Neural network modeling for stock movement prediction
A state of the art

Olivier Coupelon
Master's Degree in Computer Science
Blaise Pascal University
olivier.coupelon@wanadoo.fr

***Abstract*— This paper proposes an overview of the modeling process of artificial neural networks (ANN) in stock movement prediction. A step-by-step procedure based on the most commonly used methods is presented, showing the difficulties encountered when modeling such neural networks. Other techniques are also mentioned as neural networks are not the only tools used to predict stock movements.**

***Index Terms*— Neural network, Financial, Stock movement, Modeling, Simulation, State of the art.**

I. Introduction

STOCK movement prediction has been at focus for years since it can yield significant profits. Fundamental and technical analysis were the first two methods used to forecast stock prices. A new method appeared more recently, the technological analysis, where computers are used as a tool to predict the stock movements.

Technological analysis tries to model and simulate as accurately as possible the behavior of the stock exchanges, by different techniques which I will discuss in the following.

The first market hypothesis is that stock prices follow a random walk. However, many researchers and economists were able to extract rules for predicting the stock prices evolution, showing that some basic observations significantly increase the quality of their predictions ([1] and [2]).

First I will show the different available forecasting techniques, then I will focus on neural networks as they tend to be the most widely used and studied tool for forecasting stock movements.

II. Forecasting techniques

Many tools are used to try to model the behavior of the stock movements, and many researches have been conducted with this goal in mind. However, many of them were done by financial corporations, who keep them confidential as they are used to guide their financial investments. Another point to consider when dealing with researches is their correctness. It has been shown that for many reasons, in financial forecasting using neural networks most of the search results can't be used as is, as researchers did not fully investigate the potential of their solutions (see [3] and [4]).

Among all the available techniques, here are the most commonly used:

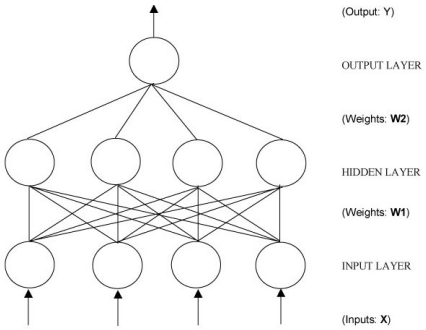

Fig. 1. A Typical Feed-forward Neural Network (*Source: [11]*)

- Genetic algorithms which help in finding optimal parameters for technical indicators by making them evolve, by combinations and mutations, starting with a population of a given model having established different parameters ([5], [6] and [7]).
- Fuzzy logic controllers, which are for instance used in combination of artificial neural networks and genetic algorithm in [8]. Fuzzy neural network are also studied, and sometimes demonstrate good performances, as in [9].
- Genetic programming, which make different investment strategies evolve and keeps only the most adapted ones ([10]).
- Hidden Markov Models (HMM) have recently been used for financial forecasting. The objective is to determine the parameters of a Markov process that would fit the stock movements.
- Neural networks which are the most commonly used techniques. I will introduce them in the next section.

III. Neural Networks concept

Neural networks, also called artificial or simulated neural network, are composed by computing units (artificial neuron) interconnected so that each neuron can send and receive signals to or from others (See figure 1). Neural networks are a good answer for the modeling of distributed nonlinear systems. As their design is based on the human brain, they were made to acquire knowledge through learning.

The process of learning for a neural network consists in adjusting the weights of each of its nodes considering the input

of the neural network and its expected output. This process requires the availability of a set of input data, stock quotes in our case.

IV. Neural Networks for stock movement prediction

[12] and [3] both suggest a step-by-step approach for modeling neural network for financial forecasting, which I will use as a guideline to present the most commonly used techniques for building those models. Although they are presented in a chronological way considering the development of a neural network, it is agreed that they should be revisited when significant changes are made to increase the performance of the neural network.

A. Selection of Input variables

The selection of input variables is fundamental to accurately forecast the stock movements. It primarily depends on a clear understanding of the economical background of the stock price to forecast.

The first value that comes to mind is the evolution of the stock price to forecast over time. Although it's an essential variable, it is often not used directly as a raw variable (see part IV-B and the following).

The economical context of the studied stock quotes has to be carefully analysed, for instance raw material cost, which if taken into account by the neural network can help the forecasting process. These types of input variables are sometimes called external indicators [13], and are commonly used if available.

Once the raw data have been chosen, a set of indicators based on those values might be developed. Upon these, at least five indicators are commonly used as input for neural networks (Source: *investopedia.com*):

- *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.
- *Money Flow Index (MFI):* This one measures the strength of money in and out of a security.
- *Moving Average (MA)* This indicator returns the moving average of a field over a given period of time.
- *Stochastic Oscillator (SO):* This function compares a security's closing price to its price range over a given time period.
- *Moving Average Convergence/Divergence (MACD):* This function calculates difference between a short and a long term moving average for a field.

At this step, depending on the amount of resources available, it can be decided to incorporate more input variable than might seem necessary, as part IV-C will help in keeping only the most sensitive variables.

B. Data Preprocessing

Before the data is analyzed by the neural network, it has to be preprocessed in order to increase the accuracy of the output and to facilitate the learning process of the neural network. This is a critical operation since neural networks are pattern matchers, thus the way data are represented directly influence their behavior.

In case of a day without trading, the missing data might be filled up ([3]) either by interpolating the missing values, or by creating a specific neural network to rebuild those values. Depending on the importance of those data and the amount of missing values, it can also be decided that they should simply be dropped.

Another important preprocessing operation is data normalization, which is useful to remove noisy data by creating more uniform distribution if needed. Indeed, such distribution tend to help the neural network forecasting process. Citing [14], [11] explains there are at least four methods for data normalization:

1) Along channel normalization: This method normalizes each input variable individually.
2) Across channel normalization: Normalization is done across all the input elements in a data pattern.
3) Mixed channel normalization: As the name suggests. This method uses some kind of combinations of along and across normalization.
4) External normalization: Data is normalized into a specific range.

The input data is usually scaled between -1 and 1, sometimes between 0 and 1. This is important so that the transfer function (IV-D.5) can be used. The two most common techniques used to scale the data are linear and standard/mean scaling. While linear scaling strictly preserves the uniformity of the distribution, mean scaling helps in creating a more uniform distribution, which can be desired if most of the extreme values of the original distribution are considered to be noisy.

Although the necessity of preprocessing data is not agreed by every researcher, there have been significant demonstrations showing that it can greatly improve the forecasting performance, as showed in [15].

C. Sensitivity Analysis

The sensitivity analysis is the process which determines whether an input variable influence the output of the neural network or not. There are several ways to find this. The most common approach is to run the neural network with and without each input variable, and to check the variations of the input. If there are no noticeable changes, it surely means that the input variable can be omitted. This generally happens when a variable is highly correlated with another. Because those variables where chosen for being related to the current stock to forecast, such a situation is likely to happen.

As demonstrated in [16], one might expect a great improvement of the performance of the neural network by doing such a sensitivity analysis.

D. Model Construction

Three types of neural networks are commonly used by economists and researchers in stock market prediction ([16]):

- *MultiLayer Perceptrons (MLP):* Those are layered feed-forward networks.
- *Generalized Feed-forward Networks (GFN):* This is a generalization of MLP which can jump over one or more layer.
- *Radial Basis Function (RBF):* Hybrid networks containing a single hidden layer of processing elements which uses Gaussian transfer functions rather than the standard sigmoid functions.

The type of neural network to use should be decided by determining the most adapted architecture, which is often done by developing and comparing every possibility, and by keeping only the best ones. The following will discuss the most important points to pay attention to when modeling a neural network for stock movements forecasting.

1) Number of input neurons: This number is easy to determine when the previously defined steps have been done, because each input variable will be treated by a single input neuron. This is why IV-C is clearly an important step directly increasing the processing speed of the neural network.

2) Number of hidden layers: The hidden layers are composed by the set of all neurons between the input and the output neurons. The number of hidden layer that should be used can not be clearly defined as it really depends on the amount of input neurons and the properties of the data. Formulas were developed which tried to take into account those parameters, but due to the nature of the stock movements, it can't be predicted easily ([17]).

However, as stated in [7], it is commonly admitted that one or two hidden layers are enough, and that increasing the amount of those layers also increases the danger of overfitting and the computation time.

3) Number of hidden neurons: The most common technique used today to determine the appropriate number of hidden neuron to include in the model is experimentation. It's a part of the training phase of the neural network development and might require a lot of computations. The only rule to keep in mind while selecting the most appropriate number of hidden neurons is to always select the network that performs best on the testing set with the least number of hidden neurons ([12]).

4) Number of output neurons: Most neural network applications use only one output neuron for both one-step-ahead and multi-step-ahead forecasting [11]. However, some studies tried different approaches, as in [18] where multiple output neurons are shown to be of interest.

5) Transfer functions: The transfer function is the formula used to determine the output of a processing neuron. At least five different transfer functions are in use, see figure 2.

Selecting the best transfer function is once again done by a process of trial and error. The main selection criteria here is their ability to make the learning phase faster and to increase the neural network accuracy.

Neural network that implements heterogeneous transfer function have been studied in [19], but despite the fact that they seem to improve the performance of neural networks, no commercial simulators are currently fully implementing them.

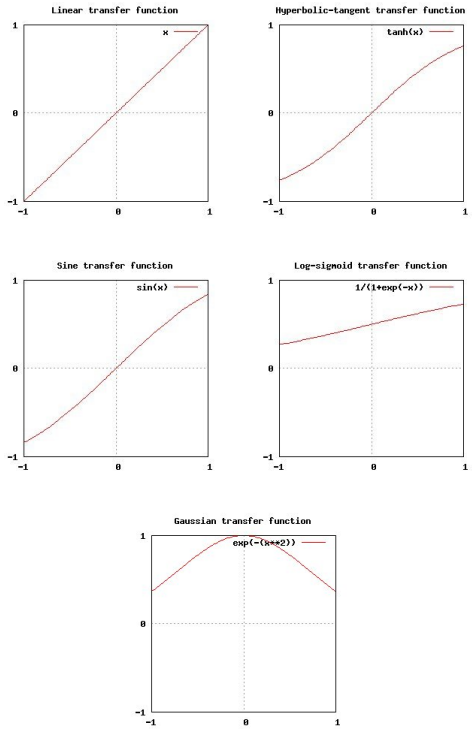

Fig. 2. Transfer functions representation

E. Training algorithm

The process of training the neural network involves iteratively presenting it the input data so that it is calibrated and can be used later as a forecasting tool. The objective of the training is to minimize a defined error function, which implies that the neural network fit the input data, given the expected results as outputs.

There are several kind of error functions used in financial forecasting, namely:

- Mean Absolute Deviation (MAD): $\frac{1}{n}\sum_{i=1}^{n}|x_i - \bar{x}|$
- Mean Squared Error (MSE): $\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$
- Mean Absolute Percentage Error (MAPE): $\frac{1}{n}\sum_{i=1}^{n}|\frac{x_i - \bar{x}}{x_i}|$

where n is the number of error terms.

Using those error functions to adjust the weights of the hidden neuron has for long been an insurmountable problem, but [20] introduced the so called BackPropagation Networks (BPN), which consists in neural networks able to issue a backward propagation of errors, even to the hidden neurons. Whenever an error is detected, the weights of the network are slightly modified in order to lower it. This process is generally done using a gradient descent algorithm which adjusts the weights to move down the steepest slope of the error surface.

In the process of backpropagation, two degrees of liberty are available to the modeler:

- The *learning rate* which determines how fast the neural network learns the pattern in the training data. This value must be carefully chosen as a too small one will make the learning process slow, but a large one might lead to

divergence. One way to dynamically modify the learning rate is to increase it as long as the gradient keeps pointing in the same direction, but lowering it when it changes ([21]).

- A *momentum term* which influence the way past weight changes affect current ones. It helps in preventing the gradient method to get stuck into a local minimum. Once again, this value must be chosen carefully, and should be determined by experimentation. It has to be noted that although the use of a momentum can be omitted, it greatly improves the neural network performances and though is commonly used.

The backpropagation process can suffer of three problems:

- Reaching the global minimum is not guaranteed by this method, as there can be several local minima from which the algorithm might possibly not escape.
- This method might lead to overfitting, as it insists on each input data. This point is actually criticized, as it is argued that overfitting is more a problem of having too many neurons in the network ([12]).
- There is no know configuration for this algorithm which enables it to always find the best solution, so a process of trial and error must be developed. This is generally done by redoing the descent a sufficient number of times, ensuring that the reached minimum is really the global one (Although it can't be proved). The number of iteration needed depends on the complexity of the network, but [12] reports that convergence it generally obtained from 85 to 5000 iterations.

Knowing those problems helps in defeating them, and several solutions where advanced to keep backpropagation as the algorithm to use [4]. Nonetheless, other training methods are available ([11] and [22]), for instance conjugate gradient descent, quasi-Newton algorithm, and Levenberg-Marquardt algorithm, but their use are still marginal, despite the fact that they can achieve better performances.

Another training method that is getting more and more used is based on a genetic approach. The principle is that the weigths of the neural network are not adjusted by a backpropagation algorithm but with a genetic algorithm. At first, a population consisting of randomly generated sets of weigths is created. An evolution algorithm is then applied to the population to generate and keep only the most appropriate set of weights. Such a solution generally prevents from falling into local minima and shows good performances. Those types of neural networks are generally called Genetic Algorithm Neural Networks (GANN). [23] gives a good example of their use in financial forecasting.

V. Conclusion

This state of the art tried to sum up the actual tendencies observed in the development of neural networks as tools to forecast stock movement. The modeling of neural networks is made of a lot of experimentations, which are necessary to create and calibrate the network in order to represent correctly the data it is associated with.

Following the steps presented in this paper while modeling a new neural network ensures that most of the problems that can be encountered have been dealt with, which should lead to a good financial forecast.

Researches on this domain keep on, and novel approaches are constantly imagined. Although genetic algorithms are not the most commonly implemented training algorithms, they have shown good performances and specialists are getting more and more involved in there studies. Hidden Markov Models, either alone or associated with other methods [24] are also obtaining good results and could be considered as an alternative to neural networks.

References

[1] R. Gençay. (1997) Linear, non-linear and essential foreign exchange rate prediction with simple technical trading rules. [Online]. Available: http://www.sfu.ca/~rgencay/jarticles/jie-technical.pdf

[2] ——. (1998) The predictability of security returns with simple technical trading rules. [Online]. Available: http://www.sfu.ca/~rgencay/jarticles/jef-technical.pdf

[3] J. Yao and C. L. Tan. (2001) Guidelines for financial forecasting with neural networks. [Online]. Available: http://www2.cs.uregina.ca/~jtyao/Papers/guide_iconip01.pdf

[4] M. Adya and F. Collopy. (1998) How effective are neural networks at forecasting and prediction? a review and evaluation. [Online]. Available: http://collopy.case.edu/researchArticles/neuralnets.pdf

[5] G. Armano, M. Marchesi, and A. Murru. (2005) A hybrid genetic-neural architecture for stock indexes forecasting. [Online]. Available: http://dx.doi.org/10.1016/j.ins.2003.03.023

[6] S. Chen and C. Liao. (2005) Agent-based computational modeling of the stock price-volume relation. [Online]. Available: http://dx.doi.org/10.1016/j.ins.2003.03.026

[7] S.-H. Chen. (2007) Computationally intelligent agents in economics and finance. [Online]. Available: http://dx.doi.org/10.1016/j.ins.2006.08.001

[8] N. Gradojevic. (2006) Non-linear, hybrid exchange rate modeling and trading profitability in the foreign exchange market. In Press, Corrected Proof. [Online]. Available: http://dx.doi.org/10.1016/j.jedc.2005.12.002

[9] P.-C. Chang and C.-H. Liu. (2006) A tsk type fuzzy rule based system for stock price prediction. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2006.08.020

[10] T. Yu, S.-H. Chen, and T.-W. Kuo. (2004) A genetic programming approach to model international short-term capital flow. [Online]. Available: http://www.aiecon.org/staff/shc/pdf/CH02.pdf

[11] G. P. Zhang, Ed., *Neural Networks in Business Forecasting*. Hershey, PA, USA: Idea Group Publishing, 2003. [Online]. Available: http://books.google.com/books?id=3cukH5eMpMgC

[12] I. Kaastra and M. Boyd. (1996) Designing a neural network for forecasting financial and economic time series. [Online]. Available: http://www.geocities.com/francorbusetti/KaastraArticle.pdf

[13] N. O'Connor and M. G. Madden. (2006) A neural network approach to predicting stock exchange movements using external factors. [Online]. Available: http://dx.doi.org/10.1016/j.knosys.2005.11.015

[14] E. M. Azoff and E. M. Azoff, *Neural Network Time Series Forecasting of Financial Markets*. New York, NY, USA: John Wiley & Sons, Inc., 1994.

[15] L. Yu, S. Wang, and K. K. Lai. (2006) An integrated data preparation scheme for neural network data analysis. Piscataway, NJ, USA. [Online]. Available: http://madis1.iss.ac.cn/madis.files/pub-papers/2006/2.pdf

[16] G. Weckman and R. Agarwala. (2003) Identifying relative contribution of selected technical indicators in stock market prediction. [Online]. Available: http://www.engineer.tamuk.edu/departments/ieen/faculty/ranjeet/Publications/IERC-Sens-2003-Upd.pdf

[17] P. M. Tsang, P. Kwok, S. Choy, R. Kwan, S. Ng, J. Mak, J. Tsang, K. Koong, and T.-L. Wong. (2007) Design and implementation of nn5 for hong kong stock price forecasting. [Online]. Available: http://dx.doi.org/10.1016/j.engappai.2006.10.002

[18] G. P. Zhang, B. Patuwo, and M. Hu. (1998) Forecasting with artificial neural networks: The state of the art. [Online]. Available: http://www.psiconet.net/neurogest/papers/state_of_art_forecast_nn.pdf

[19] W. Duch and N. Jankowski. (2001) Transfer functions: hidden possibilities for better neural networks. [Online]. Available: http://www.dice.ucl.ac.be/Proceedings/esann/esannpdf/es2001-400.pdf

[20] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," pp. 318–362, 1986.
[21] R. A. Jacobs, "Increased rates of convergence through learning rate adaptation," Amherst, MA, USA, Tech. Rep., 1987. [Online]. Available: http://www.bcs.rochester.edu/people/robbie/jacobs.nn88.pdf
[22] A. Abraham, "Meta-learning evolutionary artificial neural networks," *NETHERLANDS*, vol. 56c, p. 1, 2004. [Online]. Available: http://www.citebase.org/abstract?id=oai:arXiv.org:cs/0405024
[23] K. jae Kim. (2006) Artificial neural networks with evolutionary instance selection for financial forecasting. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2005.10.007
[24] M. R. Hassan, B. Nath, and M. Kirley. (2006) A fusion model of hmm, ann and ga for stock market forecasting. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2006.04.007