The Design of an Agent-Based System for Capital Markets

Marius PETRESCU[1]*, Mădălina CUC[2], Ionica ONCIOIU[3], Anca-Gabriela PETRESCU[4], Florentina-Raluca BÎLCAN[4], Mihai PETRESCU[4]

[1] Romanian Academy of Scientists, 54 Independence Splendor, 030167, Bucharest, Romania
profdrmpetrescu@yahoo.com (**Corresponding author*)

[2] Mihai Viteazul National Intelligence Academy, Odaii 20-22, Bucharest, 075100, Romania
madalina.cuc@animv.ro

[3] Titu Maiorescu University, 189 Calea Vacaresti Street, Bucharest, 040051, Romania
ionica.oncioiu@prof.utm.ro

4 Valahia University of Targoviste, 2 Carol I Blvd, Targoviste, 130024, Romania
anca.petrescu@valahia.ro; raluca.bilcan@valahia.ro; mihai_tina@yahoo.com

Abstract: The trading risk management implies analysing several types of risks: capital, market, liquidity, insolvency, business, credit, operational or financial risk. Trading models and techniques must be seen as tools that can provide an informed manager with useful insights, so they are indispensable in an increasingly integrated and sophisticated market. The main aim of this study is the conceptualization of a generic intelligent agent, based on a neural network and an agent system, applicable to a trading system of the listed companies. The results show that this model can contribute to reducing the transactional risk on the capital market and can offer solutions to improve the managerial decisions.

Keywords: Artificial neural networks, Multi-agent systems, Transaction databases, Capital markets, Validation.

1. Introduction

The development of the financial environment generates higher and more expensive risks in terms of coverage (Georgescu, 2011). In specialised literature, the computerization of the trading risk management in the decision-making process is the main key in vitalizing the strategies and achieving the objectives proposed by an organization (Sun et al., 2018; Dodd & Gilbert, 2016).

Developing a smart system of agents collaborating, transferring data and making decisions to minimize these risks can ensure a more realistic perception of the value of a portfolio and can underpin decisions about hedging strategies (Liagkouras, 2019; Tudor, 2008). Of course, the operating environment for any analyst or financial manager is probabilistic, and the results cannot be better than the input data and assumptions that have been built.

The concept of ”agent” appeared for the first time during the 1970s as a result of the studies in the field of Distributed Artificial Intelligence (DAI). Carl Hewitt proposed the model of an interactive and contestant object, able to perform tasks, which he called ”actor” (Haykin, 2009). The object had capsulated in itself several internal statuses and was capable of coordinating its actions with the ones of other similar objects by means of an email address. In time, the initial concept extended and diversified, becoming more and more complex.

Regarding the present approach, a definition suitable to the purpose of this research would be the following: the agent is a software entity able to perform autonomously or almost autonomously specific tasks in changeable environments.

Providing agents with the learning ability is useful in problem solving, in directing certain complex actions or in the control of the systems using dynamic environments (Cernazanu, 2008). Through learning, the agent becomes independent of the initial restrictions established by creation, subject to change.

This study is generated by the need to perfect the decisional support for the improvement of the managerial decisions by reducing the trading risk of the own assets in companies. By reducing the trading risk, the capital risk and the financial one are implicitly reduced as well as other components of the market and liquidity risk. The components of operational, credit, business or insolvency risk are also more or less reduced.

The advantages of the system thus built are given by the modularity, the flexibility with which the system’s architecture can be modified, the possibility to quickly establish new connections and to easily add new levels to the system. The following characteristics of the intelligent agent-based system have been pursued and carried out: mobility, platform dependence, autonomy and tasks delegation, communication ability,

https://doi.org/10.24846/v29i3y202003

coordination, reactivity and responsibility and prediction capacity.

The data sources used are taken from the databases of the Bucharest Stock Exchange (BSE) through the Arena Gateway platform offered by BSE. For reasons related to data security, the connection is made through VPN hardware which enables for the safe bidirectional data transfer. The manner of data processing and analysing is carried out through the training and prediction of the neural network and through the VaR (Value-at-Risk) simulation for the company issuer.

The article is organized as follows: section 2 gives the parameters and features of the designed agent system, section 3 contains the architecture of the designed agent system and its functionalities, section 4 contains the model validation and the final section contains the conclusion.

2. The Parameters and Features of the Designed Agent System

An agent can learn by experience (Rehar et al., 2017). Experience means the perception of the environment's evolution in the aftermath of the actions performed by the agent in various situations and of the interactions with other agents (Gil et al., 2012). An agent learns a concept if, based on a learning algorithm, it can create a representation of the concept (El Hamidi et al., 2019). Learning must not be achieved with high resources consumption.

The architecture of a learning agent described by (Zhu & Kwong, 2010) is under the form of an oriented graph. The nodes are the components of the agent and the arcs represent the information between the components. The orientation of the arcs is given by the manner in which the components interact. Each component has its own processing and control units which can operate in parallel.

The learning module will subsequently determine the improvement of the agent's behaviour (Wang et al., 2016). The pursuance of the actions performed by an agent is done by the critical module, which, based on the observations, generates the feedback. This component has a performance standard that must be set. In order to generate new experiences and to see whether the performance module performs its tasks, a component named "problem generator" is used.

On the other hand, multi-agent systems are systems consisting of groups of agents acting for the achievement of a common purpose (Wang, 2003). In order to do this, the agents making up a system of agents interact and work together for the achievement of the common task.

The shaping using intelligent agents is an approach specific to Web level 4.0, leading to the automation of the processes and, finally, to the automation of the systems based on them. This type of shaping is completed with the rest of the types of agents that do not need a training stage, being recommended a hybrid approach. Most of the classical artificial intelligence systems are static, their architecture is predefined, in contrast with the architecture of the multi-agent systems that is time-variable (Nakashima et al., 2005; Harik et al., 2017).

Another advantage of the multi-agent systems is that they are characterized by flexibility, scalability, and efficiency (Weidmann & Teuber, 2009). The agent systems can operate with decentralized and asynchronously processed data, each agent acting autonomously and having limited computing abilities. The agent systems' features entail the exchange of information among the entities making up the system, therefore the cooperation, coordination and negotiation are absolutely required for the achievement of the whole system purposes.

The informatics products used are Matlab, RapidMiner and JADE platform for the agents' development. The Matlab package is used for the creation of the neural network for price prediction, the training of the network being carried out through RapidMiner just as the validation, testing and prediction. The data are graphically exhibited by means of Matlab. The construction of agents on the agents' platform is carried out on the JADE platform.

In the present study, the agents are equipped with a predictive model and, supplementary, they benefit of generic modules which are able to interpret the XML language describing the model which they can perform by making predictions; these agents are dynamically "requalified" due to the component RapidMiner, able to make up such models, starting from the database.

The integrated system thus obtained is flexible and quite generic, enabling the automation of the (intelligent) predictive agents' instantiation, followed by their implementation in the

decisional processes as a decision support (Filip, 2014). Such an agent can auto-modify its predictive module dependent on the database it works on, because the set of instructions defining the model is an entity exterior to it (an input XML formatted). It will interface with this entity by means of the interpreter and, maybe afterwards, it will be able to use it in order to fulfil its basic function: the prediction.

3. Architecture of the Designed Agent System

The architecture of the designed and implemented multi-agent system is schematically shown in Figure 1 where the companies analysed have been noted with Issuer 1, ..., Issuer 5.

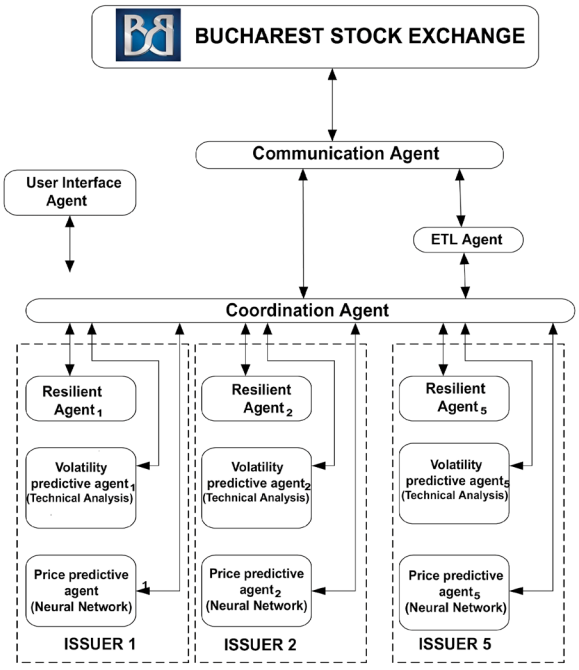

Figure 1. Multi-agent system architecture (acknowledgement to Mădălina Cuc)

The communication agent is the agent performing the connection with the Bucharest Stock Exchange. For reasons of data security, the connection is achieved through VPN hardware which enables the bidirectional safety data transfer.

The communication agent knows the communication protocols with the BSE and performs a filtering of the information by reaching it or rejecting the not interesting data.

For each order from every issuer's Order Book within the presented algorithm the following information is provided as shown in Table 1.

As a general rule, after the communication agent introduces the initial data which make up the summary for every issuer (Level 1 data) and the issuer's Order Book (Level 2 data), it receives from BSE only the data that modify the initial situation. For Level 1 it will receive the code of the field that is modified and the new value, and for Level 2, the position of the order to be introduced or modified and its values. In case of order annulment, the communication agent receives only the position of the order that will disappear from the list.

The endogenous variable considered is the share price (daily closing) and the exogenous variable considered is the trading volume.

Notice should be made that the system presented before is not enough is case the communication with BSE is interrupted due to various reasons (power failure, internet disconnection etc.). In this case, although the system receives a new updated Market picture after the failure, the data from the time when it has been disconnected from BSE are not accessible. In this situation, the communication agent requires and receives from the Extract-Transform-Load (ETL) agent the time of reception of the last valid data from BSE and sends a message to BSE requesting data about the transactions performed and the modified orders from the period the communication was cut off. Once it receives the requested data, it transfers them to the agent system so that it can correctly recalibrate itself. After a system failure, the aggregate time of recalibration does not exceed, on average, a second.

The testing of the connection with BSE is carried out through two distinct signals: Heart Beat System and Heart Beat User. Heart Beat System is sent out by BSE system once every minute, and the Heart Beat User is generated by the communication agent once in 10 seconds. If at least one of these signals has a delay exceeding two seconds, an alert for the user is generated through the mediation of the coordinating agent with the interface agent.

In case the system monitors its own transaction orders placed on the market, the communication failure with BSE may have serious consequences. The distress solution consists in the user's manual withdrawal of all active orders. The information transferred for each transaction are present in Table 2.

Table 1. Information stored by BSE for a share trading order (acknowledgement to Mădălina Cuc)

Field	Description
mkt	Market
sym	Symbol
acc	ARENA client account
act	Account type (1=Client, 2=Financial, 3=House, 4=Staff, 5=Insider, 6=Mixed)
bok	Order book (1=Regular book 2=Special order book 3=Contingent order book)
csq	Client sequence
eft	date and hour of feeding
hdi	Hidden order (0 – normal order, -1 hidden, > 0 order hidden value)
hdv	Visible volume (> 0 for hidden, 0 for the rest)
iac	Internal client account
ini	Initiator User
nmb	Order number (0 for another broker's order)
osq	Order sequence. to be taken in trd
odt	Order type (1=Regular 2=Cross 3=Quote 4=Deal)
opd	Last date of availability (yyyyMMdd)
oqy	Order Quantity
ost	Order status (Order Status (FIX): -1=N/a, 0=New, 1=Partial Fill, 2=Fill, 3=Done, 4=Canceled,7=Stopped, 8=Rejected, 9=Suspended, 10=Pending New, 12=Expired)
own	Own order (true or false)
prc	Price
prt	Price type (Input Price Type (0=NA 1=Market 2=Unpriced 3=Limit))
ref	Reference - Comment
rol	Order role (0=N/a 1=MM 2=Cross)
sde	Order side (1=Buy side 2=Sell side)
shv	Volume to be publicly displayed
siz	Size
ssl	Short sell flag (1=Short Sell 0=N/A 2=Short Sell Exempt)
sts	Order Status (0=Inactive 1=Active 2=Suspended)
tgp	Trigger price
tpa	Trigger type (1=None 2=Stop 3=If Touched 4=TAL)
trm	Order term (0=Fill or Kill 1=Day 2=Open 3=Good Till Date 4=Immediate or Cancel5=Valid for Auction 6=Valid for Closing 7=Valid for Opening)
trd	Number of the latest transaction involving the order
txt	Comment (reject reason)
uti	Update time (0 for other brokers' order)
uty	Update Type (Update type 1=New, 0=Deleted, 2=Changed, 3=Filled,4=Rejected, 5=Confirmed, 6=Released, 7=Suspended, 8=Activated, 9=Rejected FOK, 10=Rejected Odd Lot FOK, 11=Rejected Out of Term, 12=Rejected Out of Price, 13=Rejected Cross Account, 14=Reject New,15=Reject Cancel, 16=Reject Replace, 17=Reject Update MMO,18=Reject Cancel MMO)
uui	Update user
ver	Volume execution restriction (0=None 1=Minimum fill 2=Minimum block 3=All of None)
uts	Update time status

Table 2. Information stored by the BSE for a stock trade

Field	Description
ext	Flag external transaction 1=Yes; 0=No
sty	Symbol type code
cls	Symbol class (share, bond, bill or future)
sde	Transaction side 1=Buy; 2=Sell
sts	Transaction status 1=Active; 0=Canceled
mkt	Market
tss	Transaction sequence allotted
tcs	Clear trade sequence
tck	Transaction number
trt	Transaction moment. If it is uty = 1 (new trade) trt = uti from order participating in the transaction
sym	Symbol
prc	Price
dtp	Dirty price
siz	Volume
val	Value (clearing currency)
vlt	Value (trading currency)
brk	Broker code
mbr	Member code
acc	Account number
act	Account type 1=Client; 2=Financial; 3=House; 4=Staff; 5=Insider; 6=Mixed
grp	Group account number
alv	Allotted volume
ava	Allotted value
clv	Cleared volume
ord	Order number
uid	User Id for the latest modification
ini	User Id initially creating the order
bnk	Bank code
bka	Bank account
std	Settlement date (yyyyMMdd)
din	Deal Indicator >0 =order number; 0=No
fst	Settlement Indicator 1=Yes (the trade will be cleared by Arena); 0=No (the trade will be cleared on another clearing system)
clt	Cleared 1=Yes; 0=No
grs	Gross settlement flag 1=Yes; 0=No
ssl	Short sell 1=Short Sell; 0=N/A 2=Short Sell Exempt
fal	Allocation 1=Yes (the trade has to be allocated); 0=No
ald	Allocated 1=Yes (trade was allocated); 0=No
okt	Old trade number– higher than zero if the trade was introduced to correct an old trade
ref	Reference
uty	Update type 0=Deleted; 1=New; 2=Changed; 3=Allocate; 4=Settle; 5=Deallocate; 6=Discrete settle
uti	Update instant
clc	Settlement currency
lqi	Liquidity Indicator 0=N/A; 1=Aggressive Buy; 2=Aggressive Sell; 3=Route Out; 4=Auction; 5=Cross
iac	Internal account
osq	Order sequence generating the transaction

The coordinating agent is the most complex agent in the system and the only one that is permanently connected to everyone else. Due to the fact that the implemented multiagent system does not require the existence of agents on the same machine (physical or virtual), twice per minute, the coordinating agent checks the connection with the other agents in the system, signalling any interruption of connection to the user. If the user interface agent is not accessible, the coordinating agent beeps on the machine on which it is installed and withdraws all the orders from the market. The coordinating agent shall receive the information from the communication agent or the ETL agent and forward it to the agents to process that information. When, following the processing, resilient, predictive or intelligent agents report critical situations, the coordinating agent sends orders to Bucharest Stock Exchange (BVB) through the communication agent.

The coordinating agent is also responsible for restarting the system when communication with one of the agents has been interrupted. As previously shown, if the coordinating agent fails to resume the interrupted communication, it notifies the user of this and carries out the established procedures to minimize operational risks.

Offline, the coordinating agent takes the update data of the predictive and intelligent models through the interface agent and transfers them to the appropriate agents to be used during the next trading session.

As the implemented multi-agent system does not command the existence of agents on the same machine (physical or virtual) twice a minute, the coordinating agent verifies the connection with the other agents in the system, flagging to the user any connection which has been cut off. In case the agent for the interface with the user is not accessible, the coordinating agent sets off sound signals on the machine where it is positioned and withdraws all the orders on the market.

Furthermore, through the user interface agent, the user is informed about the actions of the system and can interfere over the decisions automatically taken.

The role of the resilient agent in the architecture of the multi-agent system is to minimize the risk of buying at a too high price or selling at a too low price. Four analysis levels operating as triggers for the orders' withdrawal operation from the market are implemented within the agent. All the operations executed and the decisions made by this agent are based on the analysis of own quotes in relation to the best buy and sell prices on the market.

Subsequently, the intelligent agent for price prediction is specialized and always trains with the data of a certain issuer. It is an intelligent agent as it trains every day, tuning into a new agent version for each company at the end of the day. At the end of the trading day, the specific predictive model is generated and inserted in the agent that will use the model for performing predictions in the following trading day. The predictive model is generated following a supervised training process of a neural network. The training of the network is done with the data of the last 100 or 200 trading days depending on the chosen issuer. The predictive model is generated following a supervised training process of a neural network, starting from a discrete dynamic model of NARX type (Nonlinear AutoRegressive models with exogenous inputs) based on the training of a multiperceptron type neural network.

The endogenous variable was considered the share price (daily closing), the exogenous one was considered the volume of transactions. Training, validation and testing data were used and the calculated prediction errors were slightly autocorrelated for small time lags, the rest of the correlations being approximately within the 95% confidence interval around zero, for all the models resulting after training. This demonstrates the validity (adequacy) of the model chosen for price prediction. It has been trained with specific data specific to each issuer. The RapidMiner application can be used to generate this model. The advantage of RapidMiner is that the generated model is described in XML code. The price prediction agent can interpret the injected XML code and execute the model obtained by training to perform the predictive function.

In this study neural networks acquire knowledge through training. A neural network is trained if the application of a set of input vectors will produce the desired outputs (or at least a consistent set of

outputs). The acquired knowledge is stored in the weights of the connections between the neurons. The training vectors are sequentially presented to the network, and the weights are adjusted to store the information that these vectors represent.

The prediction of the price level is a very challenging, difficult and of course risky approach. According to the hypothesis of the efficient market (EMH), the process should instantly and correctly adapt in order to reflect all the available information. This means that, taking all the information into account, no prediction of the future modification of the price can be made (or according to the suggestion of (Kendall & Bradford Hill, 1953), the yields of the actions follow a random walk type process). Despite the massive traditional support for EMH, the majority of the actors on the market considers that they can predict the prices in a manner that could bring them profit.

As long as the predictability of the process is admitted, finding some methods which could improve the power of prediction of the models becomes something crucial.

The decision rules that these models incorporate are based either on the technical analysis or on the fundamental analysis of the information from the past.

The technical analysis uses projection strategies of the future trend, based on the affirmation that the changes of the prices have inertia.

That implies using the former prices of the stock shares, the traded volumes and other related data in order to project the future movement of the prices and to conclude some rules concerning the trading decisions throughout which one can establish when to buy or to sell a financial asset.

Such a set of norms or conditions is called input/output system or transaction system. When applying a set of input/output to a data set, an input/output system is generated (also called buying/selling system or transaction system). The signals of input/output indicate the best moment to buy or sell a financial asset on the market, based on one or more factors on the market.

A transaction system should have some transaction rules which should allow the investors to take advantage both in the situation when the prices are rising and in the situation when the prices are going down; as to say they should be able to decide with respect to the input or output of a certain asset from a portfolio, both for long and short positions.

If the prediction is that the price should rise, one would like to take advantage of this rise by buying the asset (entering long). If the prediction is that the price should fall, one would like to take advantage by borrowing the asset, and then selling it (entering long) with the clear intention of replacing the stocks that have been borrowed when the price was low.

An estimated model using a neural network can be directly used in order to generate the input/output signals or its outputs could be processed by a transaction system in order to produce in/out signals.

At certain time intervals, the model can be estimated once more, by using the new collected data, and thus, permitting the transaction system to update the parameters and to improve its performance.

The most spread feedforward type neural networks which proved to be universal approximators of real functions are: the multy layer perceptrons (MLP = MultiLayer Perceptrons), having one or two hidden layers, which could approximate any computable function on a compact set F(y) with a precision as good as possible, where there is a sigmoid function, the number of the hidden units, but also activation rapids and shares.

MLP offers a direct generalization of the classic manner of modelling the time series. More exactly, they can use a specific mechanism in order to operate with temporal information (layer with temporal delay, with no feedback or time window) and can expand autoaggressive linear model with exogenous variable (ARX) to the ARX nonlinear shape (NARX).

The nonlinear ARX models are potentially stronger than the linear ones, in the way that those can model more complex characteristics which are the base of the time sets and, theoretically, the stationarity of the series should not be assumed.

The feedforward networks are appropriate only for NARX models which have a predictor with no feedback. For other types of models used in the processing of the time series which implies predictors with feedback, recurrent networks can be used, if the future inputs in the network should depend on the past and future output of the network.

The neural networks used together with genetic algorithms have recently given empirical proofs that they are the most efficient instrument of technical analysis, especially due to their ability of representing nonlinear relations and their ability to learn these relations from the data which should be modelled.

A neural network can be used to directly generate signals of buy/sell transaction or its outputs can be processed by a transaction system in order to produce buy/sell signals. Once the transaction system starts to diminish its performance, the neural network can be again trained by adding the new collected data to the ones already in place.

The genetic algorithms can be combined with neural networks in order to improve their performance throughout the identification of the optimal parameters or in order to be used in the post-processing optimization phase. The most popular commercial software which implements a neuro-genetic transaction system is TradingSolutions (http://www.tradingsolutions.com/). It has the ability to use neural networks and genetic algorithms of the technical analysis in a variety of methods. Firstly, a neural network could be used to create a model which estimates the future price of a financial instrument, starting from the previous and actual process and from other and/or fundamental technical data.

The predicted price could be then used in order to recommend when to buy or to sell. The genetic algorithms could be used to optimize the inputs and the neural network parameters in order to produce the best possible model. A second method would be that of training a neural network in order to optimally predict a transaction signal, using as inputs only the current and historical data.

Because the output of this type of neural model is a transaction signal, this could be directly used, without a subsequent preparation throughout a transaction system. On the other hand, the genetic algorithms could be used in order to optimize the settings and parameters of the neural networks. A third method would be that of using a neural method to create a model to estimate the performance of a financial asset for a certain period of time in the future.

As an example, the input in the model could be the actual price, the percentage gain compared to the last week of the last month and the desired output could be the percentage gain of the next week. As in the case of the two previous methods of using the neural networks, the settings and the parameters of the network could be optimized by using genetic algorithms.

TradingSolutions comes with many models of static and dynamic neural networks. The dynamic networks include recurrent networks or those with temporal differences, which are able to train and predict by using the current inputs information and a memory of recent input values and other training values. That makes them appropriate for predicting time series, as in price prediction, because they automatically maintain a memory of recent information which can be applied together with the current information.

There are more different settings associated with predictions in TradingSolutions. The genetic optimization can be used to develop or adjust these internal settings, as the number of elements of processing, of memory and the learning rates of each component of the neural model.

In each case, the error of the neural network is the matching function used to classify the possible solutions.

The genetic algorithms can be used for the post-processing optimization of the predicted transaction signals. The post-processing applies to the output of the values in the network. Different values can be tested to determine which settings are more likely to produce the biggest profit on the basis of the predicted price.

The genetic optimization of those post-processing values is much faster than the genetic optimization of the settings of the neural model.

That is because a new model does not need to be trained for each test.

Each transaction signal in TradingSolutions is automatically analysed from the profitability perspective. This analysis stimulates the transaction of the transaction signal using the real price of this time series. As in the case of real transactions, this one starts with an initial investment and the commission are low at each transaction.

This simulated trading produces an analysis of how each transaction is performed, both on a transaction-by-transaction basis and by one-day aggregation. The curve of the personal capitals is displayed and some common measures of the risk associated with a certain signal are calculated.

In order to build the prediction model, the maximum probability calculation (ML) is used to estimate the mean returns and their volatility.

$$ARMA(0,0): r_t = \mu + \varepsilon_t, \quad (1)$$

where $\varepsilon_t = \sigma_t e_t$ and $e_t \sim i.i.d.N(0,1)$

$$GARCH(1,1): \sigma_t^2 = k + G_1\sigma_{t-1}^2 + A_1\varepsilon_{t-1}^2 \quad (2)$$

where the initial iteration is $\sigma_1^2 = \dfrac{k}{1-G_1-A_1}$

The minimum of mean squared prediction error calculated for the conditional mean and for conditional standard deviation of the returns' series, for an *h* period horizon, shows the asymptotic behavior of the stochastic process.

From

$$\begin{aligned} &\sigma_t^2 = k + G_1\sigma_{t-1}^2 + A_1\varepsilon_{t-1}^2 \\ &\sigma_1^2 = \frac{k}{1-G_1-A_1} \end{aligned} \quad (3)$$

one obtains

$$\sigma_t^2 = \sigma^2 + G_1\left(\sigma_{t-1}^2 - \sigma^2\right) + A_1\left(\varepsilon_{t-1}^2 - \sigma^2\right) \quad (4)$$

For $t = t + h$ it results:

$$\sigma_{t+h}^2 = \sigma^2 + G_1\left(\sigma_{t+h-1}^2 - \sigma^2\right) + A_1\left(\varepsilon_{t+h-1}^2 - \sigma^2\right) \quad (5)$$

It turns out that the best forecast for the volatility of returns for period *h* is:

$$E[{\sigma_{t+h}}^2 \mid I_t] = E\left[\begin{pmatrix} \sigma^2 + G_1\left(\sigma_{t+h-1}^2 - \sigma^2\right) \\ + A_1\left(\varepsilon_{t+h-1}^2 - \sigma^2\right) \end{pmatrix} \mid I_t\right]$$

or

$$E[{\sigma_{t+h}}^2 \mid I_t] = \sigma^2 + \left(G_1 + A_1\right)^{h-1}\left(\sigma_{t+1}^2 - \sigma^2\right),$$
$$h = 1,...,H$$

where $G_1 + A_1$ measures the rate of degradation of extreme volatility, and when $G_1 + A_1 \to 1$, the extreme volatility becomes more persistent.

The asymptotic behaviour is given by :

$$\lim_{h\to\infty} E[{\sigma_{t+h}}^2 \mid I_t] = \sigma^2$$

which demonstrates that long-term prediction of volatility converges to unconditional variance. Moreover:

$$\lim_{h\to 1} E[{\sigma_{t+h}}^2 \mid I_t] = {\sigma_{t+1}}^2$$

demonstrates that the short-term prediction of volatility is the conditional variance of the yield square of the next period.

4. Model Validation

Estimating a NARX price model based on the training of a neural network

Considering a dynamic discreet model NARX type (Nonlinear AutoRegressive models with eXogenous inputs):

$$y(t) = f\begin{pmatrix} y(t-1), y(t-2), \ldots y(t-n_a), \\ x_1(t-n_{k1}-1), x_1(t-n_{k1}-2), \\ \ldots x_1(t-n_{k1}-n_{b1}), \\ \ldots \\ x_m(t-n_{km}-1), x_m(t-n_{km}-2), \\ \ldots x_m(t-n_{km}-n_{bm}), \\ e_t \end{pmatrix}$$

where

$y(t)$ is the endogenous variable (the output of the model which the prediction refers to);

$y(t-1), y(t-2), \ldots y(t-n_a)$ represents a first set of predictors built throughout temporal delays applied to the endogenous variable;

n_a represents the auto-regression order of the endogenous variable;

$x_1(t-n_{k1}-1),\ x_1(t-n_{k1}-2),\ldots,x_1(t-n_{k1}-n_{b1}),$

…

$x_m(t-n_{km}-1),\ x_m(t-n_{km}-2),\ldots,x_m(t-n_{km}-n_{bm}),$

represents a second set of predictors defined by m exogenous variables and the variables obtained by applying some temporal delays of different orders;

$n_{b1}, \ldots, n_{bm}$ represent the orders of the temporal differences applied to the exogenous variables;

$n_{k1}, \ldots, n_{km}$ represent the response time (the delay) associated to the exogenous variables -the time of conveyance of the signal from the entrance of the signal to the exit of the signal for each particular exogenous variable;

e_t is a random variable sequence, uncorrelated.

The construction of the NARX model would be made by training a multiperceptron type neural network.

The endogenous variable is defined as the daily closing, the price of the stock.

The estimating procedure for the corresponding data of the society is demonstrated below.

The endogenous variable is the price.

The exogenous variable is the transaction volume.

Number of hidden layers: 20

Measurement of the predictive performance R2 and MSE:

R2 and MSE for training dataset:

0.99817 0.076039

R2 and MSE for validation dataset:

0.85873 0.14024

R2 and MSE for testing dataset:

0.83267 0.13941

R2 and MSE for validation + testing dataset:

0.86852 0.13982

The model which follows the parsimony principle corresponding to Transgaz Society has the architecture presented in Figure 2.

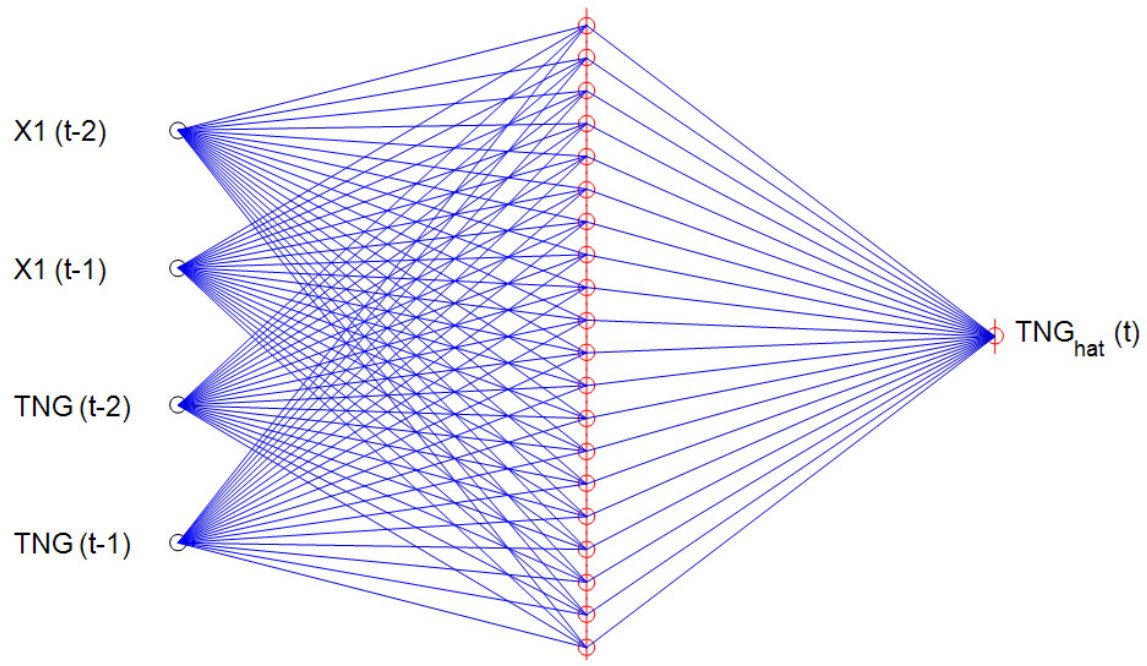

Figure 2. The open loop architecture of a neural network NARX [2, (2), (1)], number of exogenous variables: 1, Total of entries 4, number of neurons 20

To validate the model proposed and presented above, the neural network performing the price prediction for the company S.N.T.G.N. TRANSGAZ S.A. (TGN) has been built, trained, validated and tested. The analysis interval covers the period from 2013 to 2016 and the data have been extracted from the Bucharest Stock Exchange data portal (http://www.bvb.ro/) with an aggregate of 935 recordings for the data series of the variable – daily closing price. Also, the series of values for the daily traded volume has also been stored in the database. From the 935 values 735 have been retained for the network training data and 100 values for each validation and testing.

The training data (735) will generate the calculus of the gradient and weights of the network, without stopping the process.

The validation data (100) shall be used to stop the training as a protection from burning out. For the presented neural network the training data is used to calculate of the gradient and the weight of the network without stopping the process.

The stopping of the process shall be made with the validation data after evaluating the error. This error decreases during the initial training phase, as in the case of the error obtained while evaluating the errors for the validation data.

However, when the network starts to overadjust the validation data (the effect of overrating the system), the evaluated error for the validation data starts to increase. When the error for the validation subset increases for a specified number of iterations, the training process is stopped and the returned weights are those which correspond to the minimum validation error.

In this way the optimal model is selected, based only on the validation data.

The optimal validation performance is reached after 8 training epochs – Figure 3.

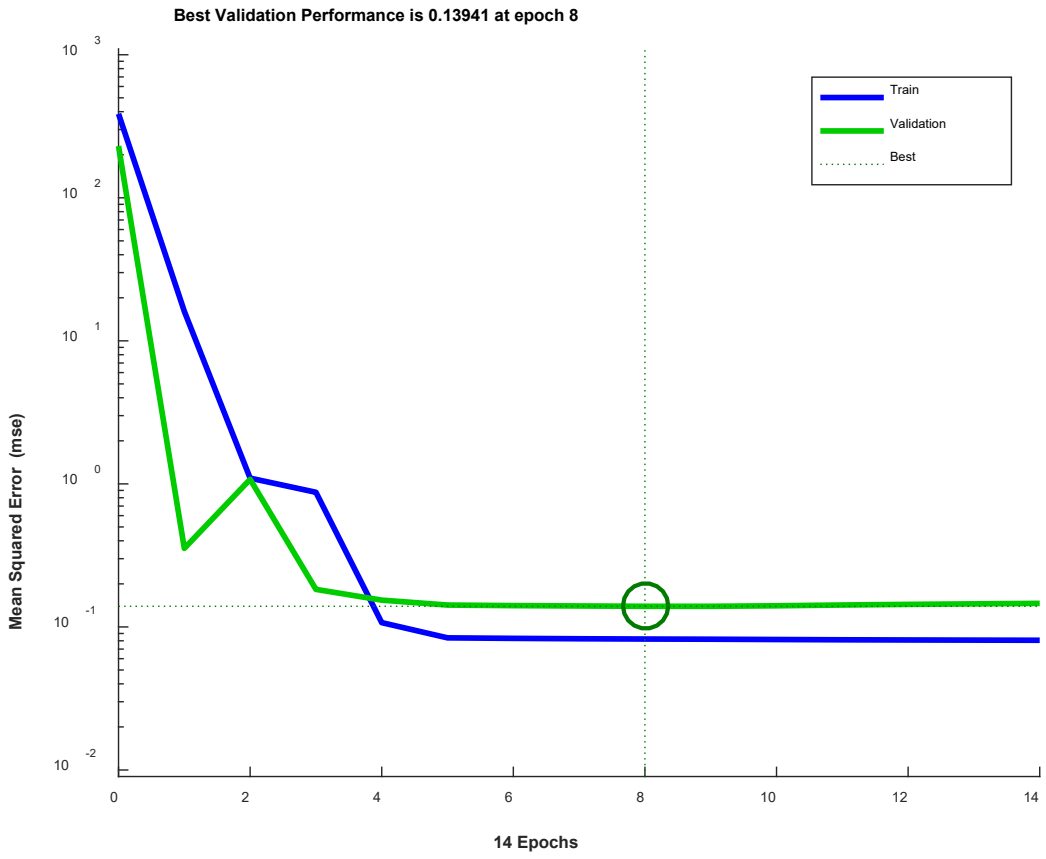

Figure 3. The validation of the performance of the model (TNG)-NARX (2, (2), (1))

Furthermore, the deviations of the predicted values will be assessed from the observed values for the training, validation and testing data. Figures 4, 5 and 6 display the graphs of the time series which have been observed, or predicted for the mentioned data.

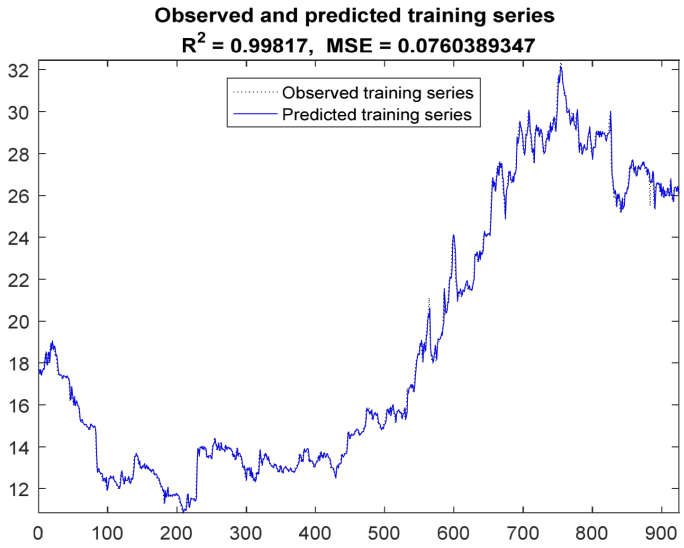

Figure 4. Graph of the observed and predicted time series, for the training data of the proposed model

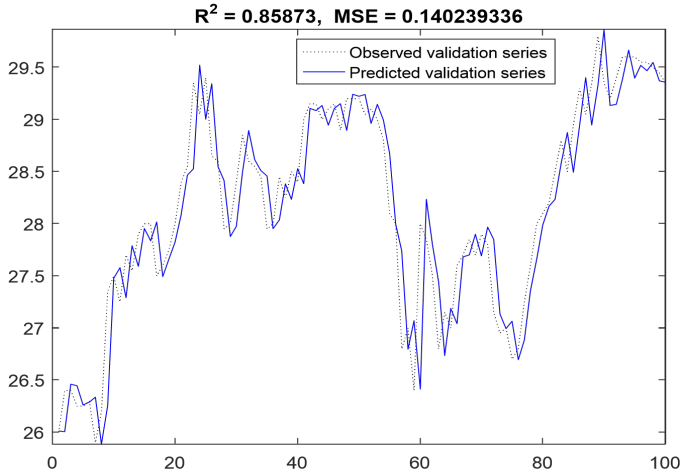

Figure 5. Graph of the observed and predicted time series, for the validation data of the proposed model

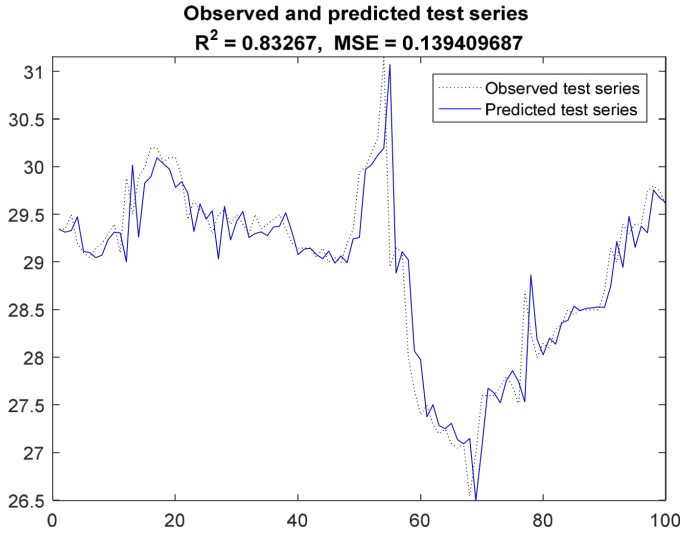

Figure 6. Graph of the observed and predicted time series, for the testing data of the proposed model

One may notice that between the observed values and the predicted ones there are very small differences, difficult to seize at the resolution of the presentation from graphs. The coefficients of determination for the three data subsets (training, validation and testing) are: 0.99817; 0.85873; and 0.83267 respectively.

5. Conclusion

In this paper in order to obtain the models which have been encapsulated in the prediction agents, the parsimony principle has been considered. According to this principle, a model which a model has to include what is required for modelling, and nothing more. The violation of this principle results in excessive adjustment of a model, a phenomenon called ”overfitting”.

This scientific research approach accounts for a vision and a way forward for the development of concrete tools to support the decision-making process, with a dual role – that of early warning for the protection of companies, on one hand, and that of increasing the company’s profitability, on the other hand.

In the first case, for the model created, the decision is made automatically: the decision of trade annulment motivated by the specific conditions which are different from the ones accepted in the trading process is automated and measured in milliseconds.

In the second case, the decision envisages the changing of the risk minimization function into the profit maximization.

Constantly looking for the simplest model to represent a time series as efficiently as possible, it is finally necessary to reach a balance between complexity and accuracy.

REFERENCES

Bucharest Stock Exchange. Available at: <*http://www.bvb.ro/*>, last accessed: 22 February 2020.

Cernazanu, C. (2008). Training Neural Networks Using Input Data Characteristics, *Advances in Electrical and Computer Engineering, 8*(2), 65-70.

Dodd, O. & Gilbert, A. (2016). The Impact of Cross-Listing on the Home Market’s Information Environment and Stock Price Efficiency, *Financial Review*, *51*(3), 299–328.

El Hamidi, K., Mjahed, M., El Kari, A. & Ayad, H. (2019). Neural Network and Fuzzy-logic-based Self-tuning PID Control for Quadcopter Path Tracking, *Studies in Informatics and Control*, *28*(4), 401-412. DOI: 10.24846/v28i4y201904

Filip, F.G. (2014). A Decision-Making Perspective for Designing and Building Information Systems, International, *Journal of Computers Communications & Control*, *7*(2), 264-272.

Georgescu, V. (2011). An Econometric Insight into Predicting Bucharest Stock Exchange MeanReturn and Volatility–Return Processes, *Economic Computation and Economic Cybernetics Studies and Research*, *3*, 25-42.

Gil, D., Girela, J. L., De Juan, J., GomezTorres, M. J. & Johnsson, M. (2012). Predicting seminal quality with artificial intelligence methods, *Expert Systems with Applications*, *39*(16), 12564-12573.

Harik, E. H. C., Gurin, F., Guinand, F., Breth, J. F, Pelvillain, H. & Pard, J.-Y. (2017). Fuzzy logic controller for predictive vision-based target tracking with an unmanned aerial vehicle, *Advanced Robotics*, *31*(7), 368-381.

Haykin, S. (2009). *Neural Networks and Learning Machines*. Prentice Hall Publishing.

Kendall, M. G. & Bradford Hill, A. (1953). The Analysis of Economic Time-Series-Part I: Prices, *Journal of the Royal Statistical Society*, *Series A (General), 116*(1), 11-34. DOI: 10.2307/2980947

Liagkouras, K. (2019). A New Three-Dimensional Encoding Multiobjective Evolutionary Algorithm with Application to the Portfolio Optimization Problem, *Knowledge-Based Systems*, *163*, 186-203.

Nakashima, T., T. Ariyama, H. Kitano, H. & Ishibushi, A. (2005). *Fuzzy Rule-Based Trading Agent: Analysis and Knowledge Extraction, Computational Intelligence Modelling and Prediction Book.* Springer.

Rehar, T., Ogrizek, B., Leber, M., Pisnic, A. & Buchmeister, B. (2017). Product Lifecycle Forecasting Using System’s Indicators, *International Journal of Simulation Modelling*, *16*(1), 45-57.

Sun, Y. F., Zhang, M. L., Chen, S. & Shi, X. H. (2018). A Financial Embedded Vector Model and Its Applications to Time Series Forecasting, *International Journal of Computers Communications & Control*, *13*(5), 881-894.

Trading Solutions. Available at: *<http://www.tradingsolutions.com/>*, last accessed: 12 February 2017.

Tudor, C. (2008). Modelarea volatilității seriilor de timp prin modele GARCH simetrice, *Romanian Economic Journal*, *11*(30), 183-208.

Wang, H., Li, S., Tian, Y. & Aitouche, A. (2016). Intelligent Proportional Differential Neural Network Control for Unknown Nonlinear System, *Studies in Informatics and Control*, *25*(4), 445-452. DOI: 10.24846/v25i4y201605

Wang, S. C. (2003). *Artificial Neural Network*. In: *Interdisciplinary Computing in Java Programming*. The Springer International Series in Engineering and Computer Science, Springer, Boston, MA.

Weidmann, C. & Teuber, L. (2009). *Conception and Installation of System Monitoring Using the SAP Solution Manager*. Galileo Press.

Zhu, G. & Kwong, S. (2010). Gbest-guided artificial bee colony algorithm for numerical function optimization, *Applied mathematics and computation*, *217*(7), 3166-3173.