DEEP LEARNING FOR STOCK MARKET TRADING: A SUPERIOR TRADING STRATEGY?

D. Fister[*], *J.C. Mun*[†], *V. Jagrič*[‡], *T. Jagrič*[‡]

Abstract: Deep-learning initiatives have vastly changed the analysis of data. Complex networks became accessible to anyone in any research area. In this paper we are proposing a deep-learning long short-term memory network (LSTM) for automated stock trading. A mechanical trading system is used to evaluate its performance. The proposed solution is compared to traditional trading strategies, i.e., passive and rule-based trading strategies, as well as machine learning classifiers. We have discovered that the deep-learning long short-term memory network has outperformed other trading strategies for the German blue-chip stock, BMW, during the 2010–2018 period.

Key words: *LSTM networks, machine learning, automated stock trading*

Received: December 19, 2018 **DOI:** 10.14311/NNW.2019.29.011
Revised and accepted: June 30, 2019

1. Introduction

In stock markets, optimal trading decisions made at the right time are crucial for successful investment strategies. In order to reduce the risk and maximize profit, investors try to obtain as much information as possible–company and market data, qualitative and quantitative data–and they often equip themselves with real-time trading systems [13], recommendation systems [64], and simulators [63]. Typically, trading systems are classification algorithms, which deal with labeling the predictor variables into classes [8]. There are many different trading systems: daily trading system [21], fuzzy logic rules trading system [10], and others [14, 22, 30, 51].

Trading systems are used by large corporations in real time, in real life, for trading with real stocks. This can be risky, expensive, and very time-consuming, since it may take years for a trading system to show its strengths. Fortunately trading systems can also be used in virtual time by simulating the actual past stock market behaviour. Using this approach, we propose a mechanical trading

*Dušan Fister – Corresponding author; University of Maribor, Faculty of Economics and Business, Razlagova 14, SI-2000 Maribor, Slovenia, E-mail: dusan.fister1@um.si

†Johnathan C. Mun; Naval Postgraduate School, Monterey, California USA, 1 University Circle, Monterey, CA 93943, E-mail: jcmun@realoptionsvaluation.com

‡Vita Jagrič; Timotej Jagrič; University of Maribor, Faculty of Economics and Business, Razlagova 14, SI-2000 Maribor, Slovenia, E-mail: vita.jagric@um.si, timotej.jagric@um.si

system (MTS) to sequentially (day-by-day) execute trading signals (decisions), and thus trade with real stocks [66] in virtual time. Consequently, the quality and effectiveness of a trading system over a period of a few years can be evaluated in a matter of seconds.

The MTS works by giving three common trading signals, i.e., buy, hold, and sell, for each of the stocks. It is given an initial amount of cash to buy stocks that can be held in the portfolio or be sold at a later date. Buying and selling stocks can generate profits if trading signals are given rationally or generate loss if they are not. Profits and losses can be monitored and reviewed at the end of the trading period to realize strengths and weaknesses of the trading decisions [47]. Trading decisions are typically given by trading strategies, i.e., automated algorithms that constantly monitor market behaviour and react accordingly. Multiple trading strategies are usually incorporated within the trading system, by each giving unique trading decisions. The latter can be evaluated quickly and inexpensively either in real-world time or using the MTS.

In this paper we are proposing a concept of an automated, single stock, trading system using the MTS, examining its potential by five different trading strategies, realizing their strengths and weaknesses, and proving the correctness of an optimal strategy. Based on this, we encourage further analysis and the design of an automated portfolio trading system for many similarly treated stocks in parallel. Here we employ three different trading strategies: (1) passive, (2) rule-based – relative strength index (RSI) and moving average convergence/divergence (MACD) technical indicators, and (3) surrogate model trading strategies using machine learning classifiers (MLC) and long short-term memory network (LSTM). Each of them can provide three trading signals: buy, hold, or sell.

Since deep learning (DL) [37] is primarily intended for engineering applications, such as image and sound processing, we have not found many examples of DL in the fields of finance, banking, or insurance. In line with this, we would like to apply the DL to the area of finance, particularly mechanical trading systems as an alternative, and thus test whether the DL can be successfully deployed into this area.

The structure of the paper is as follows: Section 2 presents the literature review, MLC applications and applications using LSTMs. Section 3 defines the methodology used in this paper. Section 4 details the dataset handling, its modifications, and dataset split. Section 5 presents the results and concludes the paper.

2. Literature review

In 1970, Fama [24] introduced the Efficient Market Hypothesis (EMH), arguing that any stock price at any time fully reflects all available information. Generally, three forms of market efficiency have been proposed: weak, semi-strong, and strong [52], implying that stock prices follow a random walk. Later researchers have shown that stock prices may not follow the random walk perfectly, but are more-or-less non-linearly related to past data [14]: [4] studies the mean reversion process compared to stock price time series; [38] empirically proves that stock prices in Japan from 1999 to 2007 do not follow EMH and thus offer arbitrage options for investors; [40] rejects the random walk model for the observed stock prices. We have found that stock markets, indeed, offer arbitrage opportunities, which may lead to arbitrage profits [16]. Additionally, [2] reviews many stock market trading techniques, variables, and statistical tests. There have been several studies concerning the German stock markets, e.g., relations between beta and realized returns [23], liquidity [31], volatility and short selling constraints [6], or behavioural finance [43].

For capturing those arbitrage opportunities, machine learning [55] can be applied, similarly to its use in other diverse application areas such as medicine [67], bank marketing [32], credit rating systems [12], and bankruptcy prediction [57]. The outstanding benefit of allowing the model to learn by itself during an iterative process makes machine learning algorithms very usable for arbitrage. Generally, three types of machine learning exist: supervised, unsupervised, and reinforcement learning [35, 36, 46], where one of the supervised tasks is classification. There are numerous machine learning classifiers, such as support vector machine [19], bagging predictors [7], random forests [9], and Fisher discriminant analysis [42]. Another group of machine learning algorithms presents neural networks, e.g., long short-term memory networks [28], which can be employed for classification purposes as well. In the past, those have been applied extensively for researching stock markets [54], econometrics [3], financial market predictions [25], and electricity price predictions [49].

3. Methodology

Following the idea in [44] we developed an MTS with passive, rule-based, and surrogate model trading strategies [33]. Initially (on the first trading day), each strategy gets 10,000.00 EUR in cash, which can be spent arbitrarily for buying stocks. For each transaction, trading costs apply, which include commissions, bid-ask spreads, market impacts, timing costs, and opportunity costs [17]. In agreement with [58, 59, 62, 65], transaction costs, which are 1% of the stock's value, are used. We incorporate the assumption of perfect market liquidity so that stocks can be bought and sold instantly. By giving the trading signals, MTS buys and sells stocks for each trading strategy and thus controls its portfolio/capitalization value. Short selling is not allowed in any of the trading strategies.

A passive trading strategy is the simplest trading strategy to apply: buying the stock on the first trading day and holding the stock until the last trading day. Hence, neither the stock's current nor historic performance data is necessary. This strategy results in profit if the stock price rises during the observed period and results in loss if the price falls. Rule-based trading strategies are, on the other hand, active trading strategies, since their decisions depend on calculated technical indicators from historic stock prices. Two popular technical indicators, RSI and MACD (see e.g., [15]), have been used in our paper. Another active trading strategy is the surrogate model, where two variants are tested: MLC and LSTM. Exceptionally, those both require a parallel surrogate model to be built before the evaluation, which is typically an iterative (learning) procedure to lower the error between the real world and surrogate model. The more the surrogate model imitates the real world, the better it is [60].

3.1 Machine learning classifiers

MLC trading strategy [35] brings a high level of reliability because it is based on several algorithms that are tested simultaneously: decision trees [50], linear discriminant analysis by Fisher [18], support vector machines [5, 19], k-nearest neighbors (k-NN) [1], and ensemble learning methods [45]. Three different branches of decision trees are used in our MLC trading strategy: fine, medium, and coarse, which differ according to their complexity, i.e., the number of splits [26]. Several support vector machines (SVMs) are tested: linear, quadratic, and cubic SVM, as well as fine Gaussian, medium Gaussian, and coarse Gaussian. The group of k-NN is tested by six types: three according to complexity (fine, medium, and coarse) and three according to distance metrics (cosine, cubic, and weighted). Ensemble learning classifiers are classification algorithms that merge various classifiers into an ensemble. Three different tree ensembles are used (boosted, bagged, and RUS-Boosted trees) [56] and two types of subspaces (subspace k-NN [27] and subspace discriminant) are used. The optimal MLC among the list is chosen next.

3.2 Long short-term memory networks

LSTMs were proposed from recurrent networks (RNNs), designed by Rumelhart, Hinton, and Williams [53] in the late 1980s. LSTMs and RNNs are feedback-looped networks and thus contain two inputs: new information from the dataset and previous output [29]. Both of them allow the information to persist from one input to another [48]. Common RNNs suffer from the so-called vanishing or exploding gradients, which makes them unable to memorize the long-term dependencies between data. The problem that arises is the gradient that diminishes or explodes exponentially. This makes the network dependent on temporal components, which may make such networks unable to account for inputs from several past steps and thus practically cause it to quickly lose information from the past.

LSTMs have been proposed to prevent this problem by reducing the information attenuation [11], which makes them especially suitable for researching the long-term dependencies in stock market prices. Although more on LSTMs can be found in [11, 20], basic working principle of LSTMs are presented in the following paragraphs.

Each LSTM unit consists of a core component, memory cell, and the three gates: forget gate, input gate, and output gate. Memory cell, or the heart of the LSTM, holds historic information learned in the past, while the three gates assure its manipulation: (1) filtration of historic information, (2) contribution of a new information, and (3) exploitation of the learned information. Filtration is performed using the forget gate, which directs information to forget and to persist. Contribution is performed using the input gate, which introduces the new information to the existing information contained in the memory cell. Exploitation is provided by an output gate, which generates (predicts) the output variable. The memory cell and three gates are illustrated in Fig. 1.

The forget gate removes the information from the memory cell by masking individual components of a vector $\mathbf{C}_{t-1}$. Masking vector $\mathbf{f}_t$ is generated by taking the sigmoid function of a previous output $\mathbf{h}_{t-1}$, concatenated with the current input $\mathbf{x}_t$. Sigmoid function limits the concatenated vector by 0 and 1, where 0

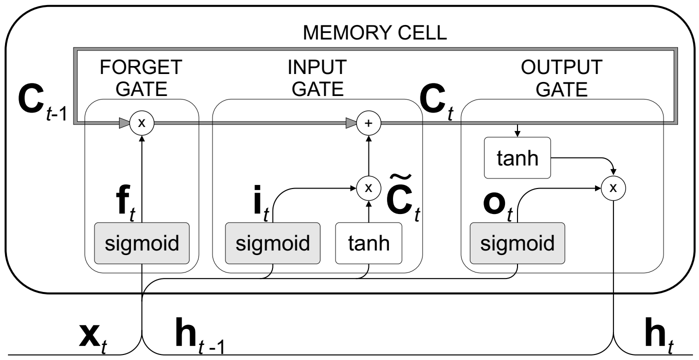

Fig. 1 *A memory cell is the heart of the LSTM unit. Source: Authors and [11].*

totally blocks the information to remain in the memory and 1 totally persists the information in the memory. Typically, analog values between 0 and 1 appear.

The memory cell thus temporarily becomes $\mathbf{f}_t \cdot \mathbf{C}_{t-1}$ before a contribution to the memory by the input gate occurs. First, the masking vector $\mathbf{i}_t$ is generated, which is next multiplied by a cell candidate $\tilde{\mathbf{C}}_t$. The latter is specified by a tanh function of a concatenated vector. Using the masking vector $\mathbf{i}_t$, specific components are only let through after the multiplication to contribute to the memory cell and form the vector $\mathbf{C}_t$. From the latter, the output $\mathbf{h}_t$ can be generated by taking the tanh function of the $\mathbf{C}_t$, and selecting specific components set by an output masking vector $\mathbf{o}_t$.

The LSTM unit is typically not self-standing, but it is incorporated into neural architecture. We have used the following architecture: (1) sequence input layer, which imports the data from dataset, (2) LSTM layer, (3) fully connected layer, which extracts information from LSTM, (4) softmax layer, which transforms extracted information, and (5) classification output layer, which is used to compare output with class labels and drive the training process.

The memory cell and the three gates constitute the majority of the LSTM network, while the rest of the LSTM consists of so-called weights: input weights $\mathbf{W}$, recurrent weights $\mathbf{R}$, and biases $\mathbf{b}$ [41]. Input weights $\mathbf{W}$ are assigned for each gate unit: forget gate, input gate, output gate, and a cell candidate [41]. Similarly, recurrent weights $\mathbf{R}$ and biases $\mathbf{b}$ are defined for each listed gate unit and a cell candidate. The forward propagation thus progresses in the following manner (for the input gate): $\sigma \cdot (W_i \cdot \mathbf{x}_t + R_i \cdot \mathbf{h}_{t-1} + b_i)$, where σ presents a sigmoid function; W_i, input gate weights; and R_i, recurrent input weights.

The learning capacity of the LSTM is defined by the number of hidden units N_{units} that define the dimension of the vector $\mathbf{h}_t$ and weights $\mathbf{W}$, $\mathbf{R}$, $\mathbf{b}$ [41]. The latter are learnable, which means that their values adjust during the training process. The training process is carried out by an optimizer, typically a gradient descent, which is employed to find the optimal weights and biases. Optimality is searched for using the objective function to lower the error between actual and predicted labels. In our case we have used a stochastic gradient descent optimizer with adaptive learning rates, called Adam [34]. Here the gradient is described with two higher moments: the first moment, or mean, and the second moment, or uncentered variance. Additionally, the exponential moving average over the moments is calculated, and both of them are corrected for bias to drive the training process.

3.3 Synthesis of a trading system

Fig. 2 presents the structure of our trading system. First, the original dataset ($\mathbf{x}$) is downloaded, preprocessed, and checked for any nonsense values, missing data, and other misconceptions. Market data (DAX30 blue-chip market index) is additionally included along with the original dataset and response variable. Next, a response variable ($\mathbf{y}$) is generated for the purpose of learning surrogate model trading strategies. The resulting dataset is widened by technical analysis variables to form a modified dataset ($\mathbf{x}^*$) that is finally split into two mutually exclusive datasets: in-sample and out-of-sample. The first one (in-sample) is used for training the MLC and LSTM models and their evaluation, while the second is for evaluation only.

Although passive and rule-based trading strategies do not require a dataset split, it is done anyway to bring a fair evaluation of passive and rule-based trading strategies in comparison with surrogate model trading strategies.

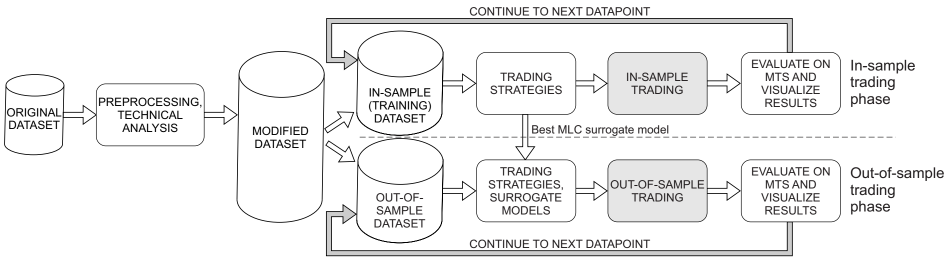

Fig. 2 *Trading system. The original dataset is preprocessed and technical analysis is executed to form a modified dataset. The in-sample and out-of-sample datasets are generated next and proceeded to in-sample and out-of-sample phases. The in-sample trading phase is responsible for building the surrogate models, while the out-of-sample is tasked with exploiting them. Next, the best MLC surrogate model among all is proceeded to in the out-of-sample trading phase. Evaluation on MTS is performed separately: first the in-sample dataset and then the out-of-sample dataset is evaluated. Source: Authors.*

4. Dataset

The original dataset was downloaded from Yahoo! Finance [61] using the GitHub library by Lenskiy [39]. A German DAX30 blue-chip stock "Bayerische Motoren Werke AG" (BMW) was chosen for the analysis because (1) it is listed on the European stock markets; (2) is relevant to European investors; (3) is intrinsically related to the German national economy and other European countries; (4) is an important economic partner in Slovenia as well as of interest to Slovenian investors; and (5) is known to be a global company with a long tradition and high liquidity of its stocks. BMW is mainly disclosed with high market capitalization: approximately 48 billion EUR with more than 657.6 million shares outstanding at the time of writing as well as high daily volume, i.e., more than 2 million transactions daily. The following data variables were downloaded for the time period from 1 January 2010 to 9 November 2018: daily close price, daily open price, highest daily price, lowest daily price, daily adjusted close price, and daily volume. The original dataset consists of $N = 2247$ datapoints with $d^{(\mathrm{orig})} = 6$, where each datapoint represents a single trading day and $d^{(\mathrm{orig})}$ presents the dimension of the original dataset. Tab. I presents the descriptive statistics of the original dataset, where "Adj. Close" presents the daily adjusted close and Jarque-Bera shows a statistical test for testing the normality of data. One-day returns on close prices have been tested for normality, and it has been assessed that none of the variables are normally distributed. Although the variables are non-stationary, we perform the Jarque-Bera test to remind readers that non-normality (and non-stationarity as well) of the variables makes the standard econometric approaches for the analysis of such data difficult. Since we do not have access to intra-daily stock prices, we assume that close prices do not vary the last few minutes of the trading day. Accordingly, we may buy or sell the stock prior to closing the market with the close price already obtained. Therefore, we react as trading still continues.

Variable	Min	Max	Median	Mean	Stdev	Jarque-Bera
Close	28.65	122.60	78.01	74.37	17.59	469.61
Open	28.90	123.30	78.20	74.40	17.63	733.04
High	29.29	123.75	78.90	75.21	17.70	447.08
Low	28.28	120.35	77.11	73.48	17.48	524.01
Adj. Close	21.59	105.03	68.91	64.25	18.43	467.73
Volume	372,209	15.6E6	1.9E6	2.1E6	1.1E6	19,812
Market	5,072	13,560	9,447	9,220	2,340	656

Tab. I *Descriptive statistics of the original dataset. According to the Jarque-Bera test, none of the variables are distributed normally. It is worth mentioning that none of the variables are stationary either.*

4.1 Response variable generation

A response variable ($\mathbf{y}$) is a categorical vector that incorporates three common trading signals: "Buy," "Hold," and "Sell." The response variable is analytically derived from the stock's close prices. Those are due to the best expressing market activity and investors' supply and demand used preferentially instead of adjusted close prices, which account for dividends and stock splits, and consequently do not represent direct market activity.

A response variable is calculated for each time horizon n. In this paper five time horizons $n = 1, 2, \ldots, 5$ are used, which apply five unique n-day returns. Those are symbolized by Δ_n and are calculated as in Eq. 1. Afterwards, the response variable is assigned according to the value of Δ_n, by Eq. 2

$$\Delta_n = \frac{x_t^{(\text{close})}}{x_{t-n}^{(\text{close})}} - 1, \tag{1}$$

$$y_t^{(n)} \begin{cases} \text{Buy}; & \Delta_n > 0.05 \\ \text{Hold}; & -0.05 < \Delta_n < 0.05 \\ \text{Sell}; & \Delta_n < -0.05 \end{cases}, \forall n = 1, 2, \ldots, 5, \tag{2}$$

where $x_t^{(\text{close})}$ presents the actual close price and $x_{t-n}^{(\text{close})}$ the close price n-days ago. If the quotient between those Δ_n exceeds the positive/negative threshold, set at +/- 0.05, the response variable y_t becomes "Buy," or "Sell" for day t. Otherwise, the response variable is assigned as "Hold." We additionally assume that once the stock is bought or sold, no further transactions are allowed following n-days. This type of response variable is used for supervised learning.

4.2 Data preprocessing and technical analysis

Technical analysis, shown in Fig. 2, is used to widen the original dataset ($\mathbf{x}$) and thus incorporate new information into the model. According to Tab. II, the resulting modified dataset ($\mathbf{x}^*$) consists of 43 variables. The following technical indicators were employed (for different time horizons): n-daily return (RET), difference in daily return (DIFF), relative strength index (RSI), moving average convergence divergence (MACD), inclination of daily returns trend (INCL), changes of returns (CHG_RET), differences between inclinations (DIFF_INCL), coefficients between stock prices (COEF), and market returns (MARKET_RET) with market inclinations (MARKET_INCL).

Stock and market data come in the form of coefficients of averages with date data in the integer form. The coefficient of average is calculated as a variation from the average value of a specific explanatory variable. Inclinations are calculated as regressions over 5, 10, 15, and 20 samples of daily returns. Differences between inclinations are derived as follows: 5-day inclination from 5 days ago is subtracted from current 5-day inclination; furthermore, 10, 15, and 20 days inclinations are subtracted from current 5-day inclination. Coefficients between stock prices are derived as relative coefficients, between high and low prices and relative coefficients between close and open prices. In the last step, linear transformation is employed to make the learning process of the LSTM easier. It does not change the information criteria, nor the objectivity of the dataset. Linear transformation is performed using the fixed scaling factors, prior to the split, which makes both the in-sample and out-of-sample datasets fixed and known in advance. Despite this fact, they are not initially shown to the surrogate model.

In fact, the MTS makes the continuous updates by updating new datapoints. It always takes only one datapoint at a time (one trading day), makes a decision, and then proceeds to the next trading day. In this way, MTS proceeds through all of the in-sample and out-of-sample trading decisions and simultaneously measures capitalization performance.

Explanatory variables	No. of var.
1. Stock and market data:	
open, close, high, low, adj. close, volume, market index	7
2. Date data:	
year, month, day, day of week, days to next trading day, days from previous trading day	6
3. Technical indicators:	
RET: $n = \{1, 2, 3, \ldots, 10\}$-day period	10
DIFF: $n = \{1, 2\}$-day period	2
RSI: 14-day period RSI, standardized	1
MACD: 12-day short, 26-day long and 9-day signal period	1
INCL: $n = \{5, 10, 15, 20\}$-day period	4
CHG_RET: $n = \{1, 2\}$-day period	2
DIFF_INCL	4
COEF	2
MARKET_RET: $n = \{1, 4\}$-day period	2
MARKET_INCL: $n = \{5, 10\}$-day period	2
Sum	43

Tab. II *The modified dataset consists of 43 explanatory variables (features), which are divided into three categories: stock and market data, date data, and technical indicators. Stock, market and date data are unique features, while the technical indicators are derivatives of those two features. Each explanatory variable is fed into the trading system.*

With the removal of a few samples, the final size of the modified dataset is reduced to $(\mathbf{x}^*, \mathbf{y}) = 2222 \times 43$. The modified dataset is split fifty-fifty, i.e., half of the datapoints ($N = 1111$) become in-sample and the other half, out-of-sample. Such a long out-of-sample is an unusual practice, but it is employed anyway to eliminate any coincidences (random effects) that might show to be beneficial during evaluation. By using such a practice, redundancy and generalization of surrogate model is desired to make sure that trading strategies work systematically.

5. Results

The purpose of our experimental work was to compare the presented trading strategies with different time horizons for automated stock trading. BMW stock was taken for the analysis. The in-sample ranges were taken from 8 Feb 2010 to 20 Jun 2014, while out-of-sample ranges were taken from 23 Jun 2014 to 9 Nov 2018. Implementation of the described MTS and its features was done in the MATLAB programming environment. A computer with the Ubuntu Linux operating system and Intel Core i5-2430M @ 2.40 GHz with 8 GB RAM was used. Tab. III presents the parameter settings of each MLC and LSTM, where $S^{(\max)}$ represents the maximum number of splits for each branch of the decision tree; k, the number of nearest neighbors; and D, subspace dimension. Cosine and cubic metric represent alternative distance metrics, while k_{scale} is a kernel scale parameter, where "auto" means its automated selection. The number of learners N_{learners} parameter in bagged trees represent the complexity of an ensemble.

No.	MLC method	MLC parameter	MLC setting
1.	Fine decision tree	max. splits $S_f^{(\max)}$	100
2.	Medium decision tree	max. splits $S_m^{(\max)}$	20
3.	Coarse decision tree	max. splits $S_c^{(\max)}$	4
4.	Discriminant analysis	covariance structure	full
5.	Linear SVM	kernel scale k_{scale}	auto
6.	Quadratic SVM	kernel scale k_{scale}	auto
7.	Cubic SVM	kernel scale k_{scale}	auto
8.	Fine Gaussian SVM	kernel scale k_{scale}	1.6
9.	Medium Gaussian SVM	kernel scale k_{scale}	6.6
10.	Coarse Gaussian SVM	kernel scale k_{scale}	26
11.	Fine k-NN	no. of neighbors k	1
12.	Medium k-NN	no. of neighbors k	10
13.	Coarse k-NN	no. of neighbors k	100
14.	Cosine k-NN	neighbors k, cosine metric	10
15.	Cubic k-NN	neighbors k, cubic metric	10
16.	Weighted k-NN	neighbors k, distance weight	10
17.	Boosted trees	max. splits $S_{\mathrm{boost}}^{(\max)}$	20
18.	Bagged trees	no. of learners N_{learners}	30
19.	Subspace discriminant	no. of dimensions D_s	22
20.	Subspace k-NN	no. of dimensions D_s	22
21.	RUSBoosted trees	max. splits $S_{\mathrm{RUS}}^{(\max)}$	20

	LSTM parameter	LSTM setting
1.	No. of hidden units N_{units}	7
2.	Batch size N_b	1111
3.	Maximum epochs $N_e^{(\max)}$	2000
4.	Initial learning rate $L_r^{(\mathrm{init})}$	0.05

Tab. III *MLC parameter settings are listed on the upper portion of the table, while LSTM parameter settings are on the lower.*

Before conducting the experiments, the LSTM neural architecture was tested for overfitting problems. Various architectures regarding the LSTM units were tested, e.g., $N_{\mathrm{units}} = 1000, 500, 200, 100, 50, 20, 15$, but all of them signaled overfitting problems. In order to ensure the greatest generalization, we reduced to $N_{\mathrm{units}} = 7$. Fig. 3 presents the architecture of neural network and the dimensions of each layer.

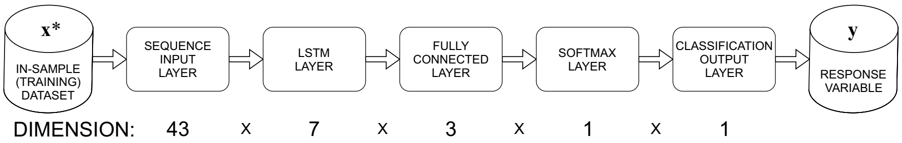

Fig. 3 *Architecture of neural network: sequence input layer, LSTM layer, fully connected layer, softmax layer, and classification output layer. Source: Authors.*

Experiments and results are divided into three subsections: (1) the effect of time horizon, a where graphical comparison is placed; (2) a detailed tabular comparison; and (3) the discussion.

5.1 Effect of time horizon

The effect of time horizon is analyzed first. Here five different time horizons are tested, $n = 1, 2, \ldots, 5$-day returns, where trading results are shown for both in-sample and out-of-sample. However, a fair evaluation is only produced by the out-of-sample. First, the most common daily stock return, i.e., 1-day, is tested by taking today's and yesterday's stock performances. Such an instance is the most responsive by giving trading signals quickly. Fig. 4a presents the in-sample results for 1-day stock returns, and Fig. 4b shows the out-of-sample results, revealing the

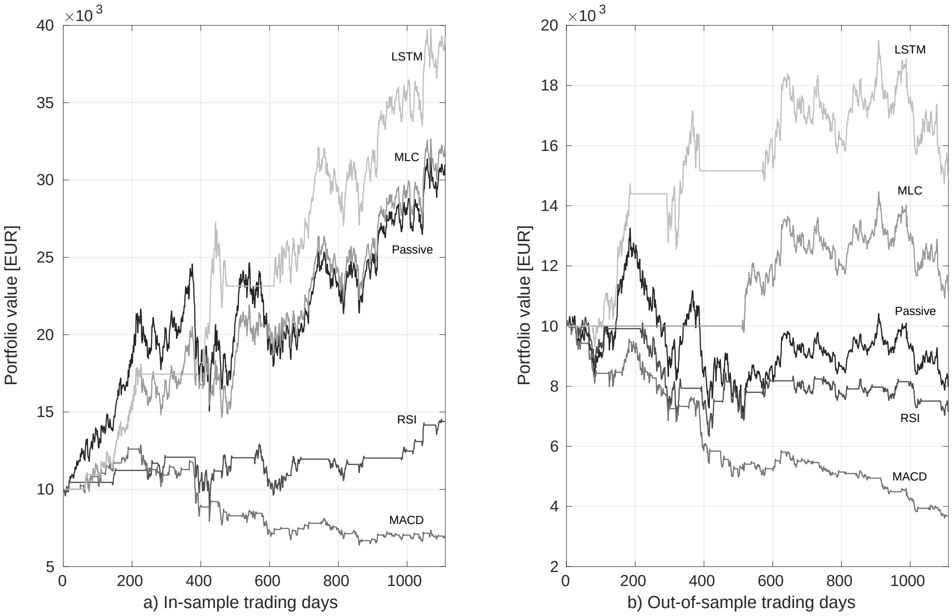

Fig. 4 *Trading results for 1-day stock returns. The left figure shows results of the in-sample trading strategy, while the right figure shows the out-of-sample trading strategy. LSTM performs best for in-sample and out-of-sample. Source: Authors.*

results of all five trading strategies. In both cases, LSTM performed the best. Stock price naturally rises heavily in the in-sample period and falls majorly in the out-of-sample period. Overall, two things are observed: first, out-of-sample profits are much lower than in-sample, and, second, benchmarking strategies perform worse. The capitalization value diminishes from the initial value in both passive and rule-based strategies. Although the use of LSTM benefits (profit almost exceeds 40 % of initial capitalization value), its potential is not exploited fully. MLC strategy, on the other hand, significantly falls behind, but still performs better than the passive strategy. Here, discriminant analysis is chosen as an optimal MLC method. Fig. 5 (upper) presents the results of 2-day stock returns' time horizon, where the response variable is calculated as the difference between current price and the price two days ago. LSTM performs by far the best among all strategies, because it results in profits, while all the other strategies end up with lower capitalization value than when they started. The fine tree is used as the optimal MLC method. Fig. 5 (lower) presents the results of using the 3-day stock returns, where another outstanding LSTM performance occurs. MLC performs better on the 2-day time horizon and ends up with only about half of the LSTM's profit on the 3-day time horizon. The fine tree is the optimal MLC method here as well. Fig. 6 (upper) presents the results for the 4-day stock returns, where LSTM vastly surpasses the other strategies on out-of-sample. MLC again employs a loss. This time the cubic SVM method is found to be the optimal MLC method. Fig. 6 (lower) presents the results of the 5-day time horizon, which is closely related to the 4-day time horizon for in-sample. Out-of-sample indicates that increasing the time horizon n further might decrease LSTM performance. Nevertheless, it performs significantly better than RSI and MACD rule-based strategies, performing solidly in the beginning of the trading period, but later losing the benefits gained. Here the medium decision tree is the optimal MLC method.

5.2 Detailed comparison among trading strategies

Capitalization values in this subsection are shown in a tabular fashion. Three comparisons are presented and discussed: (1) relative capitalization values, (2) number of transactions, and (3) computational time requirements. Tab. IV shows the final capitalization values for each of the trading strategies at each time horizon. Both the MLC and LSTM profits are highest for 2-day stock returns at in-sample and decrease afterwards. Decreasing profits apply for LSTM after 2-day stock returns on out-of-sample as well, but not for MLC, which performs best for 3-day stock returns. Passive, RSI, and MACD strategies perform consistently for all time horizons.

Tab. V shows the number of transactions. Those are held constant for passive, RSI, and MACD trading strategies, but vary significantly for MLC and LSTM strategies. The shortest time horizon provides the minimal out-of-sample transactions that could be caused by a fixed level of response variable. By decreasing the level (currently set at +/-0.05), the number of transactions would increase, but increasing too high might negatively affect the capitalization value. Generally, no pattern can be observed for the number of transactions, but by simplifying, we

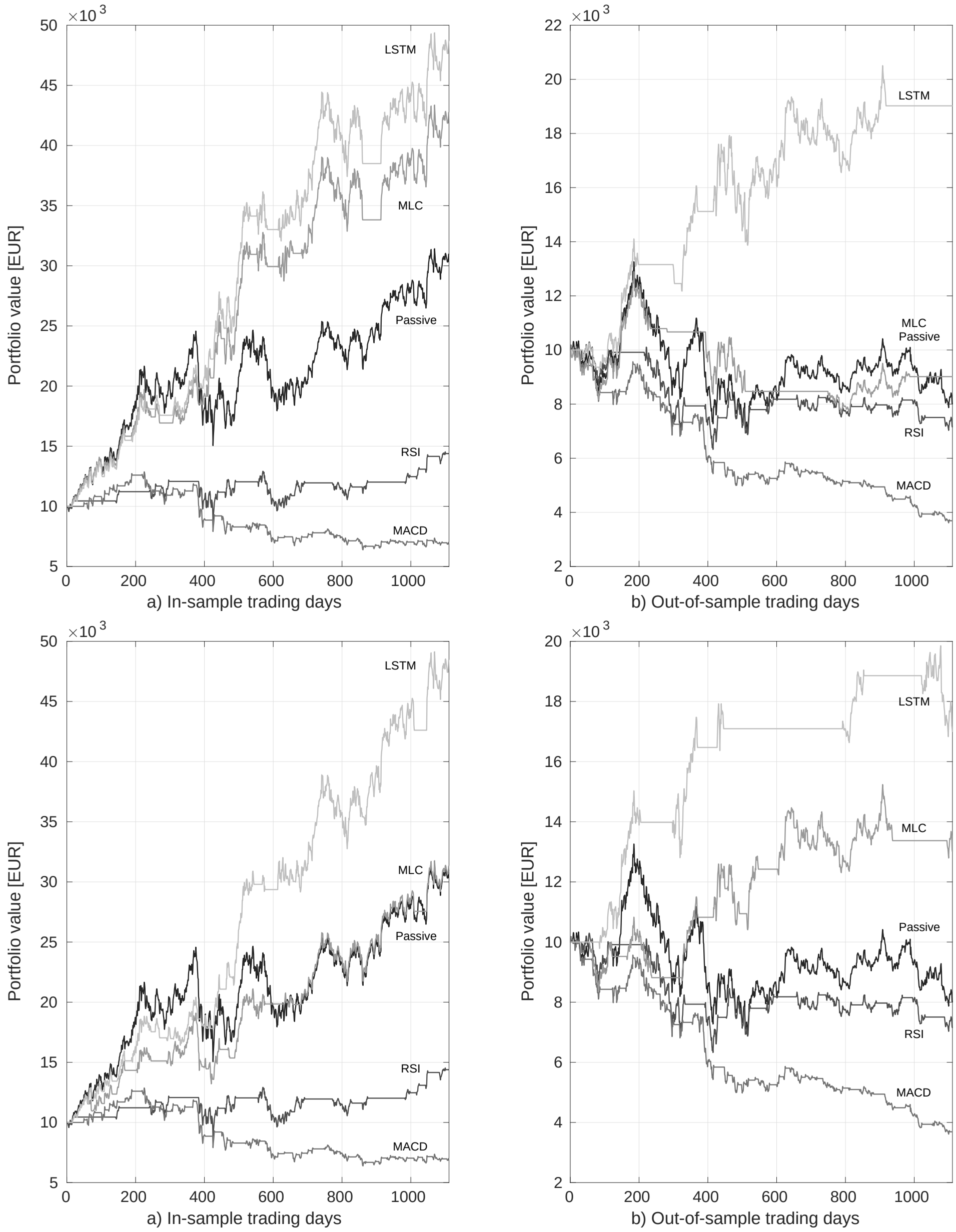

Fig. 5 *Trading results for 2-day (upper portion) and 3-day (lower portion) time horizon. The left figures show the results of the in-sample trading strategy, while the right figures show the out-of-sample trading strategy. LSTM performs best for both the in-samples and out-of-samples. Source: Authors.*

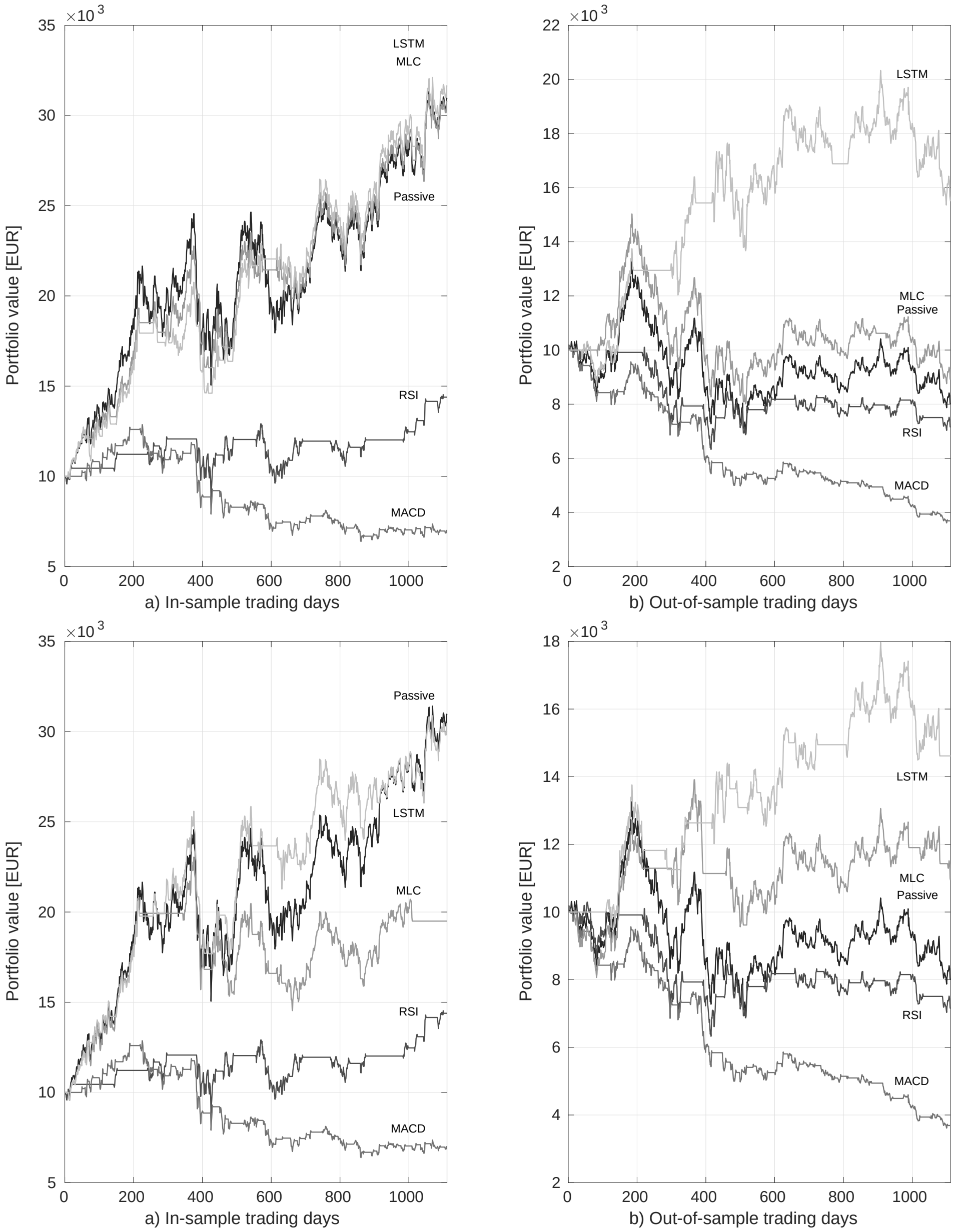

Fig. 6 *Trading results for 4-day (upper portion) and 5-day (lower portion) time horizons. The left figures show the results of the in-sample trading strategy, while the right figures show the out-of-sample trading strategy. The passive strategy performs best for out-of-sample for the 5-day and LSTM performs best for the rest of the samples. Source: Authors.*

Strategy	1-day In/Out	2-day In/Out	3-day In/Out	4-day In/Out	5-day In/Out
Passive	3.093/0.794	3.093/0.794	3.093/0.794	3.093/0.794	3.093/0.794
RSI	1.439/0.714	1.439/0.714	1.439/0.714	1.439/0.714	1.439/0.714
MACD	0.699/0.369	0.699/0.369	0.699/0.369	0.699/0.369	0.699/0.369
MLC	3.215/1.103	4.269/0.902	3.127/1.291	3.076/0.884	1.950/1.087
LSTM	3.916/1.487	4.860/1.902	4.837/1.699	3.161/1.550	3.040/1.461

Tab. IV *Summary of relative capitalization values among trading strategies.*

Strategy	1-day		2-day		3-day		4-day		5-day	
	In	Out	In	Out	In	Out	In	Out	In	Out
Passive	1	1	1	1	1	1	1	1	1	1
RSI	28	24	28	24	28	24	28	24	28	24
MACD	82	88	82	88	82	88	82	88	82	88
MLC	17	1	33	8	37	17	35	6	22	11
LSTM	13	5	37	10	45	9	35	7	39	18

Tab. V *Summary of number of transactions among trading strategies.*

could argue that odd time horizons, excluding the 1-day stock return, achieve a higher number of transactions than even.

Fig. 7 shows the computational time of MLC and LSTM trading strategies. It is clear that the LSTM time complexity considerably exceeds other MLC methods.

5.3 Discussion and conclusion

In this paper we have shown that novel and prominent deep-learning networks can be successfully used for automated stock trading. Stock markets inherently consolidate long-term dependencies, which must be taken into account prior to choosing a trading strategy. To empirically evaluate the success of each of the trading strategies in volatile stock markets, we have developed a mechanical trading system enabling fair evaluation of different trading strategies. Passive, rule-based (RSI and MACD), and surrogate model trading strategies (MLC and LSTM) have been implemented.

We found that the LSTM network outperforms all benchmarking (passive, RSI, MACD and MLC) trading strategies. MLC achieves its highest out-of-sample profit for the 3-day stock returns, while LSTM peaks for the 2-day stock returns (190.2 % of overall return). Profits and losses are consistent for passive and rule-based trading strategies and thus do not depend on the time horizon. MACD is the most frequent trading strategy.

The passive strategy does not depend on any past information and is, therefore, least demanding; the trading performance of other strategies highly depends on the information available. MLC and LSTM, nevertheless, require training and

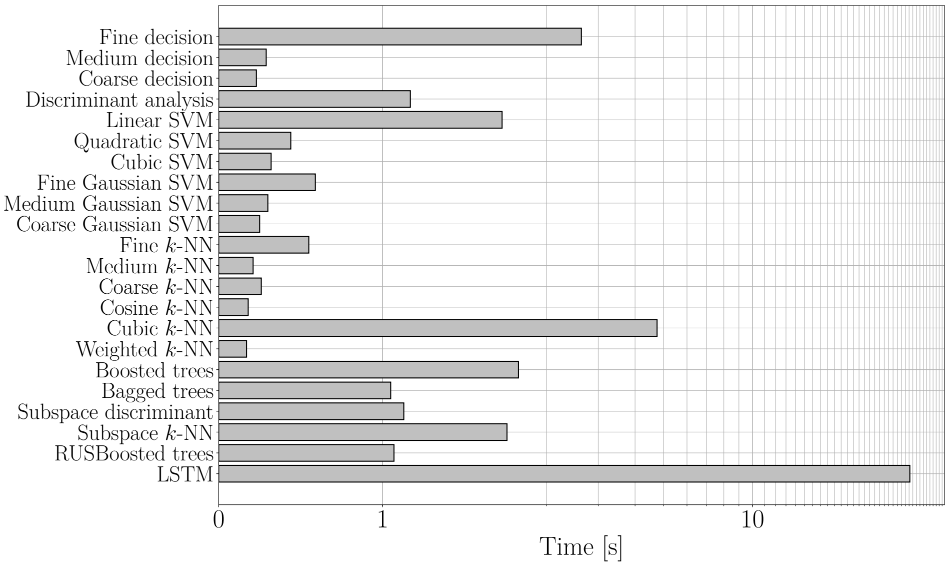

Fig. 7 *A bar chart of computational time requirements, expressed in logarithmic scale. Time [s] is located on the x-axis for each of the surrogate model trading strategies. We can see the very long computational time for the LSTM, which performs the worst according to this criterion. Source: Authors.*

validation datasets and are much more dependent on the information available. Some classification methods, e.g., SVM, might require a greater and more balanced training dataset. For example, the response variable is very imbalanced, since it consists of 17 "Buy," 20 "Sell," and 2,185 "Hold" signals for 1-day stock returns. Surrogate model methods, therefore, achieve 98 % accuracy by constantly outputting the "Hold" signal only.

Stock price movements are random. Therefore, generalization of the surrogate models is highly desired for two reasons: (1) to disregard random movements and (2) to account for systematic movements only. Complicated classifiers may lack of generalization ability, but as it turns out for the LSTM, this can be prevented by careful choice of LSTM units. Moreover, LSTM units can be chosen optimally for each of the time horizon individually. Although this seems sensible, it was not implemented in our case. We have experimentally optimized the architecture for 1-day stock returns and left the architecture universal for the rest of the time horizons. Thus, we have not fully exploited the LSTM's potential.

We found it difficult to assure the stability of the surrogate models, especially the LSTM, which has always reacted in a slightly different way by rerunning the learning process. Typically, this is reconciled by calculating the mean performance on a short out-of-sample period or even cross-validating in some cases. Despite this fact, we employed a different strategy. To accommodate for the nonstability of a surrogate model, a very prolonged out-of-sample dataset has been established to avoid any beneficial random effects. To determine the level of nonstability, multiple learning processes have been rerun. The best among them has been announced as optimal in the end, for which we (1) assumed that it properly recognized the market activity/patterns and reacted accordingly and (2) concluded that the surrogate model did not memorize/remember the data.

To extend the analysis and enlarge the dataset, a longer period of financial data could be downloaded for the BMW stock (beyond the year 2010). Although this seems a reasonable enhancement of the study, one should keep in mind the 2007-2009 global financial crisis that caused significant changes in market activity. By incorporating those years into the analysis, one would risk the infection of pre-crisis market behaviour and the relevance of financial shock for after-crisis market behaviour. We estimated that this does not seem reasonable and thus omitted the pre-crisis period. This omission was later found to be beneficial.

The LSTM neural network and its specific learning process showed a crucial role in the design of the optimal trading strategy and was found to be very useful even though we have, once again, not exploited its full potential. For future, we would like to propose three improvements: (1) longer sequence length to incorporate more comprehensive LSTM configuration; (2) hyperparameter tuning; and (3) an online surrogate model where remodeling would be performed after each trading day. In this way, the newest information would be always incorporated into the surrogate model. Additionally, out-of-sample trading would shrink to one day to further enhance trading results.

This application is a genuine proof-of-concept and a fundamental basis for a more comprehensive portfolio MTS, where many different types of stocks in parallel would be managed by such a system. Based on this encouraging application, we would like to account for all the stocks in the DAX30 blue-chip stock market index. Such an application would significantly increase model complexity and the dataset.

Acknowledgement

The authors acknowledge the financial support from the Slovenian Research Agency (research core funding No. P5-0027).

References

[1] ALTMAN N.S. An introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *The American Statistician.* 1992, 46(3), pp. 175–185, doi: 10.2307/2685209.

[2] ATSALAKIS G.S., VALAVANIS K.P. Surveying stock market forecasting techniques–Part II: Soft computing methods. *Expert Systems with Applications.* 2009, 36(3), pp. 5932–5941, doi: 10.1016/j.eswa.2008.07.006.

[3] BAILLIE R.T. Long memory processes and fractional integration in econometrics. *Journal of Econometrics.* 1996, 73(1), pp. 5–59, doi: 10.1016/0304-4076(95)01732-1.

[4] BALVERS R., WU Y., GILLILAND E. Mean reversion across National Stock Markets and Parametric Contrarian Investment Strategies. *The Journal of Finance.* 2000, 55(2), pp. 745–772, doi: 10.1111/0022-1082.00225.

[5] BATTA P., SINGH M., LI Z., DING Q., TRAJKOVIĆ L. Evaluation of Support Vector Machine Kernels for Detecting Network Anomalies. *2018 IEEE International Symposium on Circuits and Systems (ISCAS).* IEEE, 2018, doi: 10.1109/iscas.2018.8351647.

[6] BOHL M.T., REHER G., WILFLING B. Short selling constraints and stock returns volatility: Empirical evidence from the German stock market. *Economic Modelling.* 2016, 58, pp. 159–166, doi: 10.1016/j.econmod.2016.05.025.

[7] BREIMAN L. Bagging predictors. *Machine Learning*. 1996, 24(2), pp. 123–140, doi: 10.1007/bf00058655.

[8] BREIMAN L. *Classification and Regression Trees*. Routledge, 2017, doi: 10.1201/9781315139470.

[9] BREIMAN L. Random forests. *Machine Learning*. 2001, 45(1), pp. 5–32, doi: 10.1023/a:1010933404324.

[10] BRZESZCZYŃSKI J., BOULIS MAHER I. A stock market trading system based on foreign and domestic information. *Expert Systems with Applications*. 2019, 118, pp. 381–399, doi: 10.1016/j.eswa.2018.08.005.

[11] BUDUMA N., LOCASCIO N. *Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms*. O'Reilly Media, Inc., 2017.

[12] CHEN W.H., SHIH J.Y. A study of Taiwan's issuer credit rating systems using support vector machines. *Expert Systems with Applications*. 2006, 30(3), pp. 427–435, doi: 10.1016/j.eswa.2005.10.003.

[13] CHEN Y., WANG X. A hybrid stock trading system using genetic network programming and mean conditional value-at-risk. *European Journal of Operational Research*. 2015, 240(3), pp. 861–871, doi: 10.1016/j.ejor.2014.07.034.

[14] CHIANG W.C., ENKE D., WU T., WANG R. An adaptive stock index trading decision support system. *Expert Systems with Applications*. 2016, 59, pp. 195–207, doi: 10.1016/j.eswa.2016.04.025.

[15] CHONG T.T.L., NG W.K. Technical analysis and the London stock exchange: Testing the MACD and RSI rules using the FT30. *Applied Economics Letters*. 2008, 15(14), pp. 1111–1114, doi: 10.1080/13504850600993598.

[16] CHUNG Y.P. A Transactions Data Test of Stock Index Futures Market Efficiency and Index Arbitrage Profitability. *The Journal of Finance*. 1991, 46(5), pp. 1791–1809, doi: 10.1111/j.1540-6261.1991.tb04644.x.

[17] CLEMONS E.K., WEBER B.W. Restructuring Institutional Block Trading: An Overview of the OptiMark System. *Journal of Management Information Systems*. 1998, 15(2), pp. 41–60, doi: 10.1080/07421222.1998.11518208.

[18] COHEN P., WEST S.G., AIKEN L.S. *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences*. Psychology Press. 2014, doi: 10.4324/9781410606266.

[19] CORTES C., VAPNIK V. Support-vector networks. *Machine Learning*. 1995, 20(3), pp. 273–297, doi: 10.1007/bf00994018.

[20] OLAH C. colah's blog. 2015-08 [accessed 2018-11-03]. Available from: http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[21] DINH T.A., KWON Y.K. An Empirical Study on Importance of Modeling Parameters and Trading Volume-Based Features in Daily Stock Trading Using Neural Networks. *Informatics*. Multidisciplinary Digital Publishing Institute. 2018, 5(3), doi: 10.3390/informatics5030036.

[22] DYMOVA L., SEVASTJANOV P., KACZMAREK K. A Forex trading expert system based on a new approach to the rule-base evidential reasoning. *Expert Systems with Applications*. 2016, 51, pp. 1–13, doi: 10.1016/j.eswa.2015.12.028.

[23] ELSAS R., EL-SHAER M., THEISSEN E. Beta and returns revisited: Evidence from the German stock market. *Journal of International Financial Markets, Institutions and Money*. 2003, 13(1), pp. 1–18, doi: 10.1016/s1042-4431(02)00023-9.

[24] FAMA E.F. Efficient Capital Markets: A Review of Theory and Empirical Work. *The Journal of Finance*. 1970, 25(2), pp. 383–417, doi: 10.2307/2325486.

[25] FISCHER T., KRAUSS C. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*. 2018, 270(2), pp. 654–669, doi: 10.1016/j.ejor.2017.11.054.

[26] FORSYTH D. *Probability and Statistics for Computer Science*. Berlin, Heidelberg: Springer, 2018, doi: 10.1007/978-3-319-64410-3.

[27] HO T.K. Nearest neighbors in random subspaces. *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Berlin, Heidelberg: Springer, 1998, doi: 10.1007/bfb0033288.

[28] HOCHREITER S., SCHMIDHUBER J. Long Short-Term Memory. *Neural Computation*. 1997, 9(8), pp. 1735–1780, doi: 10.1162/neco.1997.9.8.1735.

[29] HOPFIELD J.J. Neural networks and physical systems with emergent collective computational abilities. In: *Proceedings of the National Academy of Sciences*. 1982, 79(8), pp. 2554–2558, doi: 10.1073/pnas.79.8.2554.

[30] ISKRICH D., GRIGORIEV D. Generating long-term trading system rules using a genetic algorithm based on analyzing historical data. *2017 20th Conference of Open Innovations Association (FRUCT)*. IEEE, 2017, doi: 10.23919/fruct.2017.8071297.

[31] JOHANN T., SCHARNOWSKI S., THEISSEN E., WESTHEIDE C., ZIMMERMANN L. *Liquidity in the German stock market* [online]. Mannheim: University of Mannheim [viewed 2018-11-15]. Working paper, 2018. Available from: https://www.bwl.uni-mannheim.de/media/Lehrstuehle/bwl/Theissen/Forschung/Paper_XetraMicrostructure.pdf

[32] KARIM M., RAHMAN R.M. Decision Tree and Naïve Bayes Algorithm for Classification and Generation of Actionable Knowledge for Direct Marketing. *Journal of Software Engineering and Applications*. 2013, 6(4), pp. 196–206, doi: 10.4236/jsea.2013.64025.

[33] LU X., SUN T., TANG K. Evolutionary optimization with hierarchical surrogates. *Swarm and Evolutionary Computation*. 2019, doi: 10.1016/j.swevo.2019.03.005.

[34] KINGMA D.P., BA J. Adam: A method for stochastic optimization. *arXiv preprint*. arXiv:1412.6980, 2014.

[35] KOTSIANTIS S.B., ZAHARAKIS I., PINTELAS P. Machine learning: A review of classification and combining techniques. *Artificial Intelligence Review*. 2006, 26(3), pp. 159–190, doi: 10.1007/s10462-007-9052-3.

[36] LÄNGKVIST M., KARLSSON L., LOUTFI A. A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognition Letters*. 2014, 42, pp. 11–24, doi: 10.1016/j.patrec.2014.01.008.

[37] LECUN Y., BENGIO Y., HINTON G. Deep learning. *Nature*. 2015, 521(7553), pp. 436–444, doi: 10.1038/nature14539.

[38] LEE C.C., LEE J.D., LEE C.C. Stock prices and the efficient market hypothesis: Evidence from a panel stationary test with structural breaks. *Japan and the World Economy*. 2010, 22(1), pp. 49–58, doi: 10.1016/j.japwor.2009.04.002.

[39] LENSKIY. Market Data Functions. 2018 [accessed 2018-09-29]. Available from: https://github.com/Lenskiy/market-data-functions.

[40] LO A.W., MACKINLAY A.C. Stock Market Prices Do Not Follow Random Walks: Evidence from a Simple Specification Test. *The Review of Financial Studies*. 1988, 1(1), pp. 41–66, doi: 10.1093/rfs/1.1.41.

[41] MATHWORKS MATLAB. Long Short-Term Memory Networks. 2018 [accessed 2018-11-03]. Available from: https://www.mathworks.com/help/deeplearning/ug/long-short-term-memory-networks.html

[42] MIKA S., RATSCH G., WESTON J., SCHOLKOPF B., MULLERS K.R. Fisher discriminant analysis with kernels. In: *Proceedings of the 1999 IEEE Signal Processing Society Workshop*. IEEE, 1999, doi: 10.1109/nnsp.1999.788121.

[43] NOFER M., HINZ O. Using Twitter to Predict the Stock Market. *Business & Information Systems Engineering*. 2015, 57(4), pp. 229–242, doi: 10.1007/s12599-015-0390-4.

[44] NUMPACHAROEN K. Classifying Trading Signals using Machine Learning and Deep Learning. 2018 [accessed 2018-10-24]. Available from: https://www.mathworks.com/videos/classifying-trading-signals-using-machine-learning-and-deep-learning-1524490851722.html.

[45] OPITZ D., MACLIN R. Popular Ensemble Methods: An Empirical Study. *Journal of Artificial Intelligence Research*. 1999, 11, pp. 169–198, doi: 10.1613/jair.614.

[46] PARR R., RUSSELL S.J. Reinforcement learning with hierarchies of machines. *Advances in Neural Information Processing Systems*. 1998.

[47] PARDO R. *Design, Testing, and Optimization of Trading Systems*. John Wiley & Sons. 1992, 2.

[48] PASCANU R., MIKOLOV T., BENGIO Y. On the difficulty of training recurrent neural networks. In: *International Conference on Machine Learning*, 2013.

[49] PENG L., LIU S., LIU R., WANG L. Effective Long short-term Memory with Differential Evolution Algorithm for Electricity Price Prediction. *Energy*. 2018, 162, pp. 1301–1314, doi: 10.1016/j.energy.2018.05.052.

[50] QUINLAN J.R. Induction of decision trees. *Machine Learning*. 1986, 1(1), pp. 81–106, doi: 10.1007/bf00116251.

[51] REZAEE M.J., JOZMALEKI M., VALIPOUR M. Integrating dynamic fuzzy C-means, data envelopment analysis and artificial neural network to online prediction performance of companies in stock exchange. *Physica A: Statistical Mechanics and Its Applications*. 2018, 489, pp. 78–93, doi: 10.1016/j.physa.2017.07.017.

[52] ROBERTS H.V. Statistical versus clinical prediction of the stock market. 1967.

[53] RUMELHART D.E., HINTON G.E., WILLIAMS R.J. Learning representations by back-propagating errors. *Nature*. 1986, 5(3), pp. 533–536, doi: 10.1038/323533a0

[54] SADIQUE S., SILVAPULLE P. Long-term memory in stock market returns: International evidence. *International Journal of Finance & Economics*. 2001, 6(1), pp. 59–67, doi: 10.1002/ijfe.143.

[55] SAMUEL A.L. Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development*. 1959, 3(3), pp. 210–229, doi: 10.1147/rd.33.0210.

[56] SEIFFERT C., KHOSHGOFTAAR T.M., VAN HULSE J., NAPOLITANO A. RUSBoost: A Hybrid Approach to Alleviating Class Imbalance. *IEEE Transactions on Systems, Man, and Cybernetics–Part A: Systems and Humans*. 2010, 40(1), pp. 185–197, doi: 10.1109/tsmca.2009.2029559.

[57] SHIN K.S., LEE T.S., KIM H. An application of support vector machines in bankruptcy prediction model. *Expert Systems with Applications*. 2005, 28(1), pp. 127–135, doi: 10.1016/j.eswa.2004.08.009.

[58] SHIN H.W., SOHN S.Y. Segmentation of stock trading customers according to potential value. *Expert Systems with Applications*. 2004, 27(1), pp. 27–33, doi: 10.1016/j.eswa.2003.12.002.

[59] ŠONJE V., ALAJBEG D., BUBAŠ Z. Efficient market hypothesis: is the Croatian stock market as (in)efficient as the US market. *Financial Theory and Practice*. 2011, 35(3), pp. 301–326, doi: 10.3326/fintp.35.3.3.

[60] SULMASY D.P., SNYDER L. Substituted Interests and Best Judgments: An Integrated Model of Surrogate Decision Making. *JAMA*. 2010, 304(17), doi: 10.1001/jama.2010.1595.

[61] YAHOO FINANCE. 2018 [accessed 2018-10-10]. Available from: https://finance.yahoo.com/.

[62] WANG J.L., CHAN S.H. Trading rule discovery in the US stock market: An empirical study. *Expert Systems with Applications*. 2009, 36(3), pp. 5450–5455, doi: 10.1016/j.eswa.2008.06.119.

[63] WANG J., GEORGE V., BALCH T., HYBINETTE M. Stockyard: A discrete event-based stock market exchange simulator. In: *Simulation Conference (WSC) 2017 Winter*. IEEE, 2017, doi: 10.1109/wsc.2017.8247866.

[64] WANG W., MISHRA K.K. A novel stock trading prediction and recommendation system. *Multimedia Tools and Applications*. 2018, 77(4), pp. 4203–4215, doi: 10.1007/s11042-017-4587-z.

[65] WEBER B.W. Screen-based trading in futures markets: recent developments and research propositions. In: *Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences. 1999. HICSS-32*. Hawaii, USA. IEEE, 1999, doi: 10.1109/hicss.1999.772767.

[66] WEISSMAN R.L. *Mechanical Trading Systems: Pairing Trader Psychology with Technical Analysis.* John Wiley & Sons. 2005, 220.

[67] ZACHARAKI E.I., WANG S., CHAWLA S., YOO D.S., WOLF R., MELHEM E.R., DAVATZIKOS C. Classification of brain tumor type and grade using MRI texture and shape in a machine learning scheme. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*. 2009, 62(6), pp. 1609–1618, doi: 10.1002/mrm.22147.