Complex Intell. Syst. (2018) 4:95–104
https://doi.org/10.1007/s40747-017-0056-6

ORIGINAL ARTICLE

Forecasting of financial data: a novel fuzzy logic neural network based on error-correction concept and statistics

Dusan Marcek[1]

Received: 14 October 2016 / Accepted: 5 September 2017 / Published online: 23 September 2017

Abstract First, this paper investigates the effect of good and bad news on volatility in the BUX return time series using asymmetric ARCH models. Then, the accuracy of forecasting models based on statistical (stochastic), machine learning methods, and soft/granular RBF network is investigated. To forecast the high-frequency financial data, we apply statistical ARMA and asymmetric GARCH-class models. A novel RBF network architecture is proposed based on incorporation of an error-correction mechanism, which improves forecasting ability of feed-forward neural networks. These proposed modelling approaches and SVM models are applied to predict the high-frequency time series of the BUX stock index. We found that it is possible to enhance forecast accuracy and achieve significant risk reduction in managerial decision making by applying intelligent forecasting models based on latest information technologies. On the other hand, we showed that statistical GARCH-class models can identify the presence of leverage effects, and react to the good and bad news.

Keywords RBF neural networks · Support vector machines · ARMA/GARCH models · Volatility modelling · Error-correction mechanism

Introduction

To investigate effects of good and bad news on volatility in the BUX return time series as well as predict the high-frequency time series of the BUX stock index, we used quantitative approaches based on time series models which can be derived from the linear filter model. Box and Jenkins [4] integrated all the knowledge about autoregressive and moving average (ARIMA) models. From that time, ARIMA models have been very popular in time series modelling for long time. tof non-linearity in the financial data. Financial markets behave like complex, random, or chaotic non-linear systems. In this context, asymmetric GARCH models introduced by Bollerslev [3] arose as an appropriate framework for studying these problems.

Artificial neural networks (ANNs) which are the mathematical models inspired by biological neural system are regarded as universal approximators. They are able to perform tasks like pattern recognition, classification, or predictions [1,8,11]. They also have the biggest potential in predicting time series, which are applied very often in financial risk management. The big potential in applying ANN in finance was also confirmed by Hill et al. [13], where the authors showed that ANNs work best in connection with high-frequency financial data. According to Orr [22], Marcek [18], the most used model of regression neural network type is the RBF neural network. A very important concept, which was put forward in scientific field, is Fuzzy cognitive maps (FCM). According to Magalhães et al. [17], FCMs are originated from the theories of neural networks, fuzzy logic, and evolutionary computing. Miao [20] FCM extended to dynamic cognitive network (DCN), taking in the account dynamic nature of the cause with uncertainties. Using historical data, these tools are capable to identify a pattern assuming that the identified pattern will continue into the future.

The first objective of this paper is to investigate the effect of good and bad news on volatility in the Hungary stock market. The second objective is to find any functional relation in the behaviour of the BUX time series and, in turn, fore-

✉ Dusan Marcek
dusan.marcek@vsb.cz

1 Department of Applied Informatics, VŠB-Technical University of Ostrava, Sokolská 33, 701 21 Ostrava, Czech Republic

cast future vales of the series. Firstly, the paper investigates volatility in the BUX return time series using specified non-linear asymmetric models EGARCH, PGARCH which allow identify the presence of leverage effects. Then, the paper proposes three forecasting modelling approaches [statistical, neural networks and support vector machines (SVM)] that generate forecasts on the BUX stock time series. The aim is to examine whether potentially non-linear neural networks outperform latest statistical methods or generate results which are at least comparable with those of statistical models.

This paper is organized in following manner. “Theoretical Background section” deals with the theoretical background of ARMA/GARCH-family models, multilayer feed-forward networks and some basic concepts of SVM theory are refreshed for this paper. “Data and volatility modelling” section examines asymmetric response of volatility to return in the Hungary stock market using asymmetric ARCH-types models. In “Building a prediction model for BUX stock time series and results” section, the resulting statistical models with heteroscedastic noise, the neural networks, and SVMs are applied on 1-day prediction of the BUX stocks index. Here, a novel RBF neural network model is introduced based on error-correction concept. The empirical analysis and findings are presented in “Empirical comparison and discussion” section. “Conclusion” section concludes this paper.

Theoretical background

ARIMA and asymmetric GARCH-type models

Time series models have been initially introduced either for descriptive purposes like prediction or dynamic control. ARIMA models belong to the group of Box-Jenkins methods [4], which are well-established for time series prediction. They combine autoregressive (AR) process, and moving average (MA) process, ‘I’ is an operator for differencing a time series. An ARMA(p, q) model of orders p and q is defined as

$$\begin{aligned} y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t + \theta_1 \varepsilon_{t-1} \\ + \theta_2 \varepsilon_{t-2} + \ldots + \theta_q \varepsilon_{t-q}, \end{aligned} \quad (1)$$

where $\{\phi_i\}$ and $\{\theta_i\}$ are the parameters of autoregressive and moving average parts, respectively, and ε_t is white noise with mean zero and variance σ^2. We assume that ε_t is normally distributed, that is, $\varepsilon_t \sim N(0, \sigma^2)$. Then, the ARIMA (p, d, q) model represents the d-th difference of original time series as a process containing p autoregressive and q moving average parameters. The method of building an appropriate time series forecast model is an iterative procedure that consists of the implementation of several steps. The main four steps are as follows: identification, estimation, diagnostic checking, and forecasting. For details, see [4].

To model the effect of good and bad news on volatility in the Hungary stock market, we will also investigate response of equity volatility to stock return shock. The first model that provides a systematic framework for volatility modelling is the ARCH model proposed by Engle [7]. Bollerslev [3] proposed a useful extension of Engle’s ARCH model known as generalized ARCH (GARCH) model for time sequence $\{\varepsilon_t\}$ in the following form:

$$\varepsilon_t = \nu_t \sqrt{h_t},\ h_t = \alpha_0 + \sum_{i=1}^{m} \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^{s} \beta_j h_{t-j}, \quad (2)$$

where $\{\nu_t\}$ is a sequence of Independent Identical Distribution (IID) random variables with zero mean and unit variance. α_i, β_j are the ARCH and GARCH parameters, and h_t represents the conditional variance of time series. One of the primary restrictions of the GARCH models is that it forces a symmetric response of volatility to positive and negative news. The asymmetric response of good and bad news to future volatility, or leverage effect is such that bad news should increase future volatility while good news should decrease future volatility. In the basic GARCH model (2), if only squared residual ε_{t-i} enter the equation, the signs of the residuals or shocks have no effects on conditional volatility. Nelson [21] proposed the following exponential GARCH model abbreviated as EGARCH to allow for leverage effect in the form:

$$\log h_t = \alpha_0 + \sum_{i=1}^{p} \alpha_i \frac{|\varepsilon_{t-i}| + \gamma_i \varepsilon_{t-i}}{\sigma_{t-1}} + \sum_{j=1}^{q} \beta_j h_{t-j}. \quad (3)$$

Note if ε_{t-i} is positive or there is “good news”, the total effect of ε_{t-i} is $(1 + \gamma_i)\, \varepsilon_{t-i}$. However, contrary to the “good news”, i.e., if ε_{t-i} is negative or there is “bad news”, the total effect of ε_{t-i} is $(1 - \gamma_i)\, |\varepsilon_{t-i}|$. Bad news can have a large impact on the volatility. As stated in the work by Zivot and Wang [24], the value of γ_i would be expected to be negative.

The basic GARCH model can also be extended to allow for the leverage effect. This is performed by treating of the GARCH-type model as a special case of the power GARCH (PGARCH) model proposed by Ding et al. [6]:

$$\sigma_t^d = \alpha_0 + \sum_{i=1}^{p} \alpha_i \left(|\varepsilon_{t-i}| + \gamma_i \varepsilon_{t-i}\right)^d + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^d \quad (4)$$

where d is a positive exponent, and γ_i denotes the coefficient of leverage effects.

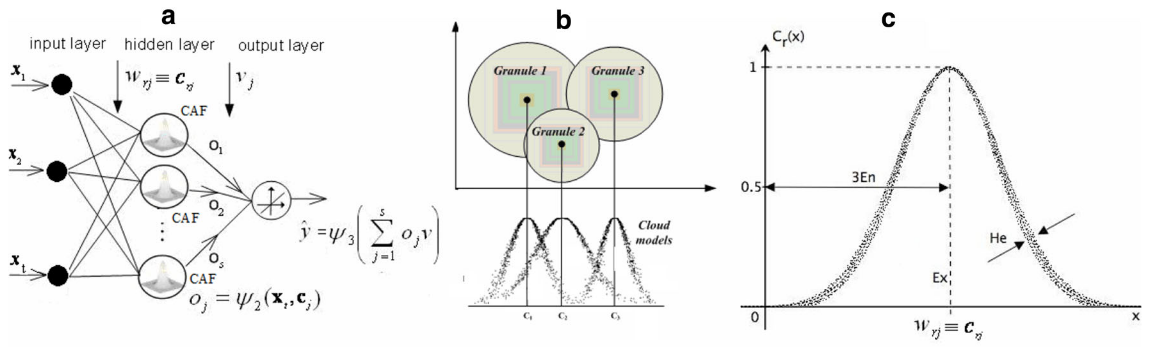

Fig. 1 Construction of RBF neural network from granular computing perspective. **a** RBF neural network with classic or cloud Gaussian activation functions; **b** demonstration of clusters (granules) extracted from data and described by cloud concept; and **c** description of normal cloud concept

Neural networks

Neural networks can be understood as systems which produce output based on inputs the user has defined. It is important to say that user has no knowledge about internal working of the system of ANN. Examples are brought forward the network and then the network tries to get as close as possible to the given output by adapting its parameters (weights). Neural network model has a large number of internal variables which are supposed to set up well to optimize the outputs.

Mathematical model of the neuron is constructed on the basis of functional neuron as a central element of human nervous system, whose task is to transform information from one neuron to the others. The goal of mathematical neuron is the process identification. In other words, we try to find input–output functions, so that the output would have desired parameters and the predicted error would be minimal.

Let $F : x_t \in R^k \rightarrow y_t \in R^1$ is a projection assigning k-dimensional vector of inputs, $x_t^T = (x_{1t}, x_{2t}, \ldots, x_{kt})$ in one-dimensional output y_t in specific time t.

Let $G : G(x_t, w_t) : x_t \in R^k_{\text{train}} \rightarrow y_t \in R^1_{\text{train}}$ is a restriction of F. The task is then to find the values of the weights w_t, so that functional values of G would be so close to known sample as it is possible. Let E(w) is a function defined as

$$E(w_t) = \sum_{x_t, y_t \in R^k_{\text{train}}} (G(x_t, w_t) - y_t)^2. \tag{5}$$

This function represents the squares of the deviations of function G from the expecting values of function F. If a minimum is found, G is adapted for approximation of F. Training or adaption of the network weights w_t is performed on training data set. The validation data set is used for validation of learned network.

According to Anderson [1], the most common used neural network for prediction tasks is the feed-forward network that has at least one hidden layer of neurons. In this network, each layer comprises neurons that receive weighted inputs from the preceding layer and send weighted outputs to the neurons in the succeeding layer. There is no feedback connections. The most known representatives of feed-forward networks are perceptrons and their modified version called RBF neural network (RBFNN) [10].

The RBFNN architecture illustrated in Fig. 1a is quite similar to the perceptron type network; however, there are some differences which include: calculation of the potential and different activation functions in the hidden layer neurons.

The RBFNN defines the potential of the jth neuron in the layer as a difference of the Euclidean distance given by the vector:

$$u^j = ||x - w^j||^2, \quad \text{for} \quad j = 1, 2, \ldots, s, \tag{6}$$

where x is a k-dimensional neural input vector, and w^j represents the weights in the hidden layer. The RBFNN also uses different types of the activation functions known as Gaussian or radial basis function. The activation function for jth hidden neuron is defined as

$$\psi_2(u^j) = e^{\frac{-u^j}{2\sigma_j^2}} = e^{\frac{||x-w^j||^2}{2\sigma_j^2}}, \quad \text{for} \quad j = 1, 2, \ldots, s, \tag{7}$$

where σ_j^2 is the variance of input data sets. If the data in the input vectors are not orthogonal, then the activation function looks like this

$$\psi_2(u^j) = e^{\frac{-(x-w^j)^T \Sigma^{-1} (x-w^j)}{2}}, \tag{8}$$

where $\sum^{-1}$ is the inverse of the variance–covariance matrix of input data.

The activation function of the output neuron is also different; the output neuron is always activated by the linear function $y = u$, where u is the potential of the output neuron calculated as the scalar product of the weight vector $\boldsymbol{v}$ and the output vector $\boldsymbol{o}$.

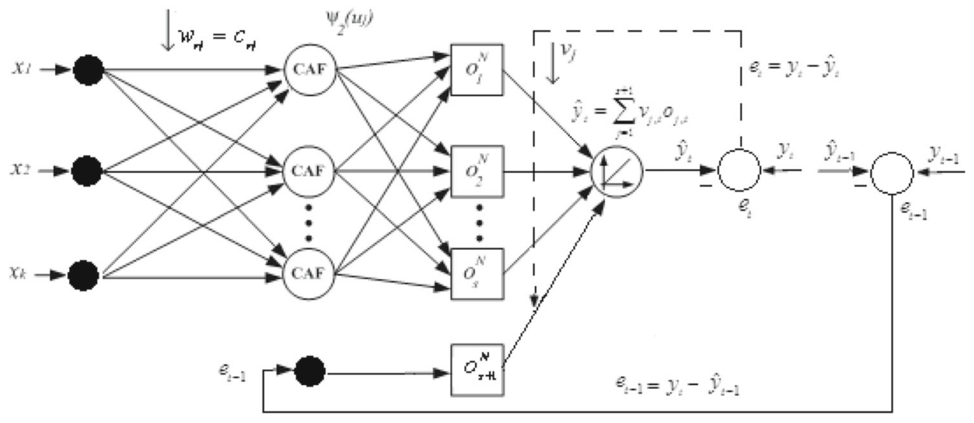

Fig. 2 Architecture and learning scheme of classic RBF neural network for time series prediction with lagged error term e_{t-1} feedback. *Dashed arrow* indicated that weights adaption depends on the error e_t. CAF means the cloud activation function, o_j^N, $j = 1, 2, \ldots, s$ the normalized outputs from neurons in the hidden layer

To find the weights $\mathbf{w}_j$ or centres of activation functions for the neurons in the hidden layer, we used the following adaptive (learning) version of K-means clustering algorithm for s clusters, which uses the Kohonen's adaptive rule [15]

Step 1. Randomly initialize the centres of RBF neurons $c_j^{(t)}$, $j = 1, 2, \ldots, s$ where s represents the number of chosen RFB neurons (clusters).
Step 2. Apply the new training vector $x^{(t)} = (x_1, x_2, \ldots, x_k)$.
Step 3. Find the nearest centre to $x^{(t)}$ and replace its position as follows: $c_j^{(t+1)} = c_j^{(t)} + \lambda(t)(x^{(t)} - c_j^{(t)})$, where $\lambda(t)$ is the learning coefficient selected as a linearly decreasing function of t by $\lambda(t) = \lambda_0(t)\left(1 - \frac{t}{N}\right)$ where $\lambda_0(t)$ is the initial value, t is the presented learning cycle, and N is the number of learning cycle.
Step 4. After chosen epochs number, terminate learning. Otherwise go to **Step 2**.

The RBFNN computes the output data set as

$$\hat{y} = G(\mathbf{x_t}, \mathbf{c}, \mathbf{v}) = \sum_{j=1}^{s} v_{j,t}\psi_2(\mathbf{x_t}, \mathbf{c}_j)$$
$$= \sum_{j=1}^{s} v_j o_{j,t}, t = 1, 2, \ldots, N, \tag{9}$$

where N is the size of data samples, and s denotes the number of hidden layer neurons. Hidden layer neurons receive the Euclidian distances $(\|\mathbf{x} - \mathbf{c_j}\|)$ and computes the scalar values $o_{j,t}$ of the Gaussian function $\psi_2(\mathbf{x_t}, \mathbf{c}_j)$ that form the hidden layer output vector $\mathbf{o_t}$. Finally, the single linear output layer neuron computes the weighed sum of the Gaussian functions that form the output value of $\hat{y}_t$.

If the output values $o_{j,t}$ from the hidden layer will be normalized, where the normalisation means that the sum of the outputs from the hidden layer is equal to 1, then the RBF neural network will compute the output data set $\hat{y}_t$ as follows:

$$\hat{y}_t = G(\mathbf{x_t}, \mathbf{c}, \mathbf{v}) = \sum_{j=1}^{s} v_{j,t}\frac{o_{j,t}}{\sum_{j=1}^{s} o_{j,t}}$$
$$= \sum_{j=1}^{s} v_{j,t}\frac{\psi_2(\mathrm{x_t}, \mathrm{c_j})}{\sum_{j=1}^{s}\psi_2(\mathrm{x_t}, \mathrm{c_j})}, \quad t = 1, 2, \ldots, N. \tag{10}$$

The network with one hidden layer and normalized output values $o_{j,t}^N$, according to Kecman [14], is called the fuzzy logic model or soft RBF neural network, see Fig. 2. The learning scheme uses the first-order gradient procedure. In our case, the subjects of learning are the weights $v_{j,t}$ only. These weights can be adapted by error Back-propagation algorithm. In this case, the weight update is particularly simple. If the estimated output for the single output neuron is $\hat{y}_t$, and the correct output should be y_t, then the error term e_t is given by $e_t = y_t - \hat{y}_t$ and the learning rule has the form:

$$v_{j,t} \leftarrow v_{j,t} + \eta o_{j,t} e_t, \quad t = 1, 2, \ldots, N \tag{11}$$

where the term, $\eta \in (0, 1>$, is a constant called the learning rate parameter, and $o_{j,t}^N$ is the normalized output signal from the hidden layer. Typically, the updating process is divided into epochs. Each epoch involves updating all the weights for all the examples.

Neuron in the hidden layer of RBF neural network can also be described from granular computing perspective, see Fig. 1b. Granules as basic components of granular computing are used to represent these neurons. Granules are extracted from data in the form of clusters. Clustering is performed via competitive learning using Kohonen's adaptive rule as the above K-mean clustering method. Thus, granules are extracted from data in the form of clusters, i.e., entities embracing collections of numerical data that exhibit some functional or descriptive commonalities. The centres of clusters are regarded as the means of granules. Above-mentioned competitive learning algorithm is regarded as one of the granular methods presenting bottom–up granulation, i.e., input data are combined into large granules. The standard deviation of a granule can be calculated as follows:

$$\sigma_j = \sqrt{\frac{1}{M}\sum_{t=1}^{M}\|c_j - x_m^2\|}, \tag{12}$$

where x_m is the mth input vector with input patterns which belongs to the cluster with centre c_j. M_j is the number of input vectors which belong to the cluster with the centre c_j. To improve the abstraction ability of soft RBF neural networks with architecture depicted in Fig. 1b, we replace the standard Gaussian activation (membership) function of RBF neurons with functions based on the normal cloud concept [16]. Cloud models are described by three numerical characteristics: expectation (*Ex*) as the most typical sample which represents the qualitative concept, entropy (*En*) as the uncertainty measurement of the qualitative concept and hyper entropy (*He*) which represents the uncertain degree of entropy, see Fig. 1c. Then, in case of soft RBF neural network, the Gaussian activation $\psi_2\,(./.)$ in Eq. (9) or (10) has the form [19]:

$$\begin{aligned}\psi_2(x_t, c_j) &= \exp\left[-\left(x_t - E(\mathrm{x})\right)^2/2(En')^2\right]\\ &= \exp\left[-\left(x_t - c_j\right)^2/2\left(En'\right)^2\right]\end{aligned} \tag{13}$$

where is a normally distributed random number with mean and standard deviation is the expectation operator. We call this neural network as granular RBF neural network (GRBFNN).

Novel RBF neural network model based on error-correction mechanism

We show a new approach for economic (financial) time series forecasting by means of RBF neural networks based on Gaussian activation function modelled by cloud concept. Most of the existing learning approaches do not take in account the feedback of error terms identified in the output neuron as

$$e_{t-1} = \left(y_{t-1} - \sum_{i=1}^{s+1} \upsilon_{i,t-1} o_{i,t-1}\right) = \left(y_{t-1} - \hat{y}_{t-1}.\right) \tag{14}$$

In economic modelling, the feedback of error term was proposed by Banerjee et al. [2]. According to stationary and equilibrium relationship concept, the feedback of error terms (14) plays an explicit role in economic modelling. The thought of this proposal is based on the economic theory of the co-integrated variables which are related to an error-correction model. The simple mean equation (1) can be interpreted as long-run relationship that hold among the $k = p + q$ input variables to y when the system is in equilibrium. The phrase 'long-run equilibrium' is used to denote the equilibrium relationship to which a system converges over time. Thus, long-run relationship will often hold "on average" over time [2]. If we denote the equilibrium by $y_t = \sum_{i=1}^{s+1} \upsilon_{i,t} o_{i,t}$, then the previous discrepancy $\left\{y_{t-1} - \sum_{i=1}^{s+1} \upsilon_{i,t} o_{i,t}\right\}$ represents the previous disequilibrium. This previous disequilibrium or error should be useful explanatory variable for the next direction of movement of output variable $\hat{y}_t$. For example, when $\left\{y_t - \sum_{i=1}^{s+1} \upsilon_{i,t} o_{i,t}\right\}$ is positive, y_t is too high relatively to $\hat{y}_t$, and on average, we might expect an increment in $\hat{y}_t$ in future periods relatively to its trend growth. The formula $\left\{y_t - \hat{y}_t\right\} = \left(y_t - \sum_{i=1}^{s+1} \upsilon_{i,t} o_{i,t}\right), t = 1, 2, \ldots, N$, is also called an error-correction mechanism and is also denoted as 'short-run relationship', because in comparison with long-run relationships, those holding over a relatively short period.

The true parameters (weights) υ_i characterizing the relationships expressed by Eq. (14) are not known, but they can be estimated in the course of modelling and learning the variable of interest. The network architecture with new learning scheme taking in account the feedback of error term for BUX time series predictions is shown in Fig. 2.

Support vector regression model

Despite the fact that RBF neural networks possess a number of attractive properties such as universal approximation ability and parallel structure, they still suffer from problems like the existence of many local minima and the fact that it is unclear how one should choose the number of hidden units. The SVM method is comparatively new learning system that is used in both forecasting and classification problems. This machine uses the hypothesis space of the linear functions in a high-dimensional feature space, and it is trained with a learning algorithm based on optimization theory. The SVM has been recently, introduced by Vapnik [23]. Support vector regression (SVR) is an extension of the support vector machine algorithm for numeric prediction. Its decision boundary can create complex non-linear decision boundaries while reducing the computational complexity.

Non-linear SVR is frequently interpreted by using the training data set $\{y_k, \mathbf{x}_k\}_{k=1}^{N}$ with input data $\mathbf{x}_k \in \Re^N$ and output data $\mathbf{x}_k \in \Re$ as follows:

$$f\,(\mathbf{x}, \mathbf{w}, b) = \sum_{i=1}^{N} \mathbf{w}_i \varphi_i\,(\mathbf{x}) + b, \tag{15}$$

where $\varphi_i\,(\mathbf{x})$ are called features (the input data are projected to a higher dimensional feature space). To perform SVM regression one optimizes the cost (empirical risk) function

$$R_{\mathrm{emp}} = \frac{1}{N}\sum_{i=1}^{N} |y_i - f\,(\mathbf{x}, \mathbf{w})|_{\varepsilon} \tag{16}$$

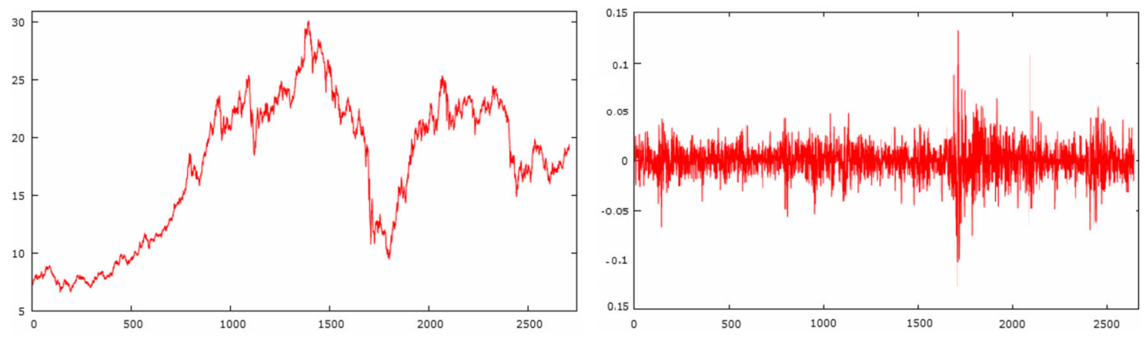

Fig. 3 Time series of the daily closing prices of BUX stocks (1.2004–12.2012) on the *left*. Time series of the BUX return values (1.2004–12.2012) on the *right*

containing Vapnik's ε-insensitive loss function with ε-insensitivity zone which is defined as follows [23]:

$$\begin{cases} 0, \text{ if } |y - f(x, w)| \leq \varepsilon, \\ |y - f(x, w)| - \varepsilon, \text{ otherwise.} \end{cases} \tag{17}$$

This leads to the optimization problem:

$$\min R_{\text{emp}}\left(w, \xi, \xi^*\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) \tag{18}$$

subject to

$$\begin{cases} y_i - \mathbf{w}^T x - b \leq \varepsilon + \xi_i, & i = 1, 2, \ldots, N \\ \mathbf{w}^T x + b - y_i \leq \varepsilon + \xi_i^*, & i = 1, 2, \ldots, N \\ \xi_i, \xi_i^* \geq 0, & i = 1, 2, \ldots, N \end{cases} \tag{19}$$

where ξ_i, ξ_i^* are positive slack variables and C is regularization parameter which influences a trade-off between an approximation error and weights vector norm.

Finally, the SVM non-linear function estimation takes the form:

$$f(\mathbf{x}, \mathbf{w}, b) = K\left(\mathbf{x}_i, \mathbf{x}_j\right)\mathbf{w} + b, \tag{20}$$

where so-called kernel trick was applied $K\left(\mathbf{x}_i, \mathbf{x}_j\right) = \boldsymbol{\varphi}^T\left(\mathbf{x}_i\right)\boldsymbol{\varphi}\left(\mathbf{x}_j\right)$ (so-called Kernel function) within the formulation of this quadratic programming problem. Note that in the case of RBF kernels, the parameters C, σ, ε have to be considered as additional tuning parameters.

Data and volatility modelling

Our goal is to examine and compare the behaviour of asymmetric response of equity volatility on return shocks in the Hungary stock market before, during and after financial crisis for the period January 7, 2004 to December 21, 2012, which provided of 2256 daily observations. We have 9 year long time series of the closing prices of BUX stocks data. To access the BUX time series data, see [5]. The return r_t at time t is defined in the logarithm of BUX indices y_t values that is, $r_t = \log(y_t) - \log(y_{t-1})$. Time series of daily BUX values which is depicted in Fig. 3 on the left exhibits non-stationary behaviour. However, after its first differencing becomes stationary. As can be seen from Fig. 3 on the right, returns fluctuate around mean value that is closed to zero and also shows volatility clustering where large returns tend to be followed by small returns. Since the volatility is highest in 2008 when the values of BUX stocks reached the minimum in investigated period, we divided the basic period into two periods. First period (as the training data set) was defined from January 2004 to the end of June 2007, i.e. the time before the global financial crisis, and the second one so-called crisis and post-crisis period (validation data set or ex post period) started at the beginning of July 2007 and finished by the end of 2012.

Table 1 Descriptive statistics of BUX returns (2004–2012)

Period: 1.2004 – 12.2012	
Mean	0.000214
SD	0.007343
Skewness	−0.101284
Kurtosis	9.528772
Jacques-Bare test	4174.147

The summary statistics (mean, standard deviation, skewness, and kurtosis) for the daily BUX returns in both investigated periods are given in Table 1. From Table 1 can be seen that the skewness coefficient is negative, suggesting that the BUX return series have a long tail while kurtosis is high what means that the distribution is leptokurtic. Jacques–Bare test is significant at 0.05.

We estimated appropriate GARCH, EGARCH, and PGARCH models for BUX returns series using the maximum likelihood method assuming the Gaussian standard normal

Table 2 Estimated volatility models BUX returns

Coefficient	Value	SD	p value
Period: 1.2004–6.2007			
PGARCH(1, 1)			
d	1.9E−08	1.21E−07	0.8717
α_0	0.03902	0.032215	0.2257
α_1	−0.0114	0.084162	0.8918
γ	0.89404	0.039859	0.0000
β	2.81050	1.201266	0.0193
EGARCH(1,1)			
α	−0.4520	0.262047	0.0845
α_1	0.10597	0.039903	0.0079
γ_1	−0.0093	0.017803	0.6006
β_1	0.96474	0.023303	0.0000
Period: 7.2007–12.2012			
PGARCH(1, 1)			
d	2.9E−06	5.69E−06	0.6063
α_0	0.09100	0.017187	0.0000
α_1	0.30564	0.082398	0.0002
γ	0.89006	0.017038	0.0000
β	1.81247	0.385426	0.0000
EGARCH(1,1)			
α	−0.3421	0.06805	0.0000
α_1	0.18012	0.02502	0.0000
γ_1	−0.06690	0.01409	0.3031
β_1	0.97965	0.00594	0.0000

distribution. The estimation results for BUX returns are given in Table 2. The asymmetric effect captured by the parameter estimate γ was positive and significant in the PGARCH(1,1) model suggesting the presence of leverage effect in both periods. For instance, in crisis period, it is clear that the good news has an impact of 0.30564 magnitude and the bad news has an impact of 0.30564 + 0.89006 = 1.19570. The estimation results of EGARCH(1,1) model captured by the parameter estimate γ was insignificant for both periods.

In many cases, the basic GARCH-family models (2), (3), (4) which are modelled with normal Gaussian error distribution provides a reasonably good model for analysing financial time series and estimating conditional volatility. However, there are some aspects of the model which can be improved, so that it can better capture the characteristics and dynamics. Furthermore, we re-estimated asymmetric GARCH models after having eliminated the restrictive assumption that their error terms follow a normal distribution. Table 3 presents Akaike Information Criteria (AIC) and Log-Likelihood functions (LL) based on assumptions that residuals follow successively a Student's distribution or Generalized Errors Distribution (GED). From Table 3, it is shown that the assumption of Student's errors is the best.

Table 3 Information criteria and log-likelihood function for re-estimated volatility models

Criteria model	PGARCH, Eq. (3)	EGARCH, Eq. (4)	Distribution
Period: 1.2004–6.2007			
AIC	−7.632126	−7.624700	Normal
LL	3444.273	3439.928	
AIC	−7.639619	−7.636049	Student's
LL	3448.648	3446.040	
AIC	−7.638942	−7.633344	GED
LL	3448.344	3444.822	
Period: 7.2007–12.2012			
AIC	−7.118371	−7.112786	Normal
LL	6601.171	6594.997	
AIC	−7.131425	−7.128624	Student's
LL	6614.265	6610.670	
AIC	−7.129529	−7.126068	GED
LL	6612.509	6608.302	

Building a prediction model for BUX stock time series and results

We illustrate the ARIMA/GARCH methodology on the developing a forecast model for BUX stocks using the data sets as for each GARCH-type model above.

Statistical approach

The relevant lag structure of potential inputs was analysed using traditional statistical tools, i.e., the autocorrelation function (ACF), partial autocorrelation function (PACF), and the Akaike information criterion. We look to determine maximum lag for which PACF coefficient was statistically significant and the lag gives the minimum AIC. According to these criterions the ARIMA(1, 1, 0) model was specified as follows:

$$\Delta y_t = \xi + \phi_1 \Delta y_{t-1} + e_t, \tag{21}$$

where Δ is the difference operator defined as $\Delta y_t = y_t - y_{t-1}$, e_t are the residuals. The estimated parameters of specified ARIMA(1, 1, 0) model are reported in Table 4.

As we mentioned early, high-frequency financial data, like our BUX stock time series, reflect a stylized fact of changing variance over time. An appropriate model that would account for conditional heteroscedasticity should be able to remove possible non-linear pattern in the data. Various procedures are available to test an existence of ARCH-type model. A commonly used test is the LM (Lagrange Multiplier) test [7]. The LM test assumes the null hypothesis $H_0 : \alpha_1 = \alpha_2 = \cdots = \alpha_p = 0$ that there is no ARCH. The LM statistics

Table 4 Estimated mean Eq. (21) for BUX stock time series

Coeff.	Value	SD	p value	D-W
ξ	3.938362	5.899953	0.5045	1.994470
ϕ_1	0.047107	0.019045	0.0134	

Table 5 Information criteria and log-likelihood functions for re-estimated asymmetric variance models (see text for details)

Criteria model	GARCH	PGARCH	EGARCH	Distribution
Period: 1.2004 – 6.2007				
AIC	−12.87944	−12.48636	−12.88632	Student's
LL	5777.869	5611.864	5779.957	
AIC	−12.88034	−12.48725	−12.88750	GED
LL	5778.271	5612.262	5780.490	

Table 6 Re-estimated mean Eq. (21) for BUX stock values, assuming that the random component follow EGARCH(1,1) GED process

Coeff.	Value	SD	$p$value	D-W
ξ	11.6961	3.71671	3.1469	1.93590
ϕ_1	−0.0157	0.03488	−0.4493	

has an asymptotic χ^2 distribution with p degrees of freedom under the null hypothesis. The ARCH-LM test up to 10 lags was statistically significant of the mean equation (21).

For estimation the parameters of an ARCH or GARCH model the maximum likelihood procedure was used and resulted into the following variance equation:

$$\begin{aligned} h_t &= \alpha_0 + \sum_{i=1}^{m} \alpha_i e_{t-i}^2 + \sum_{j=1}^{s} \beta_j h_{t-j} \\ &= 675.0 + 0.125 e_t^2 + 0.854 \end{aligned} \tag{22}$$

Furthermore, the variance model given by Eq. (22) was re-estimated considering that the residual follow Student's distribution, and after GED. The model with the lowest value of AIC fits the data best. Table 5 presents AIC, log-likelihood (LL) functions in all cases. As can be seen in Table 5 the smallest AIC has just the EGARCH(1,1) with GED distribution. After these findings we re-estimated the mean Eq. (21) assuming that the random component e_t follows EGARCH(1,1) GED. Re-estimated parameters are given in Table 6.

Neuronal and SVM approach

The GRBFNN according to the architecture depicted in Fig. 2 was trained using the variables and data sets as the statistical ARIMA(1,1,0)/EGARCH(1,1) model above. In the GRBFNN, the non-linear forecasting function $f(\mathbf{x})$ was estimated according to the expression (10) with radial basic function $\psi_2(./.)$ given by Eq. (13). We used own application of the feed-forward neural network of RBF type with one hidden layer. For standard neural network, we tested three-to-ten hidden neurons to achieve the best results of the network. For every model, only the result with the best configuration was stated. We used linear function as an activation function for the output layer too. The weights of investigated networks were initiated randomly-generated from the uniform distribution $<0, 1)$. The learning rate of back-propagation algorithm was set to 0.005 to avoid the easy imprisonment in the local minimum. Final results were taken from the best of 5000 epochs and not from the last epoch to avoid overfitting of the neural network.

The prediction of BUX stock values for the ex post period was also done by SVR model using software developed by Gunn [9] which is the implementation of Vapnik's Super Vector Machine for the problem of pattern recognition, regression, and ranking function [23]. To achieve high testing accuracy, a suitable kernel function, its parameters, and the regularization parameter C should be properly selected. Hua and Sun [12] have proved that the Gaussian kernel provide superior performance in the generalization ability and convergence speed. To set the standard deviation σ of Gaussian kernel function, and the magnitude of insensitivity zone ε, we examined various combination of ε, σ and searched which combination provides the lowest prediction error. The regularization parameter C which control the trade-off between complexity and misclassified training example was set to 10^5. The best parameter settings ε, σ on estimated SVR function (20) are 0.2, 0.52 respectively.

Empirical comparison and discussion

Table 7 presents two statistical measures of model's forecast accuracy based on the Mean Absolute Percentage Error (MAPE) and the root mean square error (RMSE) calculated over the validation data set and shows the results of the methods used for comparison. The best performing method is GRBFNN with error-correction mechanism followed SVR and soft GRBFNN. A comparison between latest statistical and intelligent methods shows that intelligent prediction methods outperformed the latest statistical forecasting method. Practical advantage of the error-correction mechanism is that the extent of adjustment in a given period to deviations from long-run equilibrium is given by the estimated input–output equation directly from neural network without any further calculation. Further, from Table 7 it is shown that all forecasting models used are very accurate. The development of the error rates on validation data set showed a high inherent deterministic relationship of the underlying

Table 7 The statistical summary measures of model's forecast accuracy

Model	RMSE	MAPE
ARIMA (1,1,0)/EGARCH(1,1) GED	463.93	1.650
Soft GRBFNN	407.17	1.453
GRBFNN with error-correction mechanism	295.56	1.197
SVR	392.3	1.421

variables. Though promising results were achieved with all approaches, for the chaotic financial markets a purely linear (statistical) approach for modelling relationships does not reflect the reality. For example if investors do not react to a small change in the BUX stock values at the first instance, but after crossing certain interval of threshold react all the more, then a non-linear relationship between Δy_t and Δy_{t-1} exists in model (20).

RBFNNs have such attributes as computational efficiency, simplicity, and easy adjusting to changes in the process being forecast. Thus, neural networks are usually used in the complicated problems of prediction, because they minimize the analysis and modelling stages and the resolution time.

Conclusion

We investigated the volatility of the BUX stock return using two non-linear asymmetric models: PGARCH (1,1) and EGARCH(1,1). We found that the BUX stock return series exhibits leverage effects. In addition to leverage effect, it exhibits other stylized facts such as volatility clustering and leptokurtosis associated with return on developed market. In case of the BUX returns we found that PGARCH(1,1) model with Student's errors can be appropriate representative of the asymmetric conditional volatility process in both pre-crisis and crisis or post-crisis periods.

In case of construction forecasting models for BUX stock time series, we proposed four approaches. The first one was based on latest statistical ARMA/GARCH methodologies, the second one on soft GRBFNN, the third on novel GRBFNN based on incorporation of an error-correction mechanism. The fourth approach was forecasting model based on SVR method, which is comparatively new learning method that uses hypothesis space of linear function in a high-dimensional feature space and it is trained with a learning algorithm based on the optimization theory.

After performed experiments it was established that forecasting model based on SVR is better than ARMA/GARCH one to predict high-frequency financial data for the BUX stock time series. The direct comparison of forecast accuracies between the statistical ARMA/GARCH forecasting model and its neural representation shows that both investigated methodologies yield very little MAPE values. Moreover, our experiments show that neural forecasting systems are economical and computational very efficient, well suited for high-frequency data forecasting. In future also more ways of combining prediction techniques will also be tested to see if hybrid network architectures are better than single one.

Acknowledgements This work was supported within Operational Programme Education for Competitiveness—Project No. CZ.1.07./2.3.00./20.0296.

References

1. Anderson JA, Rosenfeld E (1988) A collection of articles summarizing the state-of-the-art as of 1988. Neurocomputing: foundations of research. MIT Press, Cambridge
2. Banerjee AJ, Dolado J, Galbrait JW, Hendry DF (1999) Co-integration, error correction, and the econometric analysis of non-stationary data. Oxford University Press, Oxford. http://www.oxfordscholarship.com/view/10.1093/0198288107.001.0001/acprof-9780198288107
3. Bollerslev D (1986) Generalized autoregressive conditional heteroscedasticity. J Econ 31:307–327. http://www.sciencedirect.com/science/article/pii/0304407686900631
4. Box GEP, Jenkins GM (1970) Time series analysis, forecasting and control. Holden-Day, San Francisco. http://garfield.library.upenn.edu/classics1989/A1989AV48500001.pdf
5. BUX time series data. http://www.bse.hu. Accessed 21 Sep 2014
6. Ding Z, Granger CW, Engle RF (1993) A long memory property of stock market returns and a new model. J Empir Finance 1: 83–106. http://www.sciencedirect.com/science/article/pii/0927-5398(93)90006-D
7. Engle RF (1982) Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. Econometrica 50(4): 987–1007. http://links.jstor.org/sici?sici=0012-9682%2819820...O%3B2-3&origin=repec
8. Gooijer JG, Hyndman RJ (2006) 25 years of time series forecasting. Int J Forecast 22:443-473. http://www.robjhyndman.com/papers/ijf25.pdf
9. Gunn SR (1997) Support vector machines for classification and regression. Technical report, image speech and intelligent systems research group. University of Southampton, Southampton. http://www.isis.ecs.soton.ac.uk/isystems/kernel
10. Hecht-Nielsen R (1990) Neuro-computing. Addison-Wesley, Reading
11. Hertz H, Krogh A, Palmer RG (1991) Introduction to the theory of neural computation. Westview Press, USA
12. Hill TL, Marquez M, O'Connor, Remus W (1994) Neural networks models for forecasting and decision making. Int J Forecast 10. http://www.sciencedirect.com/science/article/pii/0169207094900450
13. Hua S, Sun Z (2001) A method of protein secondary structure prediction with high segment overlap measure: support vector machine approach. J Mol Biol 308:397–407. https://www.ncbi.nlm.nih.gov/pubmed/11327775

14. Kecman V (2001) Learning and soft computing—support vector machines, neural networks and fuzzy logic models. Massachusetts Institute of Technology. http://dl.acm.org/citation.cfm?id=559026&preflayout=flat
15. Kohonen T (2012) Self-organization and associative memory. Springer, Berlin. http://scholar.google.cz/scholar?q=Kohonen+T+2012+Self-Organization+and+Associative+Memory.+Berlin:+Springer-Verlag&hl=cs&as_sdt=0&as_vis=1&oi=scholart&sa=X&ved=0ahUKEwjh0rzL4c_PAhUMCcAKHZViDw8QgQMIGTAA
16. Li D, Du Y (2008) Artificial intelligence with uncertainty. Chapman & Hall/CRC, Taylor & Francis Group, Boca Raton. http://dlia.ir/Scientific/e_book/Science/Cybernetics/006280.pdf
17. Magalhães MH, Ballini R, Gomide FAC (2008) Granular models for time-series forecasting. In: Pedrycz W, Skowron A, Kreinovich V (eds) Handbook of granular computing. Wiley, Chichester, pp 949–967
18. Marcek D (2004) Some intelligent approaches to stock price modelling and forecasting. J Inf Control Manag Syst 2(1). http://jicms.fri.uniza.sk/index.php/jicms/article/view/742
19. Marcek M, Marcek D (2008) Granular RBF neural network implementation of fuzzy systems: application to time series modeling. J Mult.-Valued Logic Soft Comput 14(3–5):401–414. http://dblp2.uni-trier.de/db/journals/mvl/mvl14
20. Miao Y, Liu Z, Siew C, Miao C (2001) Dynamical cognitive network an extension of fuzzy cognitive map. IEEE Trans Fuzzy Syst 9(5):760–770. http://dl.acm.org/citation.cfm?id=2234978
21. Nelson DB (1991) Conditional heteroskedasticity in asset returns: a new approach. Econometrica 59(2): 347–370. http://scholar.google.sk/scholar?q=Nelson+DB+1991+Conditional+Heteroskedasticity+in+Asset+Returns:+A+New+Approach,+Econometrica+59+(2),+pp+347-370&hl=sk&as_sdt=0&as_vis=1&oi=scholart&sa=X&ved=0ahUKEwjx48mF08bPAhUDSBQKHekaCb8QgQMIGDAA
22. Orr MJL (1986) Introduction to radial basis function networks. http://www.anc.ed.ac.uk/rbf/intro/intro.html
23. Vapnik V (1990) Statistical learning theory. Wiley, New-York. http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0471030031.html
24. Zivot E, Wang EF (2005) Modeling financial time series with S-PLUS. Springer, New York. http://www.springer.com/la/book/9780387279657