the past ten years, researchers changed different types of ANN for time series forecasting. The major advantage of ANNs is that it could capture linear features and nonlinear features. So the performance of ANN for real-world forecasting is much better than ARIMA model. Another advantage of ANN is that it does not pre-assign function. ANN model is auto-suitable based on the features provided in the dataset. Using ANN could forecast only future short-term data. This is its major limitation. It cannot memory long-term information of lots of inputs. For our sequential information, ANN cannot make use of it.

Recurrent Neural Networks (RNNs) is different with ANNs. The structure of RNNs are more complex than ANNs. The improvement of RNNs is that it can memory long-term information. So it can process sequential data. Rnns is very suitable for time series forecasting tasks. The improvement of RNNs is that in hidden layer there has a hidden state, and using hidden state could memory former information. The advantage of conventional RNNs is that it can lead to gradient descent and gradient explode [10]. In other word, conventional RNNs cannot memory very long time information. Long Short-Term Memory (LSTM) [8] and the Gated Recurrent Unit(GRU) [3], are designed to catch more long-term memory so that it can be long-term memory inputs more easily. The different of LSTM and GRU is that the structure of GRU is simple than LSTM.

In this paper, for handling the sequential prediction task in the financial market field, we will use LSTM for stock price prediction. Firstly, we only use one single time series (high price) for training our model. Comparing its performance with traditional ARIMA model. For better improving our model, additional six factors (open price, close price, low price, volume, money, and change) are considered. Retraining the neural network and compared with a single factor model. Finally, we test prediction capacity of Recurrent Neural Networks with stock price dataset in weekly step, half monthly step, monthly step and 100 days step.

The rest of this paper organized as follows. Section Ⅱ we review related works for time series forecasting. Section Ⅲ presents the definition and notations used in RNNs. We explain the details of our proposed model in Section Ⅳ. Then, experimental results explained in Section V. After that, Section VI concludes the paper.

II. RELATED WORK

A. Conventional ARIMA model

ARIMA model is proposed by Box and Jenkins [2]. This model measures time series data over time. And this model usually used in statistic and econometrics areas. The following is the formula of ARIMA model.

$$x_t = \theta_0 + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + \varepsilon_t - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_q \varepsilon_{t-q} \quad (1)$$

In this formula, x_t represents the real time series value. ε_t represents the random error. There have another two parameters p and q. P is the corresponding to time series of real value and q is the corresponding to MA model. And ε_t represent the random errors. And the training time series have the mean of zero and a constant variance. We use the past observations and random error in this formula to forecast future data.

According to this formula, in fact ARIMA model has three types. One is AR model when q=0 in our formula. another is MA model when p=0 in our formula. And if p and q both are not equals to zero, it is ARMA model. Select real time dataset belongs to which model is an important task.

ARIMA model is not very well in modeling nonlinear relationship in time series. In this study, ARIMA modeling is implemented via the EViews9 system.

B. Time Series Forecasting using Conventional Neural Networks

We all know ARIMA only can capture linear functions. And ARIMA model is not good at real-world complex time series data. So one model must could capture linear and nonlinear functions exist in time series [6]. Artificial Neural Networks (ANNs) is the approach that could solve this problem. It can capture lots of nonlinear functions in dataset.

The motivation of ANN is from people's brain. Artificial Neural Network learn something by recognize example in real world. It different with conventional programming. The learning information is stored in weights which connect different layers. One simple ANN include input layer, hidden layer, and output layer.And every layer includes large of neurons. Input layer corresponding to time series inputs. In hidden layer, every hidden neuron has an activation function. And output layer corresponding to the result of ANN. Using ANN can solve many complex tasks such as image recognition, speech recognition, machine translation and so on.

In general, researchers usually use one hidden layer for our time series forecasting tasks [19]. We use following formula to train our neural network:

$$h_t = f(\beta_{0j} + \sum_{i=1}^{p} \beta_{ij} x_{t-i}) \quad (2)$$

$$y_t = \alpha_0 + \sum_{j=1}^{q} \alpha_j h_t \quad (3)$$

(y_t) represents the output of neural network. From x_{t-1} to x_{t-p} are our input data corresponding to input neurons. And $\alpha_j (j = 0,1,2,...,m)$ with $\beta_{ij} (i = 0,1,2...,n; j = 1,2,...,m)$ are the model's weights. In general, weights and bias set randomly by programmer at the beginning. n is the number of input neuron, and m is the number of hidden neuron. And f is activation function in every hidden neurons. The activation function can be a linear function or a nonlinear function. Sigmoid function is often used in artificial neural networks,

$$f(x) = \frac{1}{1+exp(-x)} \quad (4)$$

III. RECURRENT NEURAL NETWORKS

A. Conventional Recurrent Neural Network

In the past decade, Recurrent Neural Networks (RNNs) used in various tasks. Such as image recognition [16], machine translation [12], and speech recognition [7] and so on. This network not only can capture large of linear and nonlinear relationships in time series but also could process sequential dataset. One major different with ANNs is RNNs could memory long-time information. However ANN just memory short-term memory since the inputs in ANN are independent. So researchers make some improvements in the structure of ANNs. In fact RNNs include could unfold by time. After unfolding we can see that lots of simple neural network connected by hidden state. The information of hidden state from two sides. One is from input information at time t. And another is from hidden state at time t-1. So this is why RNNs could memory long-term information. And why RNNs could process sequential dataset. Look at figure1, the right figure is unfolded by time t from left figure. There has three layer. There are input layer, hidden layer, and output layer. $o_i and x_i$ are the output value and input value of RNN. h_i is the hidden state, it stores information of neural network. U, W, and V are weights in RNNs.

$$s_i = f(Ux_i + Ws_{i-1}) \quad (5)$$

f represent a linear or nonlinear function, In general using tanh, ReLU and sigmoid function.

The output of RNNs o_i at time step i is computed by following formula:

$$o_i = softmax(Vs_i) \quad (6)$$

B. Long Short-Term Memory

To solve the vanishing gradient and gradient explode problems, researchers proposed complex Long Short-Term memory (LSTM) [9] [5]. The structure of LSTM is more complicated than conventional RNNs. LSTM need more parameters in training process. In hidden layer of LSTM,

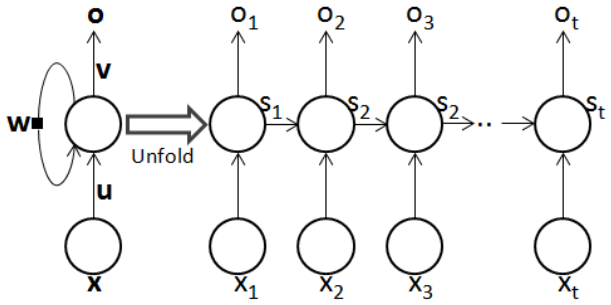

Fig. 1: Conventional RNN structure.

every hidden neuron has a memory cell. Memory cell stores the memory in neural network at time t. Hidden layer also has three different types gates. There are input gate, forget gate and output gate. Figure2 shows the detailed information of LSTM memory cell and three gates.

The function of these three gates is to manage the memory cell process the inputs information and former information. For example, if the input gate is not open, no inputs information can input to neural network. And if forget gate is not open, all the former information could be stored in memory cell. And output gate controls which part of information in memory cell can be output to next hidden state. This is the formal LSTM. In fact researchers use slightly different LSTM to their different tasks. The differences are very small.

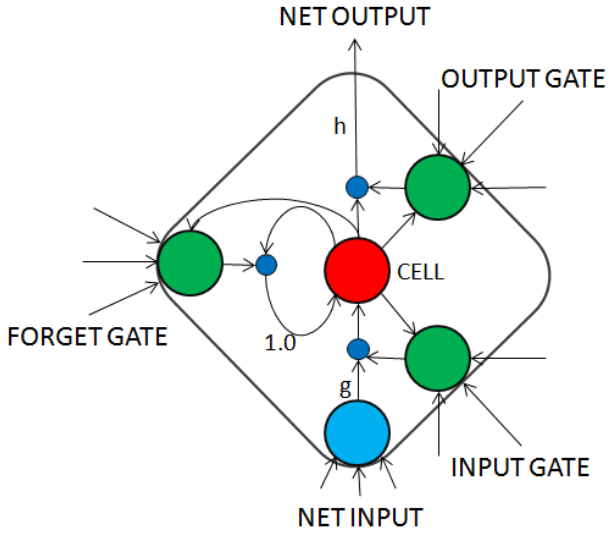

Fig. 2: Structure of LSTM memory cell. Green circle represent three different types gates. At first, input gate controls which part of input information should be input. g is the activation function. After g function, the input information stores in memory cell. The red circle represents memory cell. The forget gate connect hidden state at time t-1 and hidden state of time t. It controls which part of former information should be input to memory cell at time t. Finally, output gate controls which part of information in memory cell could be transfer to next hidden state.

C. Gated Recurrent Unit

Another Recurrent Neural Network is called Gated Recurrent Unit (GRU). GRU also could solve vanishing gradient and gradient explode problems. And GRU has ability to memory long-term former information. The structure of GRU is slightly different to LSTM. In hidden neuron, GRU only has two gates. There are reset gate and update gate. The function of reset gate is the same as input gate and forget gate in LSTM. GRU combine this two function into one gate. And the update gate controls how much previous information input to memory cell at time t. The advantage of GRU is it has fewer parameters than LSTM since the structure of GRU is simple than LSTM. So the training time of GRU is faster than LSTM. We can not say which RNN is better because it depends on different type of tasks. If you have large data the greater expressive power of LSTM may has a better choice.

IV. PROPOSED APPROACH

A. RNNs for time series prediction

By sorting according to timestamp, the input values can be represented as $[x_1, x_2, ..., x_{T-1}, x_T]$. For single-factor approach of stock high price prediction, x_T is the stock high price at time T corresponding one input neuron. For multi-factor approach, x_T includes 7 dimensions at time T corresponding 7 input neurons. Given a stock price sequence $x = [x_1, x_2, ..., x_{r-1}, x_r](1 \leq r \leq T)$, our model will give the output y.

B. Training Stage

During the training stage, as shown in figure3, we set a fixed size of time steps T for unfolding the network. We use $\{[x_1, x_2, ..., x_{T-1}]_j, [x_2, x_3, ..., x_T]_j\}_{j=1}^{N}$ as training data and train the model in a sequence-to-sequence manner. $[x_1, x_2, ..., x_{T-1}]_j$ is the input and $[x_2, x_3, ..., x_T]_j$ is the corresponding true output. Backpropagation-Through-Time (BPTT) [14] is used for propagating gradients of errors. Error is computed using loss function:

$$loss = \frac{1}{n}\sum_{i=1}^{n}(y_i - o_i)^2 \tag{7}$$

(where o_i is the predicted value and y_i is the true value). BPTT is a variant of backpropagation method used in training feedforward neural networks. Using BPTT, the error at each time step is backpropagated to previous time steps. i.e., BPTT applies backpropagation method to unfolded RNN.

C. Test stage

The goal of test stage is to predict the next element based on the information provided by a stock sequence with the fixed size. In test dataset, it contains 100 test data to compare with prediction data and compute its error using the MAD method. We trained two types of models, one is only using a single factor with one input neuron, and another model trained by seven factors with seven input neurons. The structure of this two models shown as figure4.

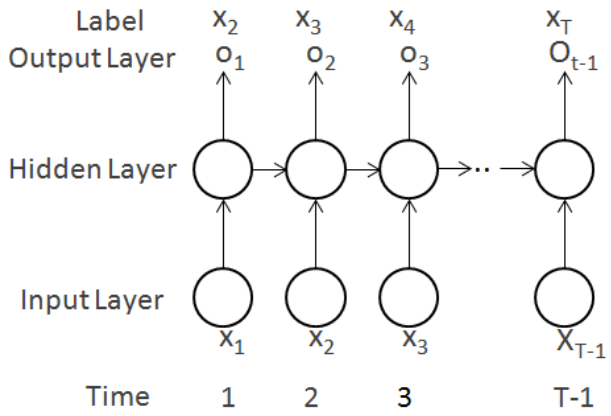

Fig. 3: Training process

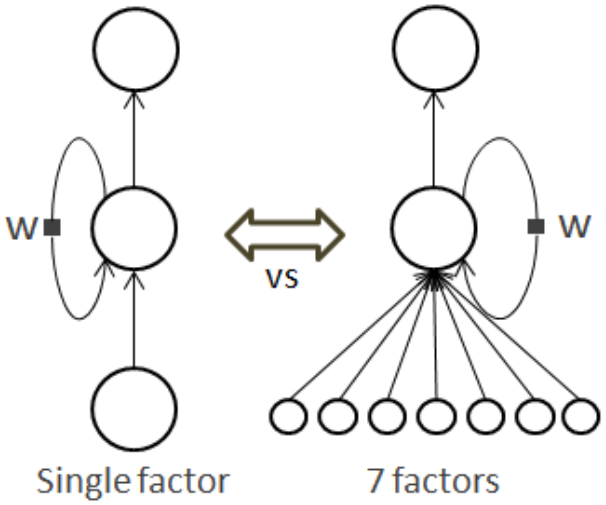

Fig. 4: Single factor model and multi-factor model

V. EXPERIMENT

A. Data set

The stock dataset we consider contains stock price from 1990 to 2015, giving a total of 6109 observations. Each record includes ten dimensions including stock index code, date, open price, close price, low price, high price, volume, money, change, and label. Usually, stock market data looks like on figure4, where is displayed high prices of every day. Dataset is split into two subsets for training and testing usages. Training subset has 6009 records. And testing subset has 100 records.

B. Data processing

Before the experiment, first, we have to normalize our input data. The input vectors of the training data are normalized in normalization function that all the feature have zero-mean and unitary variance. Usually, data are scaled to be in the range $[-1, 1]$. Here we use Z-score normalization:

$$x_{new} = \frac{x_{old} - mean}{\sqrt{var}} \tag{8}$$

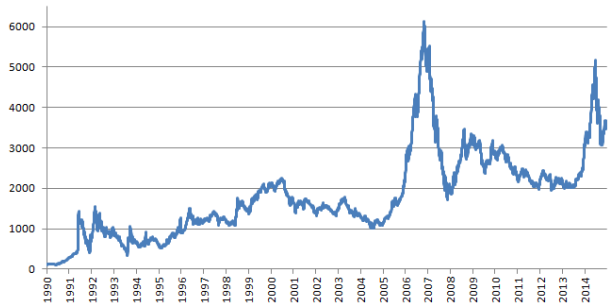

Fig. 5: Stock dataset from 1990 to 2015

The single factor model and multi-factor model contain one hidden layer including ten hidden units and only have one output neuron. We compared the performance of Gated Recurrent Unit (GRU) [4] cells and Long Short Memory (LSTM) [9] cells. Since they both address the vanishing gradient problem and able to learn for a long period. Our model is trained with 20-time steps using BPTT. The model is implemented using Tensorflow [1], which is a deep learning platform developed by Google. A GeForce GTX 1080Ti GPU is used to boost the training speed.

C. Experimental Results

We compared prediction performance between conventional ARIMA model and Recurrent Neural Networks model. Evaluations are executed using MAD measurement. Table1 shows the evaluation results of five models in 100 test data. Results show that for long-term forecasting (100 days) neural network is much better than ARIMA model. And LSTM with seven factors of stock's high price is better than GRU model.

TABLE I:
COMEPARISON OF FIVE PREDICTION METHODS.

Method	*MAD*
ARIMA	546.8176
GRU(7 factors)	311.8716
LSTM(7 factors)	76.8139

TABLE II:
FORECAST WITH DIFFERENT TIME PERIOD USING LSTM(7 factors).

Time period	*MAD*
7 days	31.4323
14 days	46.4498
30 days	38.6245
100 days	76.8139

From figure6, we observe that our multi-factor Recurrent neural network model outperforms all another model. This is an evidence showing that only one factor cannot catch relationship between time series data. And add factors of the stock price is one of a method to improve performance prediction. We all know that LSTM and GRU are both can remember long-term information. So we check the loss curve show that why the performance of GRU is worse than LSTM

in our experiment. Figure7 compared the change of loss value between LSTM model and GRU model. We can see that after 100 iterations finally, the loss value of LSTM model is lower than GRU model. So LSTM model's prediction ability is better than GRU model.

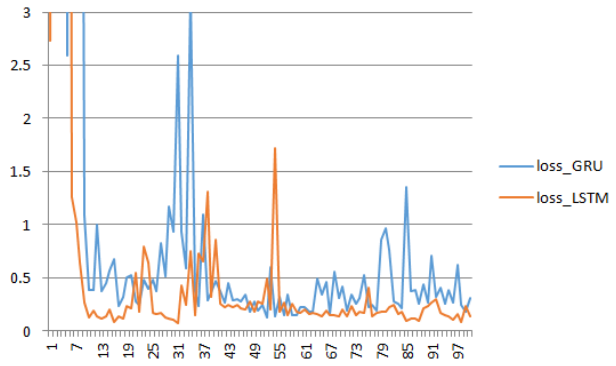

Fig. 6: Loss value changes of LSTM model and GRU model

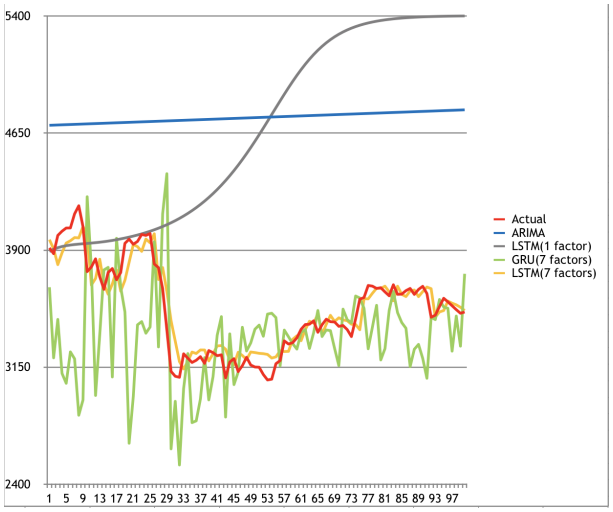

Fig. 7: Compared prediction results of different models

And this paper also tests prediction capacity of LSTM with seven factors in weekly steps, half monthly steps monthly steps and 100 days steps. From table2 we can see that as time goes on, accuracy decreased significantly.

VI. CONCLUSIONS

In this paper, we propose a multi-factor approach for stock price prediction. For complex problems that have both linear and nonlinear correlation structures. Only using linear method ARIMA cannot capture nonlinear relationship. For Recurrent neural networks, increase related factors of the stock price can significantly improve learning ability. We also compare prediction ability between GRU and LSTM in the financial area, based on experiments, the performance of LSTM is better than GRU. Therefore, we conclude that our method is competitive with previously proposed methods for handling time series forecasting.

REFERENCES

[1] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265283, 2016.

[2] G.E.P Box, G. Jenkins, Time Series Analysis, Forecasting and Control, Holden-Day, San Francisco, CA, 1970.

[3] Cho, Kyunghyun, Van Merrienboer, Bart, Gulcehre, Caglar, Bougares, Gethi, Schwenk, Holger, and Bengio, Yoshua. Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078, 2014.

[4] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine *arXiv preprint arXiv*:1406.1078, 2014.

[5] F. Gers, N. Schraudolph, and J. Schmidhuber,"Learning precise timing with LSTM recurrent networks," Journal of *Machine Learning Research*, vol.3,pp. 115-143,2002.

[6] J.G. De Gooijer, K. Kumar, Some recent developments in non-linear time series modeling, testing, and forecasting, Int. J. Forecasting 8 (1992) 135-156

[7] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neuron networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 19(6):82-97,2012.

[8] Hochreiter, S., and Schmidhuber, J. Long Short-Term Memory. *Neural Computation*, 9(8):1735-1780, 1997.

[9] S. Hochreiter and J.Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735-1780, 1997.

[10] Hochreiter, S., Bengio, Y., Frasconi, P., and Schmidhuber, J. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In Kremer, S.C. and Kolen,J.F. (eds.), A Field Guide to Dynamical Recurrent Neural Networks. IEEE Press, 2001.

[11] E.D. McKenzie, General exponential smoothing and the equivalent ARMA process, J. Forecasting 3 (1984) 333-344.

[12] Tomas Mikolov, Martin Karafiat, Lukas Burget, Jan Cernocky, and Sanjeev Khudanput. Recurrent neural network based language model. In *Interspeech*, volumn 2, page3, 2010.

[13] S. Thawornwong and D. Enke, Forecasting stock returns with artificial neural Networks. In G. p. Zhang (ed) Neural Networks in Business Forecasting (Chapter 3, pp. 47-49). Idea Group Publishing, 2004

[14] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceeding of the IEEE*, 78(10):1550-1560, 1990.

[15] H. wold, A Study in the Analysis of Stationary Time Series, Almgrist, Wiksell, Stockholm, 1938.

[16] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption caption generation with visual attention. In *International Conference on Machine Learning,* pages 2048-2057,2015.

[17] G.U. Yule, Why do we sometimes get nonsense-correlations between time series? A study in sampling and the nature of time series, J.R. Statist. Soc.89 (1926) 1-64.

[18] G. Zhang, E.B Patuwo, M.Y. Hu, A simulation study of artificial neural networks the state of the art, Int. J. Forecasting 14 (1988) 35-62.

[19] G. Zhang, E.B. Patuwo, M.Y. Hu, Forecasting with artificial neural networks: the state of the art, Int. J. Forecasting 14 (1998) 35-62.