2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

An Autonomous Trader Agent for the Stock Market Based on Online Sequential Extreme Learning Machine Ensemble

Rodolfo C. Cavalcante[1,2]
[1]Federal University of Alagoas
Campus Arapiraca
Arapiraca, Alagoas, Brazil 57309-005
Email: rodolfo.cavalcante@arapiraca.ufal.br

Adriano L. I. Oliveira[2]
[2]Center of Informatics (CIn)
Federal University of Pernambuco
Recife, Pernambuco, Brazil 50740-560
Email: alio@cin.ufpe.br

***Abstract*—Financial markets are very important to the economical and social organization of modern society. In this kind of market, the success of an investor depends on the quality of the information he uses to trade in the market, and on how fast he is able to take decisions. In the literature, several statistical and soft computing mechanisms have been proposed in order to support investors decision in the financial market. In this work we propose an autonomous trader agent that is able to compute technical indicators of the stock market and take decisions on buying or selling stocks. Our trader agent is based on a single hidden layer feedforward (SLFN) ensemble trained with online sequential extreme learning machine (OS-ELM), a variant of ELM that is able to learn data one-by-one and dynamically accommodate changes in the market. In addition, we propose a set of trading rules that guides the trader agent in order to improve the potential profit. Experimental results on real dataset from Brazilian stock market showed that our proposed trader agent based on OS-ELM ensemble is able to increase the financial gain when compared with other approaches proposed in literature.**

I. Introduction

Financial markets are very important to the economical and social organization of the modern society. In this kind of market, the success of investors is based on the quality of the information they use to support decision making and on how fast they take decisions. Many macro-economical factors can affect a financial market, including political situations, company policies, general economic conditions, supply and demand, investors expectations, among others [1]. The investment analysis studies all these factors in order to predict the future prices of a stock and support profitable decisions on when to buy or sell stocks. The stock market prediction problem has been widely studied in the fields of finance, engineering and mathematics in the last years due to its potential financial gain [2].

Two approaches commonly used to analyze and predict stock prices are (1) fundamental analysis and (2) technical analysis. The former approach studies all the economic factors that influence market movements, and it presents a longer term prediction spectrum. The technicians, on the other hand, believe that the price already includes all the fundamentals that affect it. In this sense, technicians usually study the historical behavior of a stock as a time series, believing that the history tends to repeat itself [3]. This modeling approach avoids the analysis of all those subjective economic factors. A time series is a set of observations collected sequentially in time [4]. Studying stock price movements as a time series presents several advantages arising from time series analysis, such as the identification of trends, the presence of seasonal effects or cycles, and outliers [5]. Technical analysis is more appropriated to short term trading, such as intraday, daily, or weekly [6].

Financial time series prediction can be considered one of the main challenges in the time series and machine learning literature [7]. In the last decades, several approaches have been proposed in order to predict stock markets and to provide decision-making support systems [8]. Two major classes of works on forecasting financial time series are statistical models and soft computing approaches [9]. Statistical models, such as time series regression, exponential smoothing and autoregressive integrated moving average (ARIMA), assume that the time series under study are generated from a linear process [10], and try to model the underlying process in order to make predictions about the future values of the series. However, financial time series are essentially complex, highly noisy, dynamic, non-linear, nonparametric, and chaotic in nature [11].

Soft computing techniques, such as expert systems, fuzzy systems and artificial neural networks (ANNs), have been applied with relative success in modeling and predicting financial time series [12]. Many soft computing techniques are able to capture nonlinear relations among relevant factors with no prior knowledge about the input data [13]. Among these techniques, ANNs have been widely used in forecasting time series, since they are data-driven, self-adaptive methods able to capture nonlinear behaviors of time series without any statistical assumptions about the data [7], [14]. Due to these advantages, several types of ANNs have been used in forecasting financial time series, such as feedforward neural networks (FFNN), support vector machines (SVM), Elman networks, probabilistic neural networks (PNNs) and neuro-fuzzy systems [13].

In this context, a fast training algorithm applied to single hidden layer feedforward network (SLFN) called extreme learning machine (ELM) was recently proposed in the literature [15]. ELM is a learning algorithm that presents better generalization performance and a much faster learning process, when compared with traditional gradient descent learning

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

methods, such as backpropagation. ELM does not suffer from some issues of other training algorithms, such as the need to adjust the training parameters, the stopping criteria, and get stuck in local minima [16]. However, despite its main advantage of fast training time, ELM is a batch learning algorithm. That means it requires that all training data are available before training [17]. In several applications, such as in financial market forecasting, the learning process should be continuous, since new data arrives time by time and should be incorporated in the learning process in order to improve the prediction accuracy. Liang et al. [18] proposed the online sequential extreme learning machine (OS-ELM), a variant of ELM able to learn data one-by-one or blocks of data in a very fast way. In theory, since OS-ELM is a sequential learning process, it will enable the learning process to incorporate trend and seasonality changes in price movements automatically.

In this work, we propose an autonomous trader agent that is able to negotiate in the financial market without human intervention. The trader agent implements an ensemble of SLFNs trained with OS-ELM to learn how to map the past values of a stock time series into future prices, and a set of trading rules that uses the predicted prices provided by the SLFN ensemble in order to identify trading opportunities in the market. The ensemble architecture is useful in order to overcome the randomness in adjusting the input weights provided OS-ELM training algorithm. We have tested our trader agent in the Brazilian stock market BM&F BOVESPA. To validate our trader agent, we compare the forecasting accuracy and financial returns obtained by it with a trading system that uses an MLP trained with the Levenberg-Marquardt algorithm and a trading system that uses an ELM ensemble.

The remainder of this paper is organized as follows. Section II describes some works related with our research. Section III explains our proposed trader agent in detail. In Section IV, the experimental results are discussed. Section V gives a summary and directions for future works.

II. Related Works

Artificial neural networks have been one of the most frequently soft computing mechanisms applied to financial forecasting problems, since they perform very well in uncertain and noisy environments [19]. In the work proposed in [20], the authors implemented a feedforward neural network trained with backpropagation to build a predictor used in a stock trading system applied to the Australian stock market. The neural network implemented uses as input four variables arising from fundamental analysis: price earning ratio (PE), book value, return on equity (ROE) and dividend payout ratio. The ANN output is a strength signal that represents the expected returns of the stock predicted. This output feeds a trading system that decides when buy or sell the stock.

In [21], the authors proposed a trading system based on a generalized feedforward neural network trained with backpropagation with momentum. The neural network is used to learn and forecast the relative strength indicator (RSI), one of the most used financial indicators in technical analysis. RSI is a momentum oscillator that measures the speed and direction of price movements, and can be used to measure the rate of rise or fall in stock prices. In that work, the generalized feedforward network is fed with RSI from different days in order to predict future RSI. The trading system uses two rules based on predicted RSI in order to decide whether to buy or sell stocks. If the predicted RSI is higher than 70, the trading system advises to sell. If predicted RSI is lower than 30, the trading system advises to buy stocks. Authors evaluated the neural network in terms of prediction accuracy of RSI. The trading system was evaluated by counting the number of success operations advised by the system (buy or sell) divided by the number of operations.

The work proposed in [6] investigated how to make profitable trades in the foreign exchange market (FOREX) using an automated trading system. The proposed system uses a feedforward multilayer perceptron (MLP) trained with Levenberg-Marquardt learning algorithm to forecast whether the currency is going up or down. A genetic algorithm is used to search for the best network topology in order to improve the forecaster accuracy. The authors proposed three trading strategies that analyze the predicted price movement of the currencies and decide in which currency pair the system will trade. Results showed that the trading system model proposed in the work achieve an annualized return of 23.3%.

In the work proposed in [22], the author investigated how to use Bayesian networks to predict stock markets. A Bayesian network is a kind of neural network in which the link weights are considered random variables and their density functions are written according to the Bayes rule. The adjustment of weights during the learning process consists of determining the probability density function for each weight. A three layered feedforward neural network is used in this work to predict stock price movements. Nine technical indicators are used as input and the predicted next day stock price is the output. In the experimental results, the proposed Bayesian network was compared with a fusion model with weighted average and with ARIMA. Results showed that the proposed forecasting method presented similar results to the compared methods in terms of forecasting accuracy. However, no trading system mechanism was proposed in order to evaluate the power of making profit by using the proposed methodology.

The work proposed in [23] implemented a day trading system that uses an MLP to learn the relationship among some technical indicators and to predict daily maximum and minimum stock prices. These predicted prices are used as input to a day trading system that operates in BM&F Bovespa. The MLP was trained with backpropagation with momentum. The system works by following the stock market in real time and uses the maximum and minimum predicted prices in order to decide the best time to trade during the day. The system defines some trading rules that are tested every 15 minutes to advise the investors to perform an enter or exit trade. If the current price is smaller than the minimum predicted price, the system advises the investor to buy the stock. If the current price is greater than the maximum predicted price, the investor is advised to sell the stocks. The goal is to buy in the minimum and sell in the maximum price during the day. More than one buy or sell operation is allowed in the same day, and these operations can occur in any order, as long as they are alternated. If the first operation (regardless of being a buy or sell) is done, the second operation is done compulsorily in the last minute of the same day. Authors claimed that the proposed

system achieved an annualized return close to 100% for the tested stocks.

III. Proposed Trader Agent

In this work, we propose an autonomous trader agent which is able to negotiate in the stock market in an autonomous way. Our trader agent is composed of two modules: a forecasting module and a decision-making module. The forecasting module is responsible for predicting the future prices and for feeding the decision-making module. This last module uses the predicted prices to decide which action to do, that is, whether to buy or sell stocks during the working day. In this section, we describe how these modules cooperate in order to make profitable trades in the market.

A. Forecasting Module

The success of a trader in the financial market is based on the quality of the information he has about the market and, specially, based on how he can predict whether the price will go up or down. In this sense, the accuracy of the trader's prediction is very important for improving stock returns. Technical and fundamental analysis provide their particular variables and approaches to study the market and to make forecasts about future prices. In this work, we use some technical indicators to model stock price movements as a time series in order to build an autonomous forecaster able to make short term predictions.

In the literature, several works have used multilayer perceptron (MLP) trained with backpropagation to learn some technical or fundamental indicators and make predictions about future market movement [13]. Backpropagation is a supervised learning algorithm that adjusts the network synaptic weight matrices based on the back propagation of the error in the output layer [24]. This adjustment is done in several epochs until an error criteria is achieved. Backpropagation is widely used in several problems, yet it has several disadvantages when applied to dynamic problems, such as financial market prediction. Backpropagation, as well as other gradient descent based learning methods, suffer from some problems, namely slow convergence, local minimum, and the need to tune several parameters such as, for example, the learning rate and the number of learning epochs [16].

In order to overcome the problems of the backpropagation training algorithm, Huang et al. [15] proposed the extreme learning machine (ELM), a learning algorithm developed to train single hidden layer feedforward neural networks (SLFNs). SLFNs have been extensively applied in many fields, such as pattern recognition, signal processing, short-term prediction and so on. The ELM algorithm randomly chooses the input weight matrix (which links input and hidden nodes) and the input biases, and analytically determines the output weight matrix of the SLFN. ELM presents better generalization performance than backpropagation with much faster learning speed [15]. In addiction, it is suitable for both nonlinear activation functions and kernel functions.

Despite all advantages of ELM, it is a batch learning algorithm, which requires that all training data are available before training. In financial market forecasting, the learning process should be continuous, since the complete data set is not available beforehand. As financial data is dynamic and nonlinear, by using backpropagation or ELM, we should repeat the training process with the past data as well as the new daily data in order to accommodate changes in price movements and in other time series properties, such as trends and seasonal factors.

Recently, Liang et al. [18] proposed the online sequential extreme learning machine (OS-ELM), a variant of ELM that can learn data one-by-one or chunk-by-chunk (blocks of data) with fixed or varying chunk size. OS-ELM combines the advantages of ELM, such as speed and generalization performance, with a sequential learning process in which, at any time, only the newly arrived single or chunk data is used for learning instead of the whole data set again [17].

In this work, we use an ensemble of single hidden layer feedforward networks (SLFN) trained with OS-ELM in the forecasting model. An ensemble is desired when using ELM and OS-ELM since the input weights and biases are chosen randomly and these parameters are not adjusted during the training. Methods that search for optimal parameters and face randomness tend to differ from one run of the algorithm to the next [25]. The different weights obtained in each run correspond to different generalizations of the learned problem. One way to solve this problem is by using a collective decision instead of an individual solution. The use of ensembles allows exploring additional information and the consensus among individuals that compose the ensemble with the goal of improving the generalization performance when compared with an individual predictor.

There are several ways to built an ensemble, such as simple averaging, voting, bagging, boosting, among others [26]. In [27], authors used an ELM ensemble to predict Quebec births time series which applied a weighted linear combination to combine individual outputs. In this work, we use the simple averaging as the ensemble method. Each individual SLFN is trained with OS-ELM using the same training data set, but set with different numbers of hidden neurons such that different SLFNs can converge to different weight configurations. The ensemble output is the average results of each individual SLFN. Despite its simplicity, the simple averaging ensemble method can achieve good results both in classification and regression problems [26].

Each individual SLFN has 33 input nodes and 2 outputs, which is an identical configuration to that used in the MLP proposed in [23]. The input data is composed by daily open, high, low and close prices and two classical technical indicators: the exponential moving average (EMA) and the Bollinger Bands (BB). The 33 input variables are as follows:

1) lowest and highest prices of the 5 previous days;
2) opening and closing prices of the 5 previous days;
3) EMA of the lowest and highest prices of the 5 previous days;
4) EMA of the opening and closing prices of the 5 previous days;
5) BB of the opening and closing prices of the 5 previous days;
6) BB of the lowest and highest prices of the 5 previous days;
7) opening price of the current day.

In [23], the authors claimed that these 33 variables used in their study were chosen based on experts knowledge and supported by a vast literature in the area. These inputs were preprocessed by simple normalization:

$$x_{norm} = \frac{x_i}{x_{max}}.$$

Preprocessing the data before feeding the ensemble has the goal of re-scaling data to a range $\{0,1\}$. The 2 outputs are the predicted maximum and minimum prices for the next day. The trader agent uses these two outputs in the decision-making module, to make profitable trades during the next day. The SLFN ensemble used in this paper is composed of 30 individual SLFNs, in which the number hidden neurons of each individual SLFN varies in a range from 10 to 300 in intervals of 10 units. In preliminary experiments, we tested ensembles with 50 and 100 SLFNs, but these strategies increase the running time meanwhile does not improve the overall results in average. We also tested other combinations of hidden neurons in each individual SLFNs, but the used strategy showed the best trade-off between computation cost and solution quality.

B. Decision-Making Module

In [23], the authors proposed a set of trading rules which uses the maximum and minimum outputs provided by an MLP trained with backpropagation to take business decisions in the market. In summary, these rules advise the investor to buy stocks when the current price reaches the minimum predicted price and advise to sell when the current price reaches the maximum predicted price, making profit between these two trades. The proposed rules also state that if just one operation is performed during the day, the investor is advised to perform the second operation in the last minute of the working day, regardless of whether the price trend is an uptrend or downtrend. Besides this, the system proposed in [23] defines a risk management based on stop-loss, which is a special trading order that indicates the tolerance of the investors to lose open positions. It is used to manage the risk involved in buying a stock.

In this work, we propose an agent that is able to autonomously trade in the market in real time. To build the decision-making module of the agent, we incorporate some rules proposed in [23], but we also define new rules in order to improve the chances of profit. The proposed trader agent perceives the environment by monitoring the market with a frequency of one minute, instead of every 15 minutes as in [23]. Every minute in the working day, the agent verifies and compares the current price of the trading stock with the maximum and minimum predicted prices provided by the forecaster module described above. We made preliminary experiments using 15 minutes frequency, but we observed that the agent could lose the exact moment in which the price reaches the maximum or minimum predicted prices, and it also lost a trade opportunity.

1) First Improvement – Confidence Intervals: The first issue encountered in the trading rules proposed in [23] is the fact that these rules work with exact values, i.e., the rules compare the predicted and current stock prices and advises the investor to trade when the current price is equal or less than the predicted minimum price or when the current price is equal or greater than the maximum. For example, if the minimum predicted price of a stock is R$ 18.20 (approximately U$ 7.80) and, during the day, the minimum price reached by the stock was R$ 18.22, no buying operation is done. Using this trading scheme, even when the forecaster presents a good performance, and the predicted prices are very close but not exactly the maximum and minimum prices the stock reaches during the day, no trades are done.

With this in mind, in our work, we use an ensemble composed by several SLFNs in which the actual output of the ensemble is an average of the predicted prices of the individuals SLFN, as described above. However, in order to avoid the problem of exact prices comparison, we use the confidence interval related with the predicted prices provided by each individual SLFN in the ensemble. To compute this confidence interval, we assume that outputs provided by each individual SLFNs arise from a normal distribution with unknown variance. Thus, during the working day, at every minute, the trader agent verifies whether the current stock price is within the confidence interval of the predicted maximum price or in the confidence interval of the predicted minimum price provided by the SLFN ensemble. We use a confidence interval with a significance level of 0.01, so we have an interval we can be 99% sure contains the true average output of the ensemble. Student's t distribution was used to compute this confidence interval. Using this new scheme, when the forecaster has a good performance, and the predicted prices are very close to the current prices reached during the day, the agent will not lose the opportunity to perform a trade.

2) Second Improvement – Lookahead: The second improvement made over the trading rules proposed in [23] was the change in the rule which prescripts a compulsory trade in the last minute when just one operation is made during the day. In essence, that can be considered a naive strategy since the stock price generally shows an uptrend or downtrend and, in some cases, it is more profitable to hold the stock or not buy in the last minute. We propose a new strategy that consists in running the forecaster module before the last minute of each day in order to obtain the prediction for the next day. The agent negotiates in the last minute based on this prediction for the next day.

Figure 1 illustrates an example where the strategy of selling stocks compulsorily in the last minute is not a good strategy. In this scenario, the agent buys one stock on day d_1 in the minimum predicted price by R$17.50 and, in that day, the price does not reach the maximum predicted price. According to the rules proposed in [23], in the last minute, the system compulsorily sells the stock by the price of R$18.00. In next day ($d_2$), the agent again buys one stock when the price reaches the minimum predicted by R$ 18.40 and sell it on the maximum predicted by R$ 18.90. The total profit on these two days is $p_1 + p_2 = 0.50 + 0.50 = R\$1.00$ (Figure 1(a)). But, using our trader agent rules, if the agent predicts that the minimum price of day d_2 is higher than the price in the last minute of the current day ($R\$18.00 < R\$18.40$), then it would know that the best action is to hold the stock to sell it in the next day. Using our strategy, the total profit is $p_3 = 1.40$ (Figure 1(b)). Moreover, the agent would perform just two operations instead of four, which reduces the trading costs.

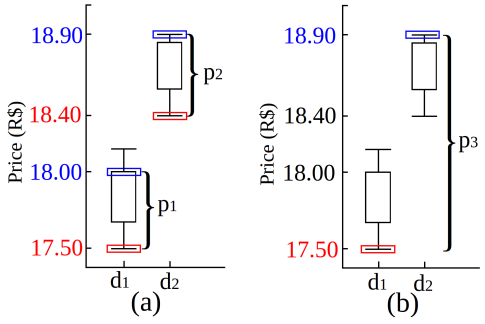

Fig. 1. (a) Strategy proposed in [23]. (b) Our strategy.

The new proposed rules are as follows. If the price in the last minute is higher than the predicted maximum price for the next day, there is a high probability that the price will fall, and the best decision is to sell the stocks or keep out of the market. If the price in the last minute is lower than the minimum predicted price for the next day, it means that the price will rise, and the best decision is to keep within the market (as one can see in Figure 1(b)). However, if the price in the last minute lies between the maximum and minimum predicted prices, the agent should observe the movements of the market to decide on the best action to perform.

Figure 2 helps to understand the case in which the closing price is between the predicted maximum and minimum prices for the next day. In this case, the price movement is in an uptrend. Given the current price in the last minute of day d_5 is equals to R$ 18.20 and the minimum and maximum predicted prices for d_6 are R$ 18.00 and R$ 18.50, respectively. As the price movement shows an uptrend, probably during d_6, the price will open near the minimum, will reach the minimum predicted price and then will increase until the maximum, closing at the end of day higher than the open price. So, if the agent has one stock and it sells this stock in the last minute of d_5 (blue mark), it gets R$ 0.60 and, in next day, it buys again one stock by R$ 18.00 in the minimum (red mark) to sell it by R$ 18.50, when the price reaches the maximum. In this case, the total profit in these two days (d_5 and d_6) is R$ 1.10.

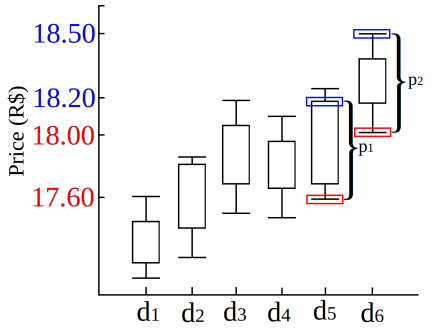

Fig. 2. Proposed strategy in an uptrend movement.

Another possible situation is the one where the price in the last minute of day d_5 is between the minimum and maximum predicted prices for the next day (d_6), but the price is in a downtrend. In this case, there is a high probability that the opening price in d_6 will be high and the maximum price will be reached before the minimum during that day, as in Figure 3. According to the strategy proposed in [23], the trading system would sell the stock in the last minute of day d_5 by the closing price R$17.60 (green mark), making a profit of R$ 0.40 in day d_5. In d_6, the price would reach the maximum price first, and system would do nothing in this point. Just when the price reaches the minimum predicted price, the system would buy the stock again, making no profit in day d_6. In our strategy, on the other hand, in day d_5, the agent computes that the price movement is in a downtrend and believes that the maximum price in day d_6 will be reached fist. Thus, the agent does not sell the stock at the end of day d_5, but sell it in d_6 when the price reaches the maximum predicted price of R$ 18.20, making a profit of R$ 1.00. Posteriorly, the agent also buys the stock again in the minimum of d_6 and it has a high probably it will sell it in the maximum of the next day (d_7). As one can see in the experiments, this new strategy increases the profit during both uptrend and downtrend movements.

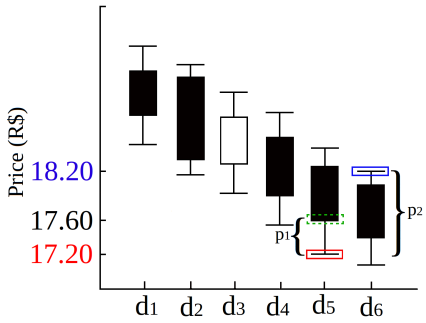

Fig. 3. Proposed strategy in a downtrend movement.

The stop-loss and start-gain rules in our trade agent are identical to that proposed in [23], which is 0.5% of the trade price. We can summary these day trading rules as follows:

1) if $price \leq min_{pred}$, buy;
2) if $price \leq stop\_loss$, sell;
3) if $price \geq max_{pred}$, sell;
4) if $price \geq start\_gain$, buy;
5) if $last\_minute$
   - if $price \geq max_{next}$, sell;
   - if $price \leq min_{next}$, buy;
   - if $min_{next} \leq price \leq max_{next}$, then
     - if $uptrend$, exit or keep outside;
     - if $downtrend$, enter or keep inside;

An important step in our strategy is how to predict whether the price is in an uptrend or downtrend movement. The heuristic used in this work consists of computing a simple moving average of the prices (SMA) and comparing with the current price. When the price is over the SMA, then it is considered in an uptrend, and when it is under the SMA, the price is considered in a downtrend. In preliminary experiments, we tested SMA varying the number of days from 2 to 20 days and the best results were achieved with a SMA of 2 days. We believe that this is due to the fact that the system uses the prediction to the next day, so it is concerned with a very short-term trend.

IV. Experiments and Results

This section reports the experiments executed in order to evaluate the performance of the proposed trader agent modules in terms of generalization performance, training time, and financial returns obtained in a real world scenario. These experiments are divided into two parts. In the first part, we evaluate the forecaster module using the mean absolute percentage error (MAPE), which measures the predictor accuracy in percentage. In the second part, we evaluate the performance of the decision-making module by computing the final capital obtained and the annualized return, a metric that is used to evaluate performance of investments with high liquidity.

A. Forecasting Evaluation

In order to evaluate the performance of the forecaster module, we implemented and compared the results of the SLFN ensemble trained with OS-ELM with a multilayer perceptron (MLP) and with an SLFN ensemble trained with extreme learning machine (ELM). All algorithms were implemented in MATLAB 7.13 in a machine with Intel i5 processor and a 4 GB DDR4 RAM. The MLP architecture has the same configuration of that used in [23]: it has the traditional three layers with 33 input neurons, 2 output neurons and the number of hidden neurons is computed by the heuristic $hidden = \sqrt{input * output}$. However, in contrast with that work, we have used the Levenberg-Marquardt to train the MLP, since the backpropagation with momentum presents a slow training process. The ensemble is composed by 30 SLFNs trained with ELM in which the number hidden neurons of each individual SLFN varies in a range from 10 to 300 in intervals of 10 units. The ensemble output is the simple average of individual SLFNs.

The dataset used in these experiments is composed by time series data from three of the most negotiated stocks in Brazilian market: Petrobras PN (PETR4), Vale PNA (VALE5) and Bradesco PN (BBDC4), which belong to different market segments, namely an oil company, a mining company and a bank, respectively. These data are in one minute frequency and correspond to a period that varies from 02 February 2009 to 25 October 2013, which totals 1171 working days. For each time series, we used the last 150 days to test and the first 1021 to train the forecasters. Figure 4 shows the plots of the PETR4, VALE5 and BBDC4 time series, respectively. The 150 testing days are in red color. As one can see, each time series has a different behavior.

In the experiments, we have executed each forecaster method 50 times for each time series and measured the averaged training time (in seconds) and averaged MAPE for the maximum and minimum predicted prices. Table I presents the results of the experiments for PETR4. One can see that in terms of computation time, the OS-ELM ensemble presents the higher costs among the three forecasters, which is approximately 54 seconds. In terms of generalization performance, OS-ELM present better results for the maximum price, but statistically similar results for the minimum predicted prices.

Table II presents the results of applying the forecasters in VALE5. In this table, one can see that training time is similar to those in PETR4. OS-ELM presents better accuracy in terms of the maximum predicted price and similar performance to the ELM ensemble for the minimum predicted price. Both OS-ELM and ELM ensembles obtained better results than the MLP. Similar results were obtained for BBDC4, as one can see in Table III.

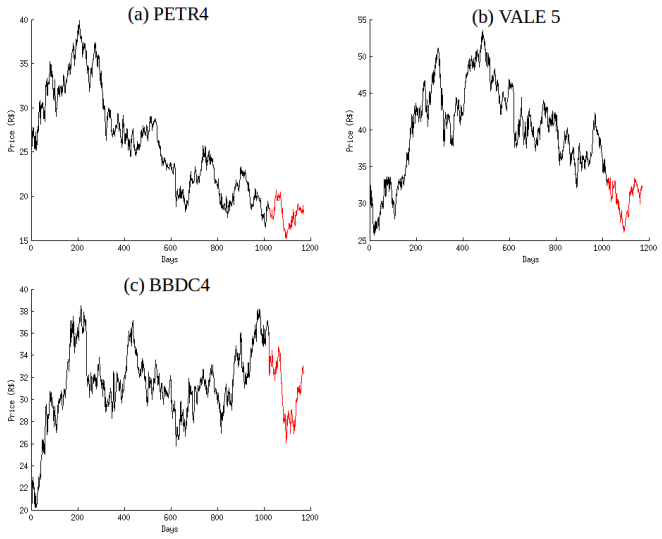

Fig. 4. (a) PETR4, (b) VALE5, (c) BBDC4 time series.

TABLE I. Comparison of the forecasters for PETR4.

	Time		MAPE max		MAPE min	
Forecaster	average	stdev	average	stdev	average	stdev
MLP	1.099	0.170	1.134	0.133	0.999	0.114
ELM Ens	1.363	0.020	1.038	0.021	1.022	0.018
OS-ELM Ens	54.74	0.063	0.918	0.121	1.016	0.682

TABLE II. Comparison of the forecasters for VALE5.

	Time		MAPE max		MAPE min	
Forecaster	average	stdev	average	stdev	average	stdev
MLP	1.147	0.253	0.890	0.054	0.911	0.047
ELM Ens	1.368	0.091	0.855	0.008	0.859	0.010
OS-ELM Ens	54.32	0.387	0.834	0.019	0.853	0.023

TABLE III. Comparison of the forecasters for BBDC4.

	Time		MAPE max		MAPE min	
Forecaster	average	stdev	average	stdev	average	stdev
MLP	1.193	0.424	0.837	0.037	0.811	0.038
ELM Ens	1.357	0.018	0.826	0.005	0.788	0.007
OS-ELM Ens	53.52	0.169	0.786	0.139	0.795	0.460

B. Profit Evaluation

In order to evaluate the decision-making module performance, we implemented day-trading system proposed in [23]. First we compare the final capital obtained by our decision-making module with that obtained by [23] by simulating a forecaster with 100% of accuracy, which we call the Oracle. This Oracle was built by using the real maximum and minimum prices of the dataset corresponding to the 150 last days of each time series instead of using the prices provided by the forecasters. We also compared these results with the buy-and-hold strategy. The initial capital used was R$ 50.000,00 (approximately U$ 21.740,00). In each trade, the agent uses

all the money available. Table IV presents a comparison of the final capital obtained by each strategy using the Oracle forecaster. One can see that our strategy achieved better results for all stocks tested. For PETR4, our strategy achieved a surplus of 19% higher than [23], 83% higher for VALE5 and 57% higher for BBDC4. In all stocks, the buy-and-hold strategy lost money.

TABLE IV. COMPARISON OF THE FINAL CAPITAL ACHIEVED BY EACH STRATEGY.

Trading strategy	PETR4	VALE	BBDC4
Buy-and-hold	R$ 49.974,00	R$ 48.380,00	R$ 44.904,00
Martinez et al. [23]	R$ 685.910,00	R$ 403.514,00	R$ 433.804,00
Our strategy	R$ 816.295,00	R$ 740.513,00	R$ 683.837,00

In the second set of experiments, we compared the power of making profit of OS-ELM ensemble with the MLP and ELM ensemble using the implementation of day-trading strategy proposed in [23]. We executed each forecaster and day-trading system 50 times for PETR4, VALE5 and BBDC4 and measured the average final capital achieved by each forecaster. Figure 5 shows a comparison of the results and associated standard deviations achieved by the day-trading system using these three implemented forecasters when applied to PETR4. In this graph, one can see that the OS-ELM obtained statistically better results than MLP and ELM ensemble.

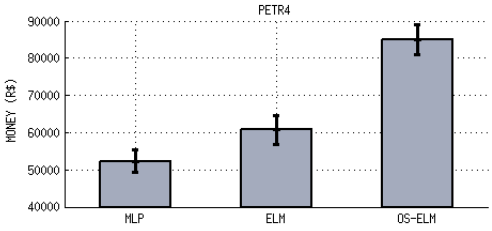

Fig. 5. Final capital achieved by trading system proposed in [23] using MLP, ELM ensemble and OS-ELM ensemble applied to PETR4.

Figures 6 and 7 present the results of experiments with VALE5 and BBDC4, respectively. In these cases, however, one can see that the OS-ELM is better than the other forecasters on average, but the results are statistically equivalent.

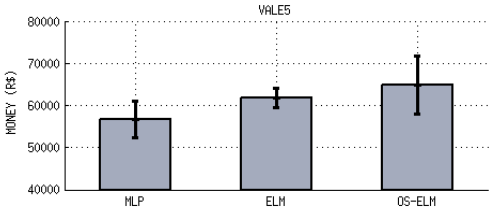

Fig. 6. Final capital achieved by trading system proposed in [23] using MLP, ELM ensemble and OS-ELM ensemble applied to VALE5.

We also made similar experiments to compare the profit obtained by each forecaster, but now applied to our proposed agent, that is, using the trading rules described in Section III. Figures 8, 9 and 10 show the results in this new scenario for PETR4, VALE5 and BBDC4, respectively. For the three

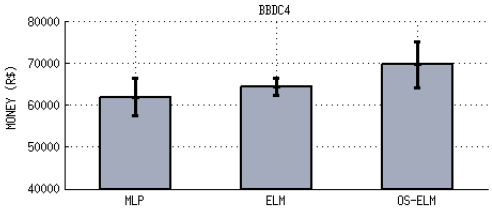

Fig. 7. Final capital achieved by trading system proposed in [23] using MLP, ELM ensemble and OS-ELM ensemble applied to BBDC4.

stocks, the final capital achieved was higher when using OS-ELM ensemble in the forecaster module. This may be due to the fact that OS-ELM is an online sequential training algorithm that is able to learn changes in time series characteristic, such as trend, seasonal effects and volatility.

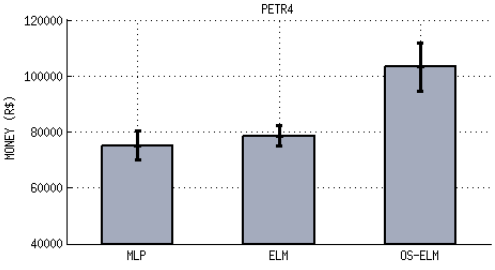

Fig. 8. Final capital achieved by our trader agent using MLP, ELM ensemble and OS-ELM ensemble applied to PETR4.

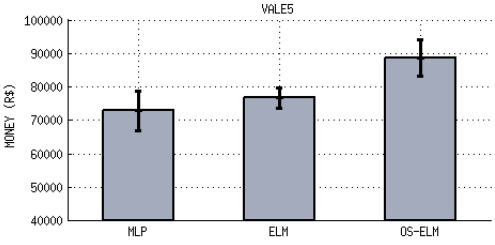

Fig. 9. Final capital achieved by our trader agent using MLP, ELM ensemble and OS-ELM ensemble applied to VALE5.

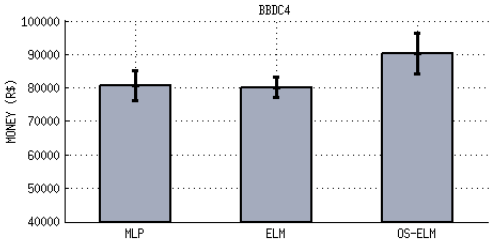

Fig. 10. Final capital achieved by our trader agent using MLP, ELM ensemble and OS-ELM ensemble applied to BBDC4.

In a final experiment, we have compared the annualized return obtained by each forecaster when applied in the day-trading system proposed in [23], called here strategy 1 (S1),

and in our trader agent, the strategy 2 (S2) for the three stocks considered here (Table V). In a horizontal analysis of the table, we can see that our agent is able to greatly improve the profit compared with the strategy proposed by [23] (S1 vs. S2). From a vertical analysis of the table, our proposal of using OS-ELM ensemble with confidence intervals remarkably increase the profit when compared with the other forecasters.

TABLE V. COMPARISON OF THE ANNUALIZED RETURN (PERCENTAGE) ACHIEVED BY EACH STRATEGY.

	PETR4		VALE		BBDC4	
Forecaster	S1	S2	S1	S2	S1	S2
MLP	12.2	183.2	36.5	154.0	69.2	221.4
ELM Ens	61.8	217.9	67.5	190.6	84.4	215.9
OS-ELM Ens	268.8	490.5	91.1	301.1	125.5	324.1

V. CONCLUSION

In this paper, we proposed a novel method to build an autonomous trader agent able to negotiate in stock exchanges. The agent is composed by two modules. The first one is a forecasting module, which is able to analyze financial time series data and predict the next day maximum and minimum prices. These predicted prices feed the decision-making module, which tracks the market in real-time and take decisions on when to buy or to sell stocks. We used an SLFN ensemble trained with OS-ELM in order to improve the accuracy of the forecaster module. Some new trading rules were proposed in order to improve the agent profit. We compared our agent with an approach proposed in the literature, which is based on MLP and with another approach based on ELM. The results showed that our idea of using the OS-ELM ensemble together with a new set of trading rules is able to achieve higher profits when compared with those methods based on MLP and ELM ensemble.

In this work, the input variables are fixed and composed by technical indicators used to train the neural networks in the forecasting module. As a future work, we aim to investigate the application of feature selection methods, in order to (1) improve the forecasting accuracy and the profit obtained by the proposed trader agent, and (2) understand the influence of each technical variable on trader performance for each stock.

REFERENCES

[1] W. Huang, Y. Nakamori, and S.-Y. Wang, "Forecasting stock market movement direction with support vector machine," *Computers & Operations Research*, vol. 32, no. 10, pp. 2513–2522, 2005.

[2] P. D. Yoo, M. H. Kim, and T. Jan, "Machine learning techniques and use of event information for stock market prediction: A survey and evaluation," in *International Conference on Computational Intelligence for Modelling, Control and Automation*, vol. 2, 2005, pp. 835–841.

[3] J. J. Murphy, *Technical Analysis of the Financial Markets*. New York Institute of Finance, 1999.

[4] P. J. Brockwell and R. A. Davis, *Time series: theory and methods*. Springer, 2009.

[5] J. D. Cryer and K.-S. Chan, *Time series analysis with applications in R*. Springer, 2008.

[6] C. Evans, K. Pappas, and F. Xhafa, "Utilizing artificial neural networks and genetic algorithms to build an algo-trading model for intra-day foreign exchange speculation," *Mathematical and Computer Modelling*, 2013.

[7] F. E. Tay and L. Cao, "Application of support vector machines in financial time series forecasting," *Omega*, vol. 29, no. 4, pp. 309–317, 2001.

[8] L. A. Teixeira and A. L. I. De Oliveira, "A method for automatic stock trading combining technical analysis and nearest neighbor classification," *Expert systems with applications*, vol. 37, no. 10, pp. 6885–6890, 2010.

[9] J.-Z. Wang, J.-J. Wang, Z.-G. Zhang, and S.-P. Guo, "Forecasting stock indices with back propagation neural network," *Expert Systems with Applications*, vol. 38, no. 11, pp. 14 346–14 355, 2011.

[10] D. A. Kumar and S. Murugan, "Performance analysis of Indian stock market index using neural network time series model," in *International Conference on Pattern Recognition, Informatics and Medical Engineering (PRIME)*, 2013, pp. 72–78.

[11] Y.-W. Si and J. Yin, "OBST-based segmentation approach to financial time series," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 10, pp. 2581–2596, 2013.

[12] M.-C. Lee, "Using support vector machine with a hybrid feature selection method to the stock trend prediction," *Expert Systems with Applications*, vol. 36, no. 8, pp. 10 896–10 904, 2009.

[13] G. S. Atsalakis and K. P. Valavanis, "Surveying stock market forecasting techniques–Part II: Soft computing methods," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5932–5941, 2009.

[14] C.-J. Lu, T.-S. Lee, and C.-C. Chiu, "Financial time series forecasting using independent component analysis and support vector regression," *Decision Support Systems*, vol. 47, pp. 115–125, 2009.

[15] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, vol. 2, 2004, pp. 985–990.

[16] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Real-time learning capability of neural networks," *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 863–878, 2006.

[17] J. Zhao, Z. Wang, and D. S. Park, "Online sequential extreme learning machine with forgetting mechanism," *Neurocomputing*, vol. 87, pp. 79–89, 2012.

[18] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1411–1423, 2006.

[19] B. Vanstone and G. Finnie, "An empirical methodology for developing stockmarket trading systems using artificial neural networks," *Expert Systems with Applications*, vol. 36, no. 3, Part 2, pp. 6668–6680, 2009.

[20] B. Vanstone, G. Finnie, and T. Hahn, "Creating trading systems with fundamental variables and neural networks: The Aby case study," *Mathematics and computers in simulation*, vol. 86, pp. 78–91, 2012.

[21] A. Rodríguez-González, Á. García-Crespo, R. Colomo-Palacios, F. Guldrís Iglesias, and J. M. Gómez-Berbís, "CAST: Using neural networks to improve trading systems based on technical analysis by means of the RSI financial indicator," *Expert Systems with Applications*, vol. 38, no. 9, pp. 11 489–11 500, 2011.

[22] J. L. Ticknor, "A Bayesian regularized artificial neural network for stock market forecasting," *Expert Systems with Applications*, vol. 40, no. 14, pp. 11 5501–11 5506, 2013.

[23] L. C. Martinez, D. N. da Hora, J. R. de M Palotti, W. Meira, and G. L. Pappa, "From an artificial neural network to a stock market day-trading system: A case study on the BM&F BOVESPA," in *International Joint Conference on Neural Networks (IJCNN)*, 2009, pp. 2006–2013.

[24] S. S. Haykin, *Neural networks: a comprehensive foundation*. Prentice Hall Englewood Cliffs, NJ, 2007.

[25] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, 1990.

[26] V. Tresp, "Committee machines," in *Handbook for neural network signal processing*. CRC Press, 2001, pp. 1–18.

[27] M. van Heeswijk, Y. Miche, T. Lindh-Knuutila, P. A. Hilbers, T. Honkela, E. Oja, and A. Lendasse, "Adaptive ensemble models of extreme learning machines for time series prediction," in *International Conference on Artificial Neural Networks*. Springer, 2009, pp. 305–314.