University of Warsaw
Faculty of Economic Sciences

Working Papers
No. 27/2020 (333)

PREDICTING PRICES OF S&P500 INDEX USING CLASSICAL METHODS AND RECURRENT NEURAL NETWORKS

Mateusz Kijewski
Robert Ślepaczuk

Warsaw 2020

WORKING PAPERS 27/2020 (333)

Predicting prices of S&P500 index using classical methods and recurrent neural networks

Mateusz Kijewski[a], Robert Ślepaczuk[b]*

[a] *Faculty of Economic Sciences, Quantitative Finance Research Group, University of Warsaw*
[b] *Faculty of Economic Sciences, Quantitative Finance Research Group, Department of Quantitative Finance, University of Warsaw*
* *Corresponding author: rslepaczuk@wne.uw.edu.pl*

Abstract: This study implements algorithmic investment strategies with buy/sell signals based on classical methods and recurrent neural network model (LSTM). The research compares the performance of investment algorithms on time series of S&P500 index covering 20 years of data from 2000 to 2020. This paper presents an approach for dynamic optimization of parameters during backtesting process by using rolling training-testing window. Every method was tested in terms of robustness to changes in parameters and evaluated by appropriate performance statistics e.g. Information Ratio, Maximum Drawdown, etc. Combination of signals from different methods was stable and outperformed benchmark of Buy & Hold strategy doubling its returns on the same level of risk. Detailed sensitivity analysis revealed that classical methods which used rolling training-testing window were significantly more robust to changes in parameters than LSTM model in which hyperparameters were selected heuristically.

Keywords: machine learning, recurrent neural networks, long short-term memory model, time series analysis, algorithmic investment strategies, systematic transactional systems, technical analysis, ARIMA model

JEL codes: C4, C14, C45, C53, C58, G13

Note: The views presented in this text are those of the authors and do not necessarily represent those of Labyrinth HF project.
This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors

1. Introduction

Using time series analysis to forecast prices of stock market is an extensively researched topic. Technical analysis is a widely applied tool for predicting future price movements on the market. However, investors are highly focused on risk management. Thus, it is crucial for optimizing trading decisions to calculate returns weighted by risk, as it enables to gain maximum revenues from strategies. Recent fast growth of data mining technologies made it much more approachable and quicker to analyze large datasets and provide explanations for received results.

The analysis prepared in this paper enabled to verify main hypothesis of this research that the stock market is inefficient, because it is possible to predict movement of prices based on historical data. Another tested hypothesis states that the combination of signals generated by different strategies will perform better than each of them separately. Classical methods are more robust to changes in parameters than recurrent neural network model. Long short-term memory model predict movements of prices better than ARIMA model.

This research focuses on creating different signals generated by strategies that predicts movement of stock prices. Study assumes that there is an asymmetric information which makes it possible to analyze the volatility in the stock, as well as predict prices within available historical data. It enables to outperform the Buy & Hold strategy. Thus, study tests multiple methods including: ARIMA model, macroeconomic factor, volatility breakout, momentum, and contrarian strategies, moving average crossover and recurrent neural network model LSTM. Each of them generated signals for 20 years period, trading on S&P500 index.

Study is divided into five chapters. First chapter describes literature focusing on automated transactional systems using technical analysis methods and machine learning. Second chapter briefly describes data used in this study including calculating Kolmogorov-Smirnov test for the data. Third chapter focuses on methodology of used methods explaining how each strategy generates signals. Fourth chapter presents empirical results of every method and compare performance statistics. Fifth chapter tests robustness of results by changing parameters set in methodology section. The last one concludes and verify hypotheses. The last part of the research provides recommendation for future research.

2. Literature Review

Application of data series analysis to forecast stock market prices is a topic which is widely discussed in a scientific literature. The interest in this field arises from high potential profits associated with correct predictions of stock prices. The discussion has been arising for decades now, as technological progress resulted in increasing computers' computational power, making the algorithms faster and more accessible. Some researchers prove that algorithmic trading bring many advantages to the market, like lowering adverse selection and decreasing the amount of price discovery that is correlated with trading, as well as increasing the informativeness of quotes (Hendershot et al. 2011). Other research provided evidence that "algorithmic trading is associated with improved liquidity, improved efficiency, and elevated volatility" (Boehmer et al. 2012). Additionally, there were attempts of combining low liquidity and not correlated investment strategies on various asset classes which resulted in significantly increased risk adjusted returns of such combined systems (Ślepaczuk et al., 2018). Such research were made not only on classical assets like: equities, bonds, currencies or commodities (Ślepaczuk et al., 2018) but on lately recognized new asset classes (volatility, cryptocurrency, etc.) as well (Jabłecki et al., 2015, Sakowski et al., 2016, Zenkova and Ślepaczuk, 2018, Kość et al., 2019,)

First articles were based on simple methodologies, which used moving averages, trend trading and ARIMA class models. James (1968) conducted a research on a relationship between monthly future stock prices for a chosen set of stocks from New York Stock Exchange from 1926 to 1960. Two methods were used – moving average and exponential smoothing. For most chosen models, the results were worse than in the „buy and hold" strategy. The scientist indicates that combining various averages depending on market volatility would improve the quality of predictions. Brock et al. (1992) verified the efficiency of autoregressive model, GARCH-M and random walk model using the Dow Jones Industrial Average data for the years 1897-1986. The research showed that these tools predict "buy" signals correctly, but they do not forecast the "sell" signal successfully. Authors believe that this drawback exists due to higher volatility in periods with price loss than with price growth.

Gerlow et al. (1990) compared VAR, ARIMA and CHL methods in their article by testing their standard deviations and return rates of the futures contracts for soybeans for the monthly data from January 1974 to December 1983 (in-sample period). The out-of-sample

period was January 1984 to June 1988. The results of this research suggest ARIMA and VAR ineffectiveness. Presented strategies provided worse outcomes than the trend trading model.

Another example of predicting prices was observed in the research conducted by Devi et al. (2013). The dataset used for this analysis was collected from National Stock Exchange of India for the period of January 2007 to December 2011 and it regarded market capital value. Using Box-Jenkins methodology enabled to identify the ARIMA model which proposed the best prediction. Similar research was conducted on daily Cheung Kong Holding (CK HDG) and HSBC Holding (HSBC HDG) prices in the period from September 1995 to July 1999 (Chiu and Xu, 2003). The results were obtained using ARMA-GARCH model, which provided better results than the mixture of AR-GARCH model.

Another trading approach that can be seen on the market is Open Range Breakout (ORB). Holmberg et al. (2013) provided a research which was conducted on the US crude oil futures prices from March 30 of 1983 to January 26 of 2011. The method used in this research depends on the volatility of prices measured by distance from open price to high and low intraday price. Then deciding on the move that prices make. Short positions are generated when the market price crosses from above threshold created as a specific percentage of the opening price of the asset, and otherwise – long positions are taken when the price crosses from below threshold created as a specific percentage of the opening price of the asset. Researchers showed that the ORB method 'results in significant positive average returns' and, what's more, real-life market can provide even better results, as losses can be limited by so-called stop losses.

Market analysis is crucial to be able to react to crisis on time. The loss associated with the market collapse impact the whole economy – not only current revenue, but also investments, real wages, and unemployment rates (Zoega, 2010). Many measures of early indicators of crisis are discussed e.g. National Reserves, GDP growth, National Debt, Unemployment rate (Eichengreen et al. 1996; Hawkins and Klau 2000; Abiad 2003; Frankel and Saravelos 2011). The research prepared by McQueen and Roley (1990) provided results which suggest that macroeconomic factors have different effect on stock prices. They tested impact on cash flows based on unemployment rates. During rapid development of economy negative information have less impact on market movements, while in times of uncertainty this impact is significantly higher.

Besides econometrical models like ARIMA, VAR, GARCH nowadays, more attention is put on neural network applications due to their high successes in pattern recognition e.g.

image recognition, speech recognition. Those patterns also can be found in time series data like stock prices. Roondiwala et al. (2017) tried to predict stock returns of index NIFTY 50. They trained multivariate LSTM model using daily open, close, high, low prices as features. Model used two hidden layers with 128 and 64 units, each. Dense layer activation function was 'ReLU' and optimizer 'RMSprop'. Sequence of days for individual input was 15 days. After testing different combinations of epochs and features the most accurate model in terms of Root Mean Square Error used all four features and used 500 epochs for model training.

Other work implementing LSTM model in scientific approach was made by Chen et al. (2015). They collected data for China stocks and divided percentage returns of prices into seven groups: [,-1.5], [-1.5,-0.5], [-0.5, 0.4], [0.4, 1.4], [1.4, 2.5], [2.5, 4.3], [4.3,]. The main aim of the research was to successfully predict a proper group of the next day return. Besides data on returns, they also used 10 different features: open, low, high, close prices and volume for given stock and the same five features for Shanghai Securities Composite Index. Model specification used in this research: 30 days sequence length, 'RMSprop' optimizer, learning rate 0.001. The best results measured by accuracy of predicted return group were given by model using all ten features achieving 27.2% accuracy, which is twice as much as randomly picked groups.

Zhang et al. (2019) presented AT-LSTM model which is combination of LSTM and Attention based model. Attention based model adds new layer to LSTM for selecting weights of importance for each of the output generated by simple LSTM model. Authors provided results from their empirical studies using three index datasets: Russell 2000, DJIA and NASDAQ. All data contains 6885 days starting from 02.01.1991 to 30.04.2018. 4500 first days were selected as training sample. Parameters used in final model: 20 days sequence length, 8 units in hidden layer, 5000 epochs, MAPE loss function. AT-LSTM model achieved best results in terms of MAPE error on each of three datasets, beating simple LSTM model and ARIMA model.

Slightly different approach using LSTM model was presented in Sang and Di Pierro (2018) instead of using prices or returns for predicting stock price movements, authors decided to use well known technical analysis trading strategies signals as features. Selected methods: Simple Moving Average, Relative Strength Index and Moving Average Convergence Divergence. Dataset used in empirical study contained five stocks with highest capitalization in each from nine sectors of S&P500. Parameters used in final model: one hidden layer, learning rate 0.001, 15 days sequence length. LSTM outperformed oscillators on 6 of 9 sectors.

One of the last approach which tested various machine learning techniques for time series forecasting problem was paper of Chlebus et al. (2020) who applied the following methods: SVR, KNN, XGBoost, LightGBM, LSTM, ARIMA, ARIMAX with features coming from such classes like: technical analysis, fundamental analysis, Google Trends entries, markets related to Nvidia. The best performance was obtained by SVR based on stationary attributes.

Presented literature do not fully cover the possibilities offered by basic models in forecasting stock market prices. Previous research was focused on choosing a proper method and testing its profitability, without changing the parameters depending on the previous results. More attention should be concentrated on the potential of merging few methodologies or adjusting the models' parameters to obtain better possible outcome.

3. Data

3.1 Data description

This paper use S&P500 index close prices for the period from 01.01.1994 to 02.05.2020, where trading period starts from 01.01.2000. Data was downloaded from Yahoo Finance API. Second time series used in this study was initial jobless claims in weekly intervals for the same period downloaded from Federal Reserve Bank Economic Dataset (fred.stlouisfed.org).

Standard and Poor's is an American credit rating agency, which holds one of the most popular index of United States companies – S&P500. This index measures performance of 500 companies with highest capitalization which are listed on U.S stock exchanges. S&P500 values for trading period used in this research are presented in Figure 1.

Figure 1. S&P500 index

Note: S&P500 index values in US dollars for the period 01.01.2000-02.05.2020.

2.2 Data analysis

This section will provide preliminary data analysis. Simple descriptive statistics for S&P500 percentage daily returns are presented in Table 1.

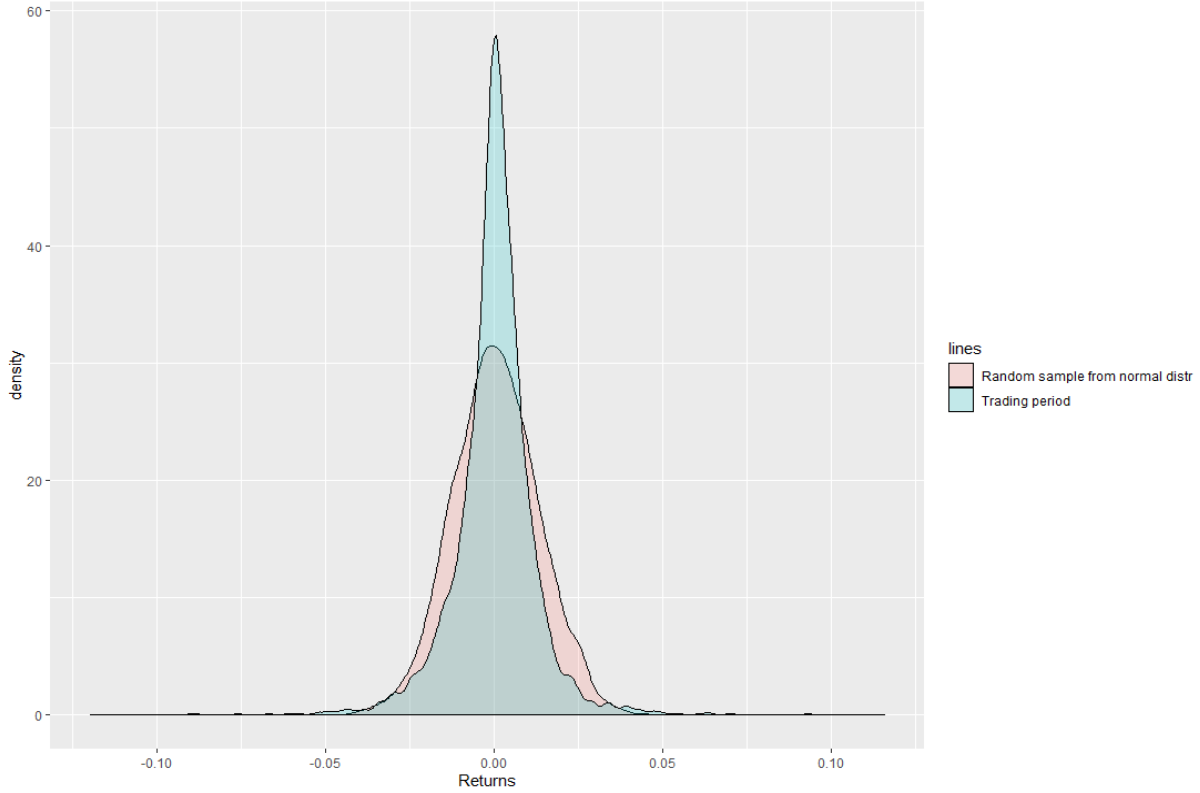
Table 1. S&P500 daily simple returns statistics

Name	Mean	SD	1st quartile	3rd quartile	Kurtosis	Skewness	Kolmogorov-Smirnov test p-value
S&P500	0.02%	1.25%	-0.47%	0.57%	10.93	-0.12	0

Note: Descriptive statistics calculated for S&P500 on simple returns for the period 01.01.2000-02.05-2020.

Daily simple returns of S&P500 statistics suggest leptokurtosis as kurtosis statistic is significantly higher than in normal distribution in which kurtosis is equal to 3. To confirm this hypotheses, Kolmogorov-Smirnov test for normality was computed. In this test null hypothesis is that given empirical data was drawn from normal distribution with assumed confidence interval. Alternative hypotheses deny the fact that empirical data was drawn from normal distribution. For this sample D statistics was equal to 0.99 and p-value was equal to 0. Test rejected null hypotheses. Thus, S&P500 daily returns are not drawn from normal distribution.

Figure 2. Normally distributed random sample vs trading period of S&P500



Note: Comparison between two same size samples, one sample is randomly generated observations from normal distribution and second sample are S&P500 returns from 01.01.2000-02.05-2020 period.

In Figure 2 two distributions are presented – blue is the distribution of simple returns in trading period, red is randomly generated array of same length, but from normal distribution. There is clear difference in kurtosis between these two distributions. These observations confirm stylized fact that distribution of stock returns is leptokurtic. Additionally, we can see that the distribution of returns has fatter tails. These observations confirm stylized fact that distribution of stock returns is leptokurtic.

4. Methodology

4.1 Description of used classical methods

Each presented strategy will follow the same general rules during investment process. There are three possible signals – buy, sell and stop. Buy signal means that strategy will use all available capital for buying asset. Sell signal means that strategy will use all available capital for deposit in short transaction (short transaction requires 100% deposit). Stop signal means that strategy will not hold any position in the following day. Research assume that each time strategy can buy or sell every decimal part of S&P500. Every trade will be charged with 0.05% transactional fees of investing capital. Thus, changing position from short to long costs 0.1% of capital invested.

Each presented strategy, except ARIMA and LSTM models, will optimize parameters in the same way – using rolling training-testing window. This is iterative process composed of following steps:

1. Select first 504 days as training window, which is the time for every strategy to obtain all information (estimation of the model, calculating moving averages etc.) on which it generates trades for every combination of tested parameters.
2. Select the next 504 days as testing window, in which strategy will use every calculated statistic and based on that generate signals. Then algorithm calculates performance statistics and chooses parameters which gave the best information ratio (IR) value.
3. Select the next 63 days as trading period and use selected parameters from point 2 and generate signals for each day.
4. Shift each window by 63 days.
5. Repeat 1-4 until last trading period will reach the end of time series.

4.1.1 ARIMA

Autoregressive Integrated Moving Average model gain popularity after wide description in first edition of Box and Jenkins „Time Series Analysis" in 1970, in which they prepared procedure for specification, estimation, diagnostics and forecasting for univariate time series.

ARIMA (p, d, q) model can be divide into three parts:

- autoregressive - add dependent variables as p lagged observations
- integrated - describes d number of differentiations needed to make time series stationary
- moving average - add independent variables as q lagged error terms

The model can be described with following formula:

$$y_t = a_1 y_{t-1} + \cdots + a_p y_{t-p} + e_t + b_1 e_{t-1} + \cdots + b_q e_{t-q} \quad (1)$$

where:

$p - is\ the\ number\ of\ autoregressive\ lags,$

$q - is\ the\ number\ of\ error\ lags,$

$y_t - \ is\ the\ value\ of\ time\ series\ at\ time\ t,$

$e_t - \ is\ the\ error\ at\ time\ t.$

There are several methods for identification of p, d, q values e.g. Box-Jenkins procedure, general to specific approach, Akaike information criterion (AIC). Due to automatization purposes Akaike information criterion was selected. AIC values are calculated using the following formula:

$$AIC = -2\log(\mathrm{L}) + 2(p+q) \quad (2)$$

where:

$L - is\ the\ value\ of\ likelihood\ function\ for\ estimated\ model$

$p - is\ the\ number\ of\ autoregressive\ lags$

$q - is\ the\ number\ of\ error\ lags$

After calculating AIC values for many combinations of parameters, model with the lowest AIC value is selected as most accurate for forecasting purposes. Second step in procedure is to estimate model with selected parameters using either OLS estimation or maximum likelihood estimation. In this research maximum likelihood estimation was chosen due to time efficiency. The last step is forecasting and in this study one step ahead forecast is calculated, simply by substituting independent variables with observed values and multiply

them by estimated parameters. The detailed procedure for forecasting future value of S&P500 index is presented below.

1. Set training sample as first 504 days in dataset

2. Identify optimal values of p, d, q from all combinations of p: 0-5, d: 0-3, q: 0-5

3. Estimate model with chosen p, d, q using maximum likelihood method

4. Forecast one-step ahead

5. If forecasted value is higher than last observation plus fees - generate buy signal, if forecasted value is lower than previous observation plus fees – generate sell signal, otherwise generate no signal.

6. Repeat 1-5 points moving training sample by one day ahead, until end of dataset.

7. Calculate performance statistics for the whole period

4.1.2 Moving average crossover

The concept of moving average was exploited in many ways in technical analysis. One of the most popular concepts is moving average crossover (Park and Irwin, 2007). Many researchers found this method profitable and beating the buy and hold strategy on out of sample period (Irwin and Uhrig, 1984; Gunasekarage and Power, 2001; Olson, 2004).

Strategy assumes that instrument on which signals are generated follow long-term trends (upward or downward) which change its direction due to strong short-term trend. Thus, there is need to calculate two moving averages – one short and one long. Buy signal is generated when short moving average crosses long moving average from below - there is expectation for prices to continue short upward trend. On the other hand, if short moving average crosses long moving average from above, sell signal is generated – there is expectation for prices to continue short downward trend. Assuming that there were such crossovers in historical data, this strategy can be easily restated to other signal generating conditions. Each day if short moving average is above long moving average – strategy is generating buy signals, otherwise it is generating sell signals.

There are two main parameters used in the strategy described above, the number of days used for calculating short and long moving average. In literature commonly used parameters for short moving average are in the range from 5 to 20 (Olson 2004) and for long moving average are in the range from 50 to 250 (Olson 2004; Gunasekarage and Power 2001).

Considering previous studies, this research used 504 days rolling training-testing window described in the beginning of this chapter and optimize this method within the following parameters:

- days for short moving average: 5, 10, 15, 20, 25, 30
- days for long moving average: 50, 100, 150, 175, 200, 225, 250, 300

Thus, in each iteration algorithm searches through 48 combinations of parameters, chooses the one with highest information ratio on testing period and uses selected combination on next 63 days. Described approach enables to generate signals created for each day.

4.1.3 Momentum and contrarian

In the area of trend-based strategies literature often tests two approaches – momentum and contrarian. First one assumes that asset prices are following long-term trends, and that in short-term prices tend to randomly fluctuate around zero returns. On these relatively simple assumptions numerous trading strategies were created e.g. winners and losers portfolios (Miffre and Rallis, 2007; Schiereck, et al. 1999). Second approach assumes that short-term price trends are changing frequently between upward and downward. Which means that if the last n days prices were raising, this approach would suggest selling this asset. Thus, first method will search for long-term trends and second method will try to obtain gains by dynamically changing its position.

This study focuses on two specific implementations of these approaches. First signal generating system is based on momentum approach. Algorithm calculates historical returns for each of following lags (7, 21, 63, 126) on training window, then on testing window chooses weights for each historical returns, with possible weights 0 or 1. Second signal generating system is based on contrarian approach. Algorithm calculates historical returns for each of following lags (1, 2, 5, 10), then on testing window chooses weights for each historical returns, with possible weights of 0 or -1.

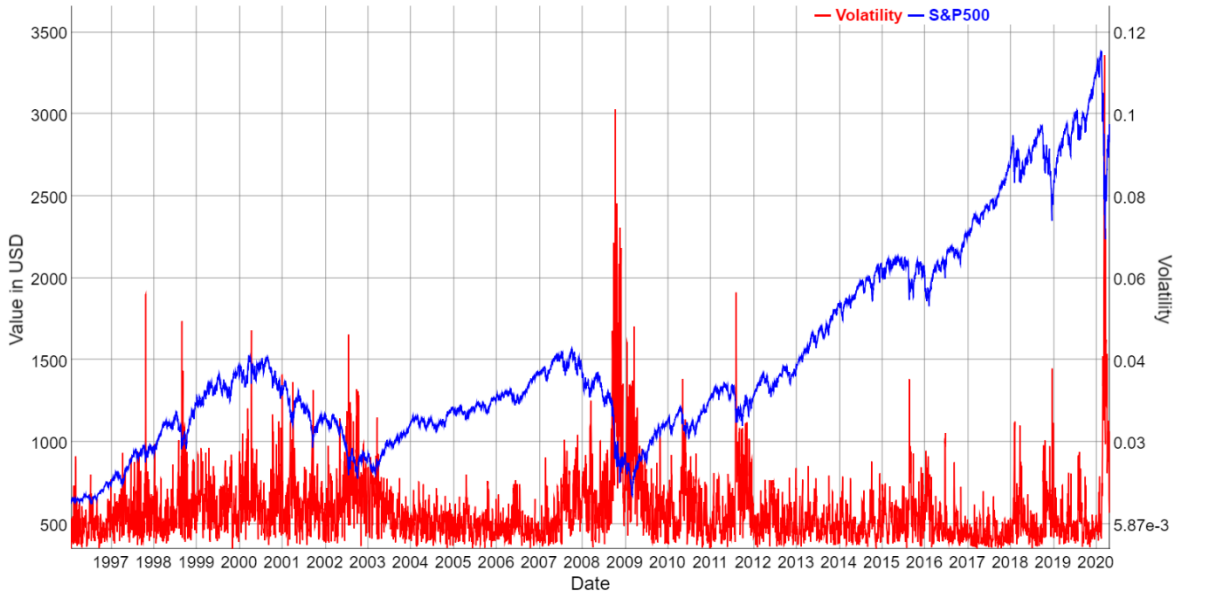
4.1.4 Volatility breakout

Breakout strategies are frequently used in technical analysis. In general such strategy depends on one or two thresholds, which are set around specific indicator. It can be price, returns or volatility. Signals are generated when our indicator crosses threshold. There are multiple methods of calculating thresholds e.g. adding standard deviation of n days to price and call it as an upper threshold and subtracting standard deviation of n days from price and call it lower

threshold. If current price crosses upper threshold, strategy generates buy signal, in opposite if current price crosses lower threshold, strategy generates sell signal.

In this research strategy using channels is based on stylized facts about asset returns, specifically volatility clustering. Assuming that fact volatility is not normally distributed in sub samples, but there can distinguished short periods of high volatility and long periods of low volatility. What is more periods of high volatility are starting in most cases with strong downward trend. In Figure 3 prices of S&P500 index are presented and its volatility measured by 3-day standard deviation of returns. As it was discussed earlier, without any problems periods of high volatility can be spotted. Taking cutoff level of volatility on 0.03 there are only 14 periods reaching this value in the last 25 years. In each case these periods have started with strong downward trend. These stylized facts are background for implemented strategy. Every time when current volatility level exceeds its historical q_1 (q_1>0.5) quantile – market is coming to high volatility period, strategy generates sell signal. However if current volatility level is lower than q_2 (q_2<0.5) quantile – market is in low volatility period, strategy generates buy signal, otherwise there is stop signal. Expected main advantage of this method is being out of the market for periods with mixed volatility and take advantage of market when tends to behave consistently.

Figure 3. Volatility and prices of S&P500



Note: Volatility of S&P500 index prices measured by 3-day standard deviation of returns. High volatility periods are often associated with strong downward trends The data covers the period from 1996 to 2020.

Using rolling training-testing window following parameters are optimized:

- days for calculating distribution of volatility: 63, 126, 252, 504, 1008

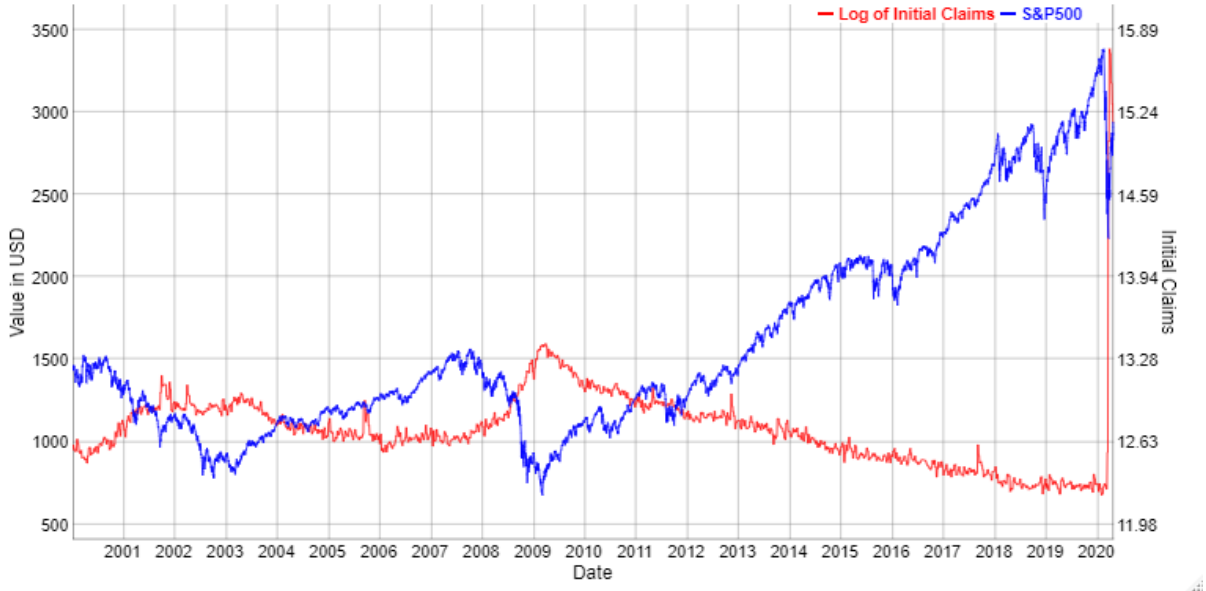
- quantile level q_1: 0.6, 0.7, 0.8, 0.9
- quantile level q_2: 0.1, 0.2, 0.3, 0.4
- number of tested combinations in each iteration: 80

4.1.5 Macro factor

S&P500 Index measures performance of 500 companies with the highest capitalization from United States stock exchanges. Thus, value of this index should partly reflect condition of US economy. For that reason, finding leading (early warning) indicator which values could inform about incoming crisis, would highly increase number of good market decisions. In literature most common indicators are: National Reserves, GDP growth, National Debt, Unemployment rate (Eichengreen, et al., 1996; Hawkins and Klau, 2000; Abiad, 2003; Frankel and Saravelos, 2011). Main issue with day-to-day trading and macroeconomics factors is data release the frequency of these factors. Most of them are with at least one-month frequency.

Thus, this research focused on unemployment rate indicator, which can be substituted with Initial Jobless Claims. There is weekly report published every Thursday by U.S. Unemployment and Training Administration about the number of people filing unemployment claims. For daily data purposes this research assumes the same value of initial claims for one week until new report is released. In Figure 3.2. there is presented the logarithm of initial claims and price of S&P500 index.

Figure 4. Logarithm of initial jobless claims and S&P500 index



Note: Initial jobless claims weekly data for the period 01.01.2000-02.05.2020. compared with S&P500 index values.

As described in the literature and as it is presented on Figure 4, decline in S&P500 prices is highly correlated with unemployment claims. These observations led to implementation of

signal investment strategy similar to Volatility Breakout. Every time initial claims exceeds its historical q_1 (q_1>0.5) quantile – economy growth is expected to decrease and strategy generates sell signal. However, if the current number of initial claims is lower than q_2 (q_2<0.5) quantile – economy growth is expected to increase, strategy generates buy signal, otherwise there is stop signal. Following parameters are optimized during rolling training-testing windows:

- days for calculating distribution of initial jobless claims: 63, 126, 252, 504, 1008
- quantile level q_1: 0.6, 0.7, 0.8, 0.9
- quantile level q_2: 0.1, 0.2, 0.3, 0.4
- number of tested combinations in each iteration: 80

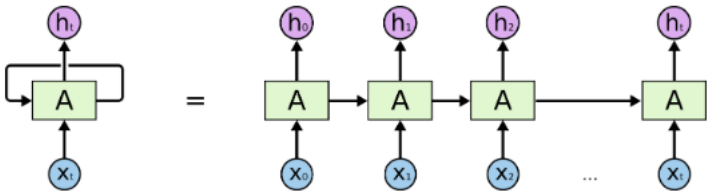
4.2 Recurrent neural network

The most known and the oldest type of neural network was feedforward. Which basically means that nodes do not form any cycles and information is not repeated for any unit. As the name describes information goes forward from input layer through hidden layers to output layer. The simple extensions to these methods which allows for information to move in cycle is recurrent neural network, which is extremely useful in series data types, where past observations have direct impact on future observations. For that reason, recurrent neural networks have many applications in speech recognition, image recognition or natural language processing, where order of words and context is crucial to understand the whole sentence. The main issue of recurrent neural network is the optimization algorithm called gradient descent. During this process weights of each node are changed to minimize error. With cycle architecture recurrent neural networks faces the problem of gradient vanishing (Kolen and Kremer, 2001), which means that changes in weights of past observations are decreasing exponentially in time. Thus, simple recurrent neural network is unable to learn long-term dependencies, such as context of certain text paragraph which is crucial for further speech recognition or recognizing and remembering patterns of time series analysis like seasonality.

4.2.1 LSTM model description

Long short-term memory (LSTM) addresses gradient vanishing problem mentioned in previous section, by adjusting RNN architecture. Vanilla recurrent neural network, with unfolded cycle which is easier to interpret, is presented in Figure 5.

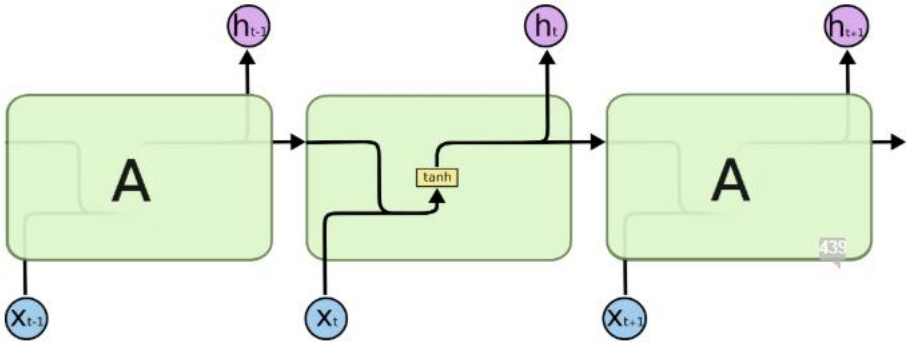
Figure 5. Unfolded recurrent neural network unit



Note: Unfolded recurrent neural network unit, where x_t are inputs and h_t are outputs. A is neural layer, which transforms input data. Source: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ on 01.05.2020.

RNN derived from vanilla neural networks by using hidden state, which is a matrix of parameters passed between sequential inputs, in Figure 6 it is presented as horizontal arrow. Parameters in this matrix are optimized to minimize loss function. Due to the fact that this matrix is passed between inputs it allows for transferring information from previous inputs about importance of the input. Thus, RNN will return different results for the same input, but different past, due to differences in weights from hidden state. Figures 6 and 7 show significant adjustments between architecture of RNN and LSTM.

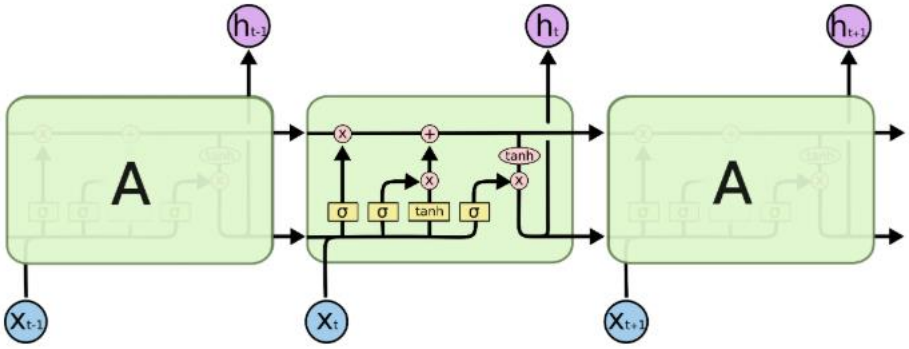
Figure 6. Architecture of recurrent neural network



Note: Example of simple recurrent neural network architecture with single layer and tanh activation function, where x_t are inputs and h_t are outputs. Source: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ on 01.05.2020.

Vanilla RNN presented in Figure 7, have only one layer in this example - tanh. There is only simple concatenation of two matrices, one from previous cell output and second is current input. Then algorithm applies nonlinear transformation to obtain desired output. This is the moment when gradient vanishing problem arises – as the algorithm go further to the past to optimize parameters, gradient is vanished by transformations made to output from cell to cell.

Figure 7. Architecture of long short-term memory



Note: Example of LSTM architecture with multiple layers, called: input gate, forget gate and output gate. x_t are inputs and h_t are outputs. Source: https://colah.github.io/posts/2015-08-Understanding-LSTMs/ on 01.05.2020.

On the first glance LSTM architecture is much more complex. The main difference is that it has three outputs instead of two like in classical RNN. First one is the memory matrix, which is never transformed using nonlinear functions, thus it is much easier for its optimization and it solves the problem of gradient vanishing. Second and third output are the same one, is passed as classical output from the layer and one is passed to the next input cell. Single cell can be divided into three parts: forget gate, input gate and output gate. Forget gate is a sigmoid function which is applied on concatenation of input x_t and previous output h_{t-1}. This function will return matrix of values in range from 0 to 1, and then this matrix will be multiplied with memory matrix. This operation can be interpreted as parameters, for which values close to 0 will be forgotten in memory matrix. Parameters, for which values are close to 1 will persist in memory matrix. Input gate updates memory matrix adding new weights to it. Firstly, sigmoid function decides about impact of every current input data, next tanh function applies nonlinear transformation on input data. Matrix of weights prepared in such manner is added to memory cell. At the end there is output gate, which benefits from all the effort made to create memory cell, because input is not only transformed by single layer like in normal neural networks, but also it is adjusted by memory matrix, which keeps information about important relations from the past.

LSTM, besides his great successes in speech recognition, can be used for time series analysis forecasting, where memory about long-term dependencies can be crucial for accurate predictions. Training data will be sliced into smaller inputs (sequences) of last n-days values of S&P500 index and output will be next day value of S&P500 index. In terms of input and output it is the same as simple AR(n) process. For system generating purposes, this research use 252 days of rolling training window – model is trained on training window, then it is used for predicting values each day on testing window, after testing window period is over, model is

refitted again on training window and algorithm repeats until last testing window will meet end of time series.

Generating signals is similar to ARIMA based strategy, if predicted value is higher than current value plus fees – generate buy signal. If predicted value is lower than current value plus fees – generate sell signal, otherwise generate stop signal.

4.2.2 Hyperparameters

One of the biggest problems connected with neural networks is the number of hyperparameters and its optimization problems due to high time complexity of compiling the model. There are two ways of solving this issue - select one training sample and use cross-validation to select best hyperparameters on training sample and use this one model for entire testing period. Second method is to use heuristic methods and literature to select optimal hyperparameters and for that reason gain time for refitting model several times which can enables to perform training on rolling window. In this study second method was chosen, because in order to get reliable results using first method, training sample need to be significantly bigger than testing sample. In order to test this method on the last 20 years, model should be tested on the last 80 years which could be time consuming and market relations could move importantly in such a long period. For that reason, it is expected to obtain more accurate results specifying hyperparameters at the start and refit model during our sample several times.

Looking on LSTM architecture there can be distinguished several activation functions like sigmoid or tanh, nevertheless there is possibility and the need to specify other common for all neural networks hyperparameters. As it was mentioned before, these parameters are chosen using heuristic methods and literature, but results for combinations of other parameters will be presented in sensitivity analysis chapter.

Selected hyperparameters:

- number of units in hidden layers: 30
- length of single input (sequence): 15
- activation function for hidden layer: ReLU
- loss function: Mean Squared Error
- optimizer: Adam
- epochs: 100
- dropout rate: 0.2
- starting learning rate: 0.01

4.3 Signal combination

Risk portfolio theory put enormous focus on diversification and correlation between assets in portfolio. Given two portfolios of assets with the same expected returns, the riskier portfolio will be the one with higher correlation between assets or in other words less diversified. Analogously there is expected lower risk on portfolio of diversified signals generated by various strategies than on individual strategy. Thus, this study combined signals from all strategies into one. Signals are generated by adding signals from six classical strategies treating each buy signal as 1, sell signal as -1 and stop signal as 0. Final signal is divided by number of strategies in this case it is 6. If generated signal is not integer number, for example signal equal to 0.5 is treated as leverage level, so for this example only half of the whole capital is invested. Using this method will probably lower annual returns compared to strategies using always 100% leverage, but there is expected significantly lower risk, measured by standard deviation and maximum drawdowns. Such that this strategy should have higher returns risk weighted returns. Strategy will invest the whole capital only on days when every strategy predicts the same price movements and will not invest at all on days when strategies are giving opposite signals.

4.4 Performance statistics

To define performance of used strategies more precisely, there is need of usage appropriate performance statistics to evaluate returns and risk measures. For this research were selected statistics used in Ryś and Ślepaczuk (2018) which extensively describes process of creating automated trading systems and the evaluation of strategies performance. Following statistics were selected:

- Annualized Return Compounded (ARC):

$$\text{ARC} = 252 * \frac{1}{N}\sum_{i=1}^{N} r_t \tag{3}$$

where:

$r_t - is\ the\ daily\ logarithmic\ rate\ of\ return,$

$N - is\ the\ number\ of\ trading\ days,$

Assumming that there are 252 trading days in a year.

- Annualized Standard Deviation (aSD):

$$\text{aSD} = \sqrt{252} * \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(r_t - \bar{r})^2} \tag{4}$$

where:

$\bar{r} - is\ the\ average\ of\ daily\ logarithmic\ rate\ of\ return,$

- Information Ratio (IR) – can be interpreted as returns weighted by risk:

$$\text{IR} = \frac{ARC}{aSD} \tag{5}$$

- Maximum Drawdown (MD) – maximum percentage drawdown in trading period

$$MD = \underset{i=1,\dots,t;t=1,\dots,N}{\text{Min}}\left(\sum_{j=i}^{t} R_j\right) \tag{6}$$

where:

$R_j - is\ the\ daily\ rate\ of\ return,$

- Average Maximum Drawdown (AMD) – average of MD for each year in trading period

$$AMD = \frac{1}{N}\sum_{i=1}^{n} MD_i^{yearly} \tag{7}$$

where:

$MD_i^{yearly} - is\ the\ MD\ calculated\ on\ one\ year\ period,$

$N\ -\ is\ the\ number\ of\ years\ in\ trading\ period,$

- Maximum Loss Duration (MLD) – the longest time (in years) required to achieve last maximum level of capital
- allRisk – the combined risk measure

$$\text{allRisk} = \frac{aSD*MD*MLD*AMD}{1000} \tag{8}$$

- Annual Return Compounded/Maximum Drawdown (ARCMD):

$$\text{ARCMD} = \frac{ARC}{MD} \tag{9}$$

- Annual Return Compounded/Average Maximum Drawdown (ARCAMD):

$$\text{ARCAMD} = \frac{ARC}{AMD} \tag{10}$$

To obtain complete view of every tested strategy the following statistics are additionally attached to every table with performance statistics: number of transactions (numbTrans) and the number of days when strategy was out of the market (stopSignal).

5. Empirical results

5.1 Classical methods

Using six classical methods presented in methodology section on S&P500 data for the last 20 years, there were generated signals and finally equity lines for each investment which started at the same price as S&P500 on first day of trading. Figures presented in this section contains three elements: buyhold line, capital line and points signals. Buyhold line refers to Buy & Hold strategy which simply means that asset is bought at the first day of trading period and hold.

Capital represents equity line of strategy performance generated by signals which are red points on plots.

5.1.1 ARIMA

In Figure 8 there are presented results of ARIMA strategy, easily there can be spotted one period, which provided most of the positive returns in whole trading period. It was during the financial crisis of 2008 to 2009. Second period with significantly higher returns is 2020 COVID crisis. For other years annual returns tend to fluctuate around slightly below 0. ARIMA model performed outstanding in times of high volatility and big downward trends. To more deeply understand why such behavior, took place, it is important to explain ARIMA forecasting logic based on trained model. Mentioned crises occurred after low volatility upward trend, thus parameters estimated in model tends to forecast values based on this trend. For that reason, if returns for the last day were significantly negative, forecasted value will continue upward trend, so for the next day strategy generates buy signal. However, if the next day will not obtain strong positive returns, autoregressive part will be dominated by negative returns which will cause creating sell signal for the next day. This is the exact behavior, which happened during both crises – often strong negative returns were followed by small positive returns. Similar gains using this fact will make significant profits in reversal strategy. ARIMA based strategy did not outperform benchmark strategy, obtaining 0.13 IR compared to 0.16 IR obtained by benchmark. All performance statistics are presented in Table 2.

Figure 8. ARIMA strategy in comparison to benchmark strategy

Note: Capital represents equity line of ARIMA strategy, using 504 training days and identifying parameters using Akaike information criterion. Buyhold represents equity line for S&P500 index for Buy&Hold strategy.

Table 2. ARIMA performance statistics

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
ARIMA	2.06	0.13	15.44	57.74	12.36	11.02	12.15	0.04	0.17	1548	2786

Note: Performance statistics of ARIMA strategy, using 504 training days and identifying parameters using Akaike information criterion.

5.1.2 Moving average crossover

Figure 9 presents results of moving average crossover strategy. This strategy outperformed Buy & Hold strategy until 2016. From 2011 when it almost tripled Buy & Hold, it started to fluctuate around 2300$. Due to methodology of choosing parameters for short moving average and long moving average there is no specific combination during the whole trading period. However this strategy performed much better during long trends – for example 2001-2008 period - and suffer most loses on rapidly changing situation, because before historical moving average could reflect new trend, prices were back on track – for example drawdowns in the late 2015 and the early 2016. Moving average crossover strategy did not outperform benchmark strategy, obtaining 0.06 IR compared to 0.16 IR obtained by benchmark. All performance statistics are presented in Table 3.

Figure 9. Moving average crossover strategy in comparison to benchmark strategy

Note: Capital represents equity line of Moving Average Crossover strategy, using rolling training-testing window with 504 training days and 504 testing days and refit every 63 days. Buyhold represents equity line for S&P500 index Buy&Hold strategy.

Table 3. Moving average performance statistics

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.15	15.92	0.05	0.18	1	0
MA Crossover	1.25	0.06	19.95	53.26	17.49	11.14	20.70	0.02	0.07	57	0

Note: Performance statistics of Moving Average Crossover strategy, using rolling training-testing window with 504 training days and 504 testing days and refit every 63 days.

5.1.3 Contrarian

Figure 10 presents results of contrarian strategy. The most interesting periods in this performance are again like in case of ARIMA – two crises, financial 2008-2009 and 2020 COVID. In case of contrarian strategy the gains in the period of crises are even higher reaching over 8800$ at June of 2009. Contrarian strategy assumes reacting opposite to the market, if market gained profits over the last couple of days (length is selected during optimization process), strategy expects prices to fall on the following day. Thus, if market is highly volatile in day to day returns this should be profitable based on this method.

Figure 10. Contrarian strategy in comparison to benchmark strategy

Note: Capital represents equity line of Contrarian signal strategy, using rolling training-testing window with 504 training days and 504 testing days and refit every 63 days and the combination of 1, 2, 5, 10 days cumulative returns. Buyhold represents equity line for S&P500 index for Buy&Hold strategy.

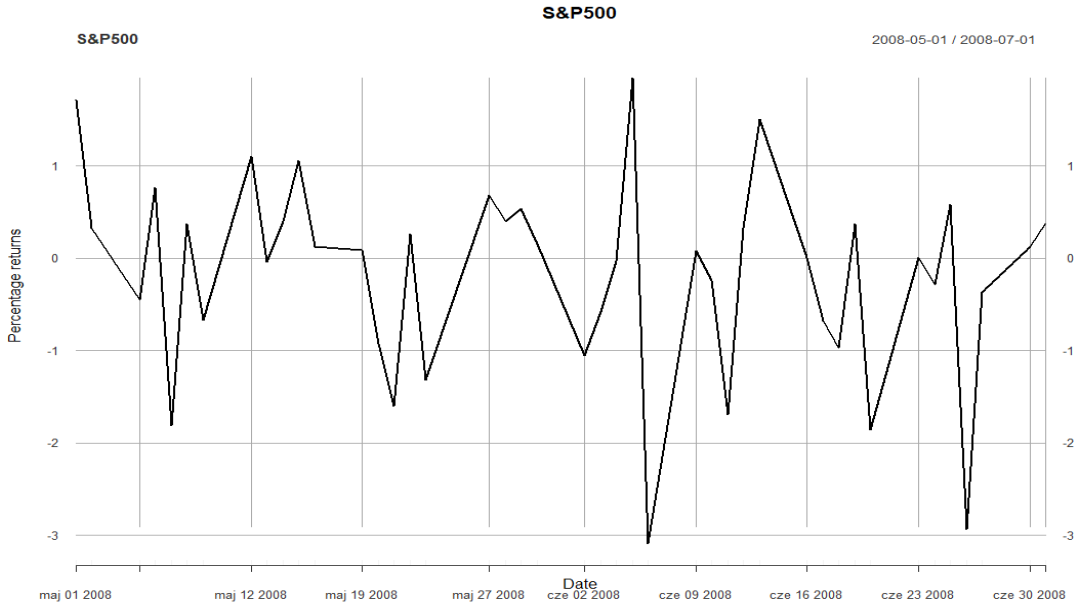
Table 4. Contrarian performance statistics.

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.15	15.92	0.05	0.18	1	0
Contrarian	6.95	0.35	19.82	68.38	15.88	10.79	23.23	0.10	0.44	2705	3

Note: Performance statistics of Contrarian strategy, using rolling training-testing window with 504 training days and 504 testing days and refit every 63 days and the combination of 1, 2, 5, 10 days cumulative returns.

In order to visualize market returns changes, Figure 11 presents day to day returns in the highly volatile period from May 2008 to June 2008. As it is shown on a Figure 11 returns were highly volatile, and they did not perform any continuation of the trend. Extension to this strategy can be done by adding volatility index for example VIX and use this strategy only for periods when VIX instrument is above certain level, it should limit number of trades when this strategy is not that successful and is generating losses. Contrarian strategy outperformed benchmark strategy, obtaining 0.35 IR compared to 0.16 IR obtained by benchmark. All performance statistics are presented in the Table 4.

Figure 11. Returns of S&P500 index during highly volatile period



Note: S&P500 returns during the period 01.05.2008-01.07-2020.

5.1.4 Momentum

Figure 12 presents results of momentum strategy. During whole trading period this strategy fluctuates around starting capital with small decrease trend, which could be caused by transaction costs. There are no specific periods when this strategy performed significantly good or significantly bad. However, this strategy does not have huge drawdowns and looks less

volatile than simple Buy & Hold strategy. Can be expected that this strategy should perform extremely well in the period from 2010 to 2018, where S&P500 value was in strong upward trend without any significant drawdowns. However, small drawdowns caused this strategy to change its signal, when it was completely unnecessary, but in time of crisis 2008-2009 and 2020 this fast adaptation protected from enormous drawdowns. Momentum strategy did not outperform benchmark strategy, obtaining -0.19 IR compared to 0.16 IR obtained by benchmark. All performance statistics are presented in Table 5.

Figure 12. Momentum strategy in comparison to benchmark strategy

Note: Capital represents equity line of Contrarian strategy, using rolling training-testing window with 504 training days and 504 testing days and refit every 63 days and the combination of 7, 21, 63, 126 days cumulative returns. Buyhold represents equity line for S&P500 index for Buy&Hold strategy.

Table 5. Momentum strategy performance statistics

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
Momen-tum	-3.87	-0.19	19.94	79.39	18.67	17.76	52.47	-0.05	-0.21	399	2786

Note: Performance statistics of Contrarian strategy, using rolling training-testing window with 504 training days and 504 testing days and refit every 63 days and the combination of 7, 21, 63, 126 days cumulative returns.

5.1.5 Volatility breakout

Figure 13 presents results of volatility breakout strategy. The aim of this method was to not risk, during uncertain market times, by being out of the market for most of the trading period and make trades only if there are strong downward trends or low volatility upward trends without many drawdowns. This goal has been achieved, because performance line is stable

without big drawdowns and there are many signals to be out of the market. Such approach is resulting in extremely low risk strategy, which still achieves gains over the long period. Volatility breakout strategy did not outperform benchmark strategy, obtaining 0.10 IR compared to 0.16 IR obtained by benchmark. All performance statistics are presented in the Table 6.

Figure 13. Volatility breakout strategy in comparison to benchmark strategy

Note: Capital represents equity line of Volatility breakout strategy, using rolling training-testing window with 504 training days and 504 testing days and refit every 63 days. During optimization the following parameters used: days for calculating distribution of volatility: 63, 126, 252, 504, 1008, quantile level q_1: 0.6, 0.7, 0.8, 0.9, quantile level q_2: 0, 0.1, 0.2, 0.3, 0.4.

Table 6. Volatility breakout performance statistics

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
Volatility Breakout	0.90	0.11	8.33	28.57	7.62	7.95	1.44	0.03	0.12	1014	3620

Note: Performance statistics of Volatility breakout strategy, using rolling training-testing window with 504 training days and 504 testing days and refit every 63 days. During optimization the following parameters used: days for calculating distribution of volatility: 63, 126, 252, 504, 1008, quantile level q_1: 0.6, 0.7, 0.8, 0.9, quantile level q_2: 0, 0.1, 0.2, 0.3, 0.4.

5.1.6 Macro factor

Figure 14. presents results of macro factor strategy, which uses jobless claims as early indicator of crisis. Most of the trading period the strategy generated buy signals, based on condition of US economy calculated by unemployment indicator and stop signals, when the economy suffered small turmoil. Sell signals defining poor condition of U.S. economy were generated only few times during this period but with high accuracy. Only 2020 COVID crisis was not

predicted by the indicator. The reason for this situation is that market started to panic and sell, much before any economic damages were done. However, when record of jobless claims were filed S&P500 prices were already rebounding from the early 2020 bottom. Macro factor strategy outperformed benchmark strategy, obtaining 0.25 IR compared to 0.16 IR obtained by benchmark. All performance statistics are presented in Table 7.

Figure 14. Macro factor strategy in comparison to benchmark strategy

Note: Capital represents equity line of Macro factor strategy, using rolling training-testing window with 504 training days and 504 testing days and refit every 63 days. During optimization the following parameters used: days for calculating distribution of initial jobless claims: 63, 126, 252, 504, 1008, quantile level q_1: 0.6, 0.7, 0.8, 0.9, quantile level q_2: 0, 0.1, 0.2, 0.3, 0.4.

Table 7. Macro factor performance statistics

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
Macro factor	4.36	0.25	17.21	40.22	13.27	6.00	5.51	0.11	0.33	352	1620

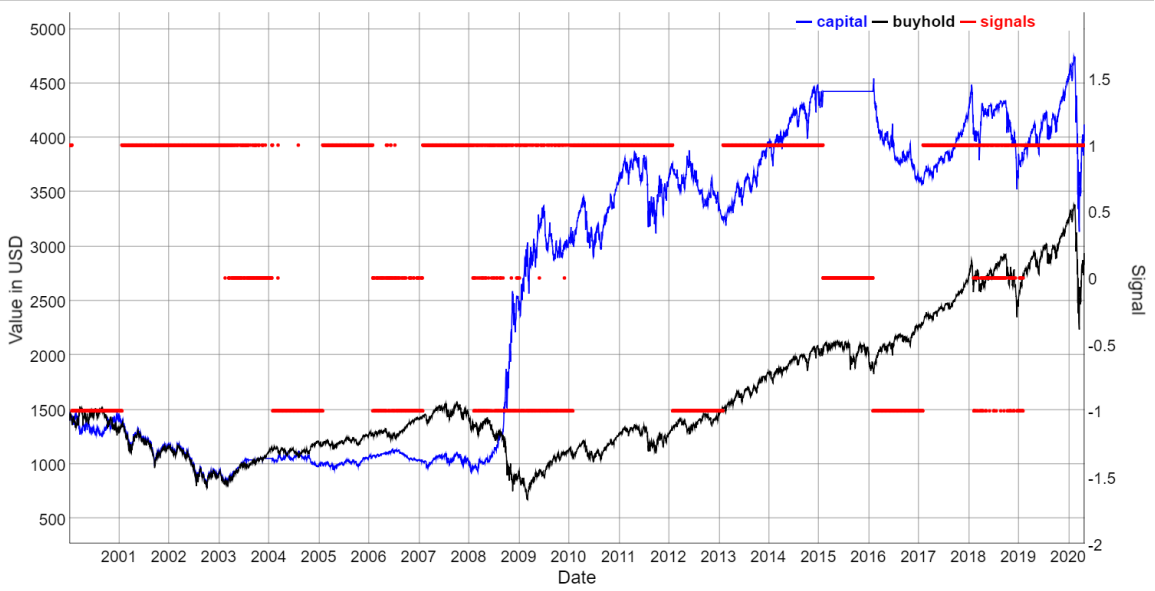
Note: Performance statistics of Macro factor signal generating strategy, using rolling training-testing window with 504 training days and 504 testing days and refit every 63 days. During optimization the following parameters used: days for calculating distribution of initial jobless claims: 63, 126, 252, 504, 1008, quantile level q_1: 0.6, 0.7, 0.8, 0.9, quantile level q_2: 0, 0.1, 0.2, 0.3, 0.4.

5.1.7 LSTM

Figure 15 presents results of LSTM strategy. Strong undervaluation of predicted values can be observed, because despite the upward trends of S&P500 LSTM model still predicts decrease in prices in most cases. Another observation is that it tends to hold the same signal for longer period in comparison to ARIMA and it is rather not changing its signal before refitting. More conclusions about behavior of this method can be done after sensitivity analysis and changing

hyperparameters which took place in chapter 5. LSTM based strategy outperformed benchmark strategy, obtaining 0.27 IR compared to 0.16 IR obtained by benchmark. All performance statistics are presented in the Table 8.

Figure 15. Long short-term memory strategy in comparison to benchmark strategy



Note: Capital represents equity line of generating signal strategy based on LSTM model. Model is using rolling 252 days training window. Hyperparameters used: number of units in hidden layers: 30, length of single input: 15, activation function for hidden layer: ReLU, loss function: Mean Squared Error, optimizer: Adam, epochs: 100, dropout rate: 0.2, starting learning rate: 0.01.

Table 8. Long short-term memory strategy performance statistics

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
LSTM	5.14	0.27	19.20	50.00	15.87	7.72	11.76	0.10	0.32	447	566

Note: Performance statistics of strategy based on LSTM model. Model is using rolling 252 days training window. Hyperparameters used: number of units in hidden layers: 30, length of single input: 15, activation function for hidden layer: ReLU, loss function: Mean Squared Error, optimizer: Adam, epochs: 100, dropout rate: 0.2, starting learning rate: 0.01.

5.2 Combination of signals

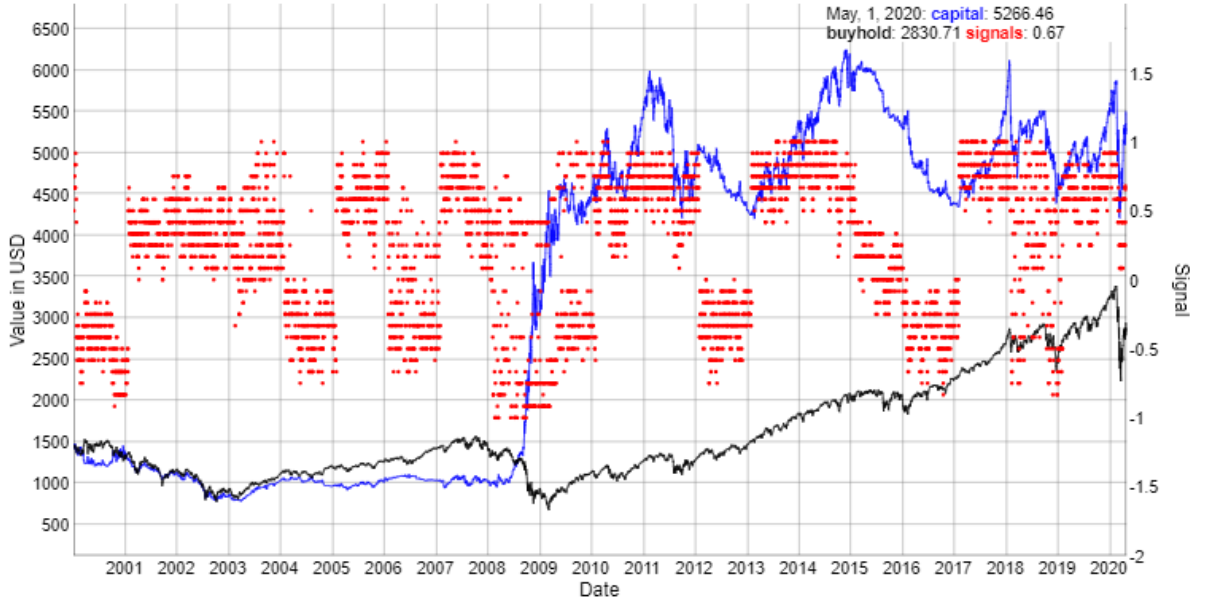
Final step of implemented strategies in this research was to combine them into one system of generating signals. Results of combination strategy are presented on Figure 16. There can be spotted main advantages of combining all signals. At first in periods when most of strategies are generating same signal for example 2008-year, performance of system is good and what is more only on periods with full agreement, the whole capital is invested. This resulted in significantly smaller risk, even though final returns are like Buy & Hold strategy. The usage of combination of signals allows multiplying leverage levels and remain similar risk levels to

Buy & Hold, but in this case returns of leveraged strategy will be significantly higher. Figure 17 presents results for combined strategy with leverage equals to 200%.

Figure 16. Combined strategy of classical methods and LSTM in comparison to benchmark strategy

Note: Combined represents equity line of combined strategy using signals from LSTM – weight 0.5, sum of classical method signals – weight 0.5. Classical represents equity line of combined strategy using signals from classical methods all with the same weight.

Figure 17. Combined strategy results with leverage multiplied by 2



Note: Capital represents equity line of leveraged combined strategy using signals from LSTM – weight 0.5, sum of classical method signals – weight 0.5. Using leverage in the level of 200%.

Table 9 presents statistics for every strategy used in this study. Buy & Hold strategy obtained 3.23 annualized compounded returns with annualized standard deviation equal 19.49 which resulted in 0.16 IR. It also took more than 7 years for achieving maximum price after

2001 downward trend. Maximum drawdown was almost 65%. Table 9 shows that momentum strategy performed the worst achieving -3.87 ARC with almost the same annualized standard deviation as benchmark. On the other hand, the best performing strategy was contrarian with 6.95 ARC and 0.35 IR which outperforms benchmark strategy doubling its results. Despite huge loses for macro factor in 2020 COVID crisis this strategy managed to obtain second best IR (0.25) and second best allRisk measure (5.5). Volatility breakout strategy statistics perfectly represents, what is the purpose of the risk averse strategies and it obtain the lowest maximum drawdown (28.5%), lowest annualized standard deviation, which led to the lowest allRisk statistic (1.44), keeping IR on still satisfying level of 0.11. There is an obvious correlation between number of days out of the market and the risk measure, which is obvious, because there is no risk of losing capital when being out of the market. Thus, adding any filtering to strategy, which will reduce the number of transactions and investing days will improve risk statistics, in exchange for changes in returns.

Combined strategy outperforms every strategy in every risk measure except aSD, which is great achievement in comparison to diversifying the risk of using only one strategy instead of combined signals, at the same time ARC is almost equal to the benchmark strategy. However, strategy with increased leverage achieved 6.52 ARC compared to 3.23 of benchmark with the same allRisk measure and comparable annualized standard deviation.

Table 9. Performance statistics of results

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
ARIMA	2.06	0.13	15.44	57.74	12.36	11.02	12.14	0.04	0.17	1548	2786
Contrarian	6.95	0.35	19.82	68.38	15.88	10.79	23.31	0.10	0.44	2705	3
Momen-tum	-3.87	-0.19	19.94	79.39	18.67	17.76	52.47	-0.05	-0.21	399	0
MA Crossover	1.25	0.06	19.95	53.26	17.49	11.14	20.70	0.02	0.07	57	0
Macro factor	4.36	0.25	17.21	40.22	13.27	6.00	5.51	0.11	0.33	352	1620
Voliatlity breakout	0.90	0.11	8.32	28.56	7.62	7.95	1.44	0.03	0.12	1014	3620
Classical methods	1.36	0.15	9.20	26.91	8.05	9.18	1.83	0.05	0.17	3603	561
LSTM	5.14	0.27	19.20	50.00	15.87	7.72	11.76	0.10	0.32	447	566
Combined	3.82	0.54	10.64	27.38	8.83	8.62	2.22	0.14	0.43	3688	98
Combined leverged	6.52	0.31	21.23	50.38	17.32	8.64	16.01	0.13	0.38	3688	98

Note: Performance statistics of every method used in this research.

6. Sensitivity analysis

6.1 Classical methods sensitivity analysis

To ensure that presented results are not completely random performances due to parameters selection performed at the start of research. There is a need to generate results with different parameters and check how much it will affect the performance. Then strategies were compared to default parameters strategies described in the methodology section and there was only one change to parameters for each new tested parameter setting.

6.1.1 ARIMA

ARIMA due to optimization process has only one parameter, which was chosen manually. It was the size of rolling training window. Thus, it is only this parameter, which was changed and tested in the sensitivity analysis. Figure 18 and Table 10, present results for three additional ARIMA models – 126 days rolling window, 504 days rolling window and training window which does not change the starting point but new data is added to already selected training window. Except reduced rolling window to 126 days, results are stable with similar risk and returns statistics. 126 days probably is too short period for estimating stable parameters and is strongly biased by volatile data.

Figure 18. ARIMA strategy sensitivity analysis

Note: Sensitivity analysis of ARIMA based strategies. Following changes in parameters were tested: 126 training window days, 504 training window days, fixed starting point for training window.

Table 10. Performance statistics of ARIMA strategy sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
ARIMA	2.06	0.13	15.44	57.74	12.36	11.02	12.14	0.04	0.17	1548	2786
ARIMA 504 training days	1.68	0.10	16.44	68.98	13.79	11.37	17.78	0.02	0.12	1773	2389
ARIMA 126 training days	-2.88	-0.19	14.94	67.60	12.20	8.78	10.83	-0.04	-0.24	1005	3393
ARIMA hooked	2.85	0.16	17.88	61.98	14.58	9.03	14.59	0.05	0.19	1999	1822

Note: Performance statistics of sensitivity analysis of ARIMA based strategies. Following changes in parameters were tested: 126 training window days, 504 training window days, expansive training window.

6.1.2 Moving average crossover

Moving average crossover strategy used rolling training-testing window on which it optimized periods used to calculate short and long moving averages. Thus, there are three parameters that could be changed: training window size, testing window size and periods used in optimization. Figure 19. and Table 11. are presenting results using this changed parameters.

Table 11. Performance statistics of moving average crossover sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
MA Crossover	1.25	0.06	19.95	53.26	17.49	11.14	20.70	0.02	0.07	57	0
MA Crossover 126 testing days	-1.52	-0.08	19.99	68.53	19.27	11.14	29.40	-0.02	-0.08	61	0
MA Crossover 252 testing days	0.99	0.05	19.98	55.16	18.04	9.79	19.46	0.02	0.06	57	0
MA Crossover 21 training days	0.60	0.03	19.95	53.90	18.22	11.14	21.83	0.01	0.03	59	0
MA Crossover 126 training days	2.08	0.10	19.95	47.09	16.86	5.61	8.88	0.04	0.12	47	0
MA Crossover changed periods	-4.30	-0.22	19.95	76.47	20.90	17.76	56.62	-0.06	-0.21	81	0

Note: Performance statistics of sensitivity analysis of moving average crossover strategies. Following changes in parameters were tested: 126 testing window days, 252 testing window days, 126 trading window days, 21 trading window days, changed periods: Short moving average: 5, 10, 15, 20; Long moving average: 50, 100,150 ,170.

There are two strategies which performed significantly worse and one strategy with much better results than default one. Similarly, to ARIMA reducing size of training window cause much worse performance. However, extending testing window to 126 days from default 63 days improved all measured statistics. Strategy with changed periods used:

- Short moving average: 5, 10, 15, 20
- Long moving average: 50, 100,150 ,170

These moving averages are reduced version of default parameters by excluding the longest periods. In results section there was stated hypothesis that longer moving averages could not react as fast as it is needed during volatile periods. However, sensitivity analysis results shown that reducing longer moving averages sour the performance.

Figure 19. Moving average crossover sensitivity analysis

Note: Sensitivity analysis of moving average crossover strategies. Following changes in parameters were tested: 126 testing window days, 252 testing window days, 126 trading window days, 21 trading window days, changed periods: Short moving average: 5, 10, 15, 20; Long moving average: 50, 100,150 ,170.

6.1.3 Contrarian

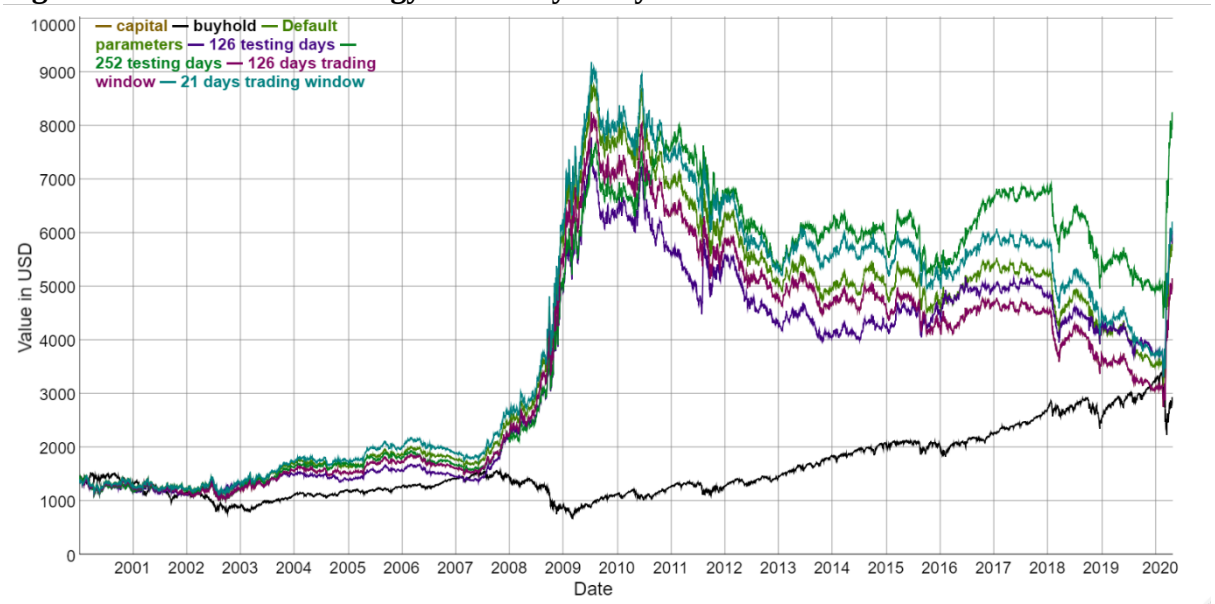
Contrarian strategy used rolling training-testing window same as moving average crossover strategy, which resulted in changing size of training and testing window. Figure 20. and Table 12 present results using these changed parameters. Sensitivity analysis results presented that contrarian strategy is robust for changes in parameters and all performance statistics are stable.

Table 12. Performance statistics of contrarian strategy sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
Contrarian	6.95	0.35	19.82	68.38	15.88	10.79	23.23	0.10	0.44	2705	3
Contrarian 252 testing days	8.60	0.43	19.82	50.06	14.82	9.11	13.39	0.17	0.58	2737	0
Contrarian 126 testing days	7.06	0.36	19.82	62.19	15.77	10.79	20.98	0.11	0.45	2545	0
Contrarian 21 training days	7.20	0.36	19.83	67.93	15.80	10.79	22.97	0.11	0.46	2717	0
Contrarian 126 training days	6.29	0.32	19.83	70.37	16.15	10.79	24.33	0.09	0.39	2695	0

Note: Performance statistics of sensitivity analysis of contrarian strategies. Following changes in parameters were tested: 126 testing window days, 252 testing window days, 126 trading window days, 21 trading window days.

Figure 20. Contrarian strategy sensitivity analysis



Note: Sensitivity analysis of contrarian strategies. Following changes in parameters were tested: 126 testing window days, 252 testing window days, 126 trading window days, 21 trading window days.

6.1.4 Momentum

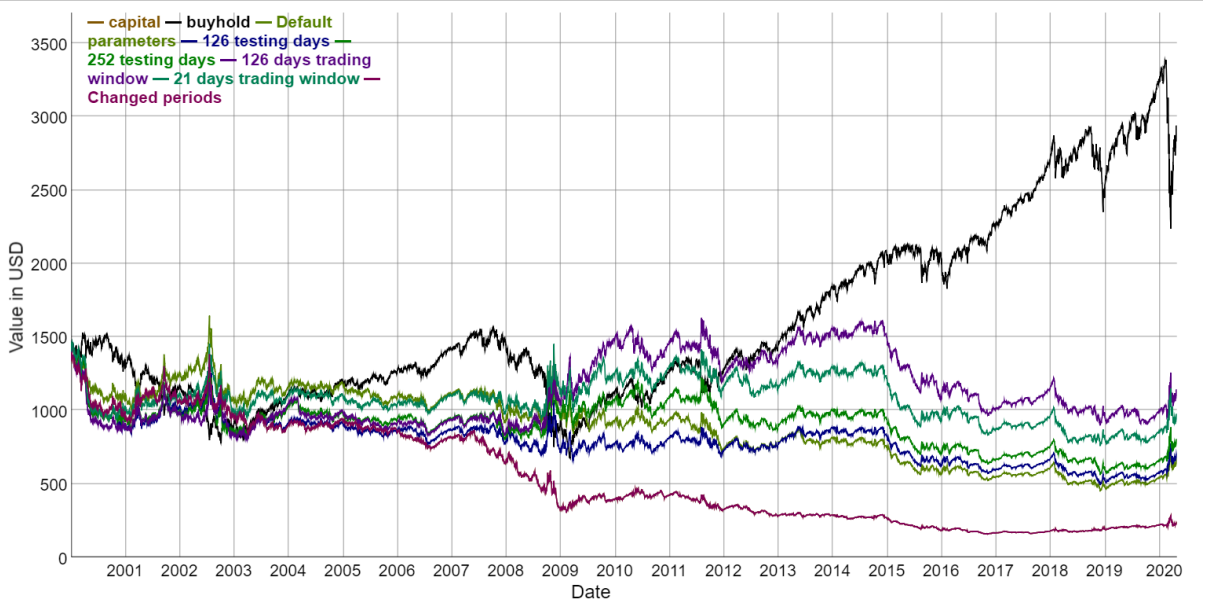
Momentum strategy used rolling training-testing window, for that reason there were tested three parameters changes: training window length, testing window length and periods for calculating compounded returns. Figure 21 and Table 13 presents results using these changed parameters. This method had the worst performance using default parameters. Changing parameters did not improve overall results. Two main differences were on changing testing window size, which again boosted performance, however period was changed replacing two longest compounded returns (63 and 126 days) for two shorter ones (5 days and 42 days). Similarly, to moving average strategy reducing longer periods worsen the results.

Table 13. Performance statistics of momentum strategy sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
Momentum Crossover	-3.87	-0.19	19.94	79.39	18.67	17.76	52.47	-0.05	-0.21	399	0
Momentum 252 testing days	-2.91	-0.15	19.94	72.61	19.05	20.26	55.87	-0.04	-0.15	383	0
Momentum 126 testing days	-3.56	-0.18	19.94	75.99	18.72	20.26	57.46	-0.05	-0.19	388	0
Momentum 21 training days	-1.96	-0.10	19.94	63.25	18.05	20.26	46.11	-0.03	-0.11	362	0
Momentum 126 training days	-1.18	-0.06	19.93	56.39	18.52	9.82	20.44	-0.02	-0.06	353	0
Momentum changed periods	-8.95	-0.45	19.93	91.89	20.52	20.29	76.25	-0.10	-0.44	523	0

Note: Performance statistics of sensitivity analysis of momentum strategies. Following changes in parameters were tested: 126 testing window days, 252 testing window days, 126 trading window days, 21 trading window days, changed periods of calculating compounded returns for 5, 7, 21, 42 days.

Figure 21. Momentum strategy sensitivity analysis



Note: Sensitivity analysis of momentum strategies. Following changes in parameters were tested: 126 testing window days, 252 testing window days, 126 trading window days, 21 trading window days, changed periods of calculating compounded returns for 5, 7, 21, 42 days.

6.1.5 Volatility breakout

Volatility breakout strategy used rolling testing-training window, thus there were tested changes in: training size window, testing size window and number of quantiles (reducing middle quantiles and leaving 0.1, 0.2, 0.8, 0.9. Figure 22 and Table 14 presents results using these changed parameters. This strategy performance is highly dependent on manually chosen

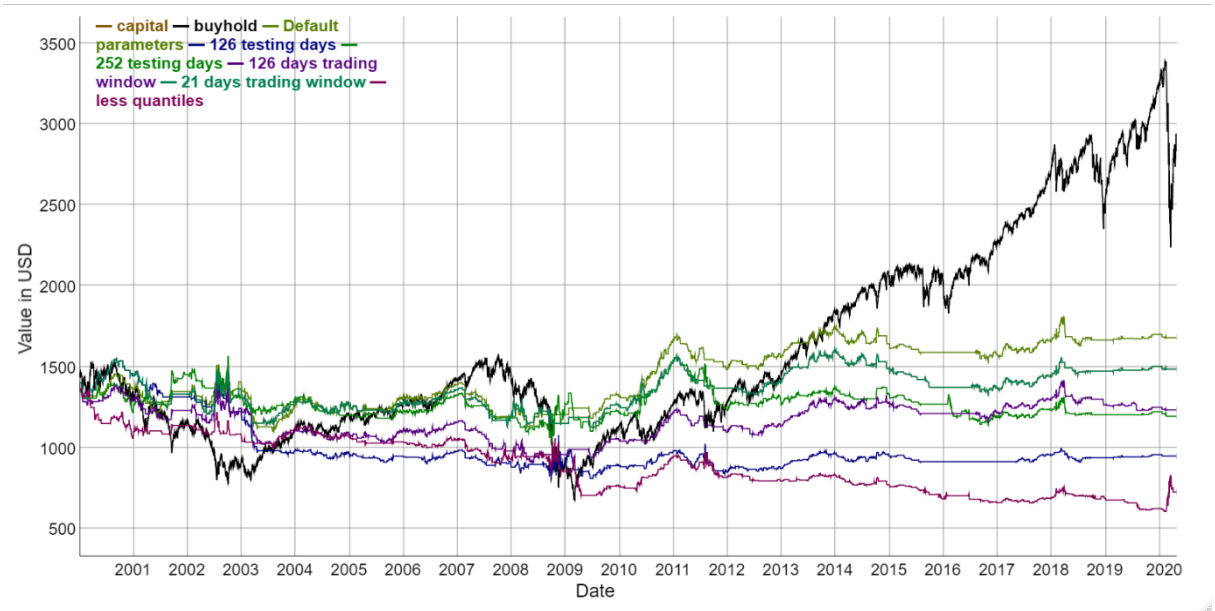
parameters. Any change significantly sore the results. Method is not robust and should be change the way of selecting parameters or optimization process, to obtain more stable method.

Table 14. Performance statistics of volatility breakout strategy sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
Volatility Breakout	0.90	0.11	8.32	28.56	7.62	7.95	1.44	0.03	0.12	1014	3620
Volatility Breakout 252 testing days	-1.93	-0.17	11.29	52.83	8.35	19.59	9.76	-0.04	-0.23	944	3843
Volatility Breakout 126 testing days	-0.79	-0.07	10.47	35.32	8.43	17.54	5.47	-0.02	-0.09	1053	3751
Volatility Breakout 21 trading days	0.29	0.03	8.97	31.19	7.74	10.31	2.23	0.01	0.04	984	3547
Volatility Breakout 126 trading days	-0.62	-0.07	8.45	38.30	7.98	18.16	4.69	-0.02	-0.08	1086	3540
Volatility Breakout less quantiles	-3.24	-0.26	12.65	62.13	10.64	20.29	16.97	-0.05	-0.30	1015	3819

Note: Performance statistics of sensitivity analysis of volatility breakout strategies. Following changes in parameters were tested: 126 testing window days, 252 testing window days, 126 trading window days, 21 trading window days, optimized quantiles: 0.1, 0.2, 0.8, 0.9.

Figure 22. Volatility breakout strategy sensitivity analysis



Note: Sensitivity analysis of volatility breakout strategies. Following changes in parameters were tested: 126 testing window days, 252 testing window days, 126 trading window days, 21 trading window days, optimized quantiles: 0.1, 0.2, 0.8, 0.9.

6.1.6 Macro factor

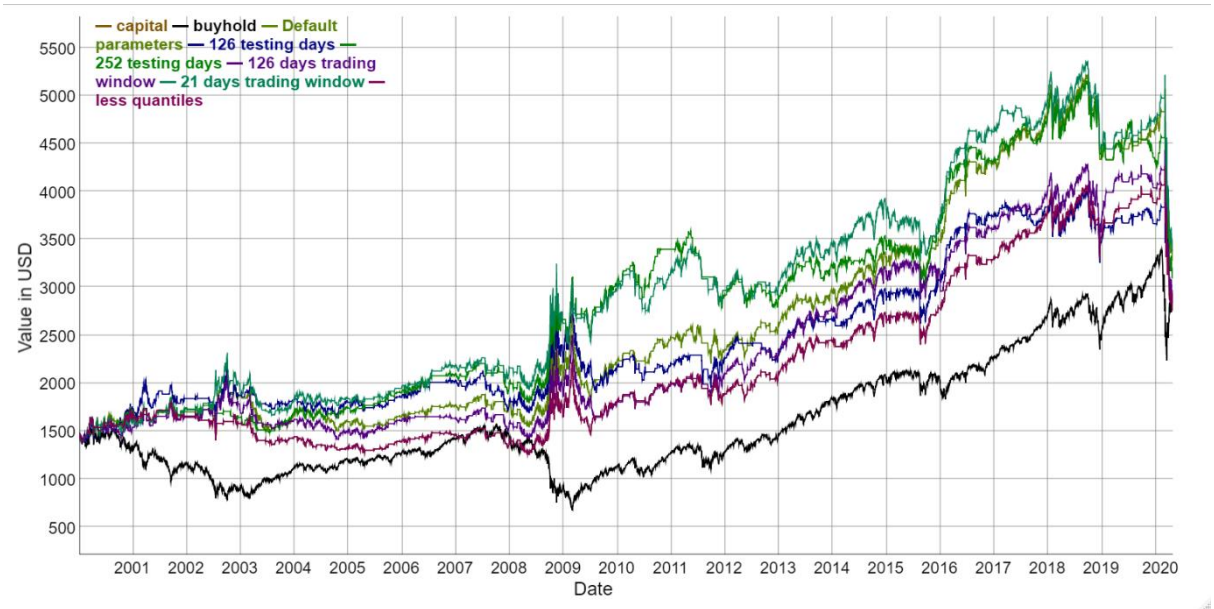
Macro factor strategy also used rolling training-testing window, thus there were tested changes in: training size window, testing size window and number of quantiles used for optimization (reducing middle quantiles, leaving 0.1, 0.2, 0.8, 0.9). Figure 23 and Table 15 presents results using these changed parameters. This strategy achieved almost the same results for every tested parameter. Thus, results are robust and are not sensitive to parameters change.

Table 15. Performance statistics of macro factor strategy sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
Macro factor	4.36	0.25	17.21	40.22	13.27	5.99	5.51	0.11	0.33	352	1620
Macro factor 252 testing days	4.02	0.24	17.04	37.75	13.11	5.98	5.04	0.11	0.31	391	1859
Macro factor 126 testing days	4.08	0.25	16.08	43.05	12.24	4.59	3.89	0.10	0.33	394	1841
Macro factor 21 trading days	4.50	0.27	16.80	40.04	12.88	5.93	5.14	0.11	0.35	381	1764
Macro factor 126 trading days	3.70	0.22	17.20	37.92	13.51	5.99	5.28	0.10	0.27	339	1715
Macro factor less quantiles	3.51	0.21	16.39	37.92	13.21	7.06	5.80	0.09	0.26	364	2121

Note: Performance statistics of sensitivity analysis of volatility breakout strategies. Following changes in parameters were tested: 126 testing window days, 252 testing window days, 126 trading window days, 21 trading window days, optimized quantiles: 0.1, 0.2, 0.8, 0.9.

Figure 23. Macro factor strategy sensitivity analysis



Note: Sensitivity analysis of moving average crossover strategies. Following changes in parameters were tested: 126 testing window days, 252 testing window days, 126 trading window days, 21 trading window days, optimized quantiles: 0.1, 0.2, 0.8, 0.9.

6.2 LSTM

Machine learning neural network models require defining many hyperparameters before the model optimization process. What is more, the estimation process is highly time consuming. Thus, LSTM model strategy in this research did not optimize any of the parameters during trading sample and all parameters were chosen manually using heuristic methods and literature. For that reason it is crucial to tests all selected settings for models in sensitivity analysis and check whether results are stable.

6.2.1 Training window

The first tested parameter was training window size. Figure 24 and Table 16 present results using this changed parameter. Decreasing the number of days in training window negatively affected performance. However, increasing the number of days to 504 in training window positively affected performance. Further increase in number of days worsen the results significantly. Thus, LSTM results are possibly not robust to changes in trading window size.

Table 16. Performance statistics of LSTM strategy training window sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
LSTM	5.14	0.27	19.20	50.00	15.87	7.72	11.76	0.10	0.32	447	566
LSTM 252 training days	0.24	0.01	19.41	57.28	16.46	9.77	17.88	0.00	0.02	45	417
LSTM 504 training days	6.26	0.31	19.95	42.96	16.01	6.20	8.51	0.15	0.39	328	159
LSTM 756 training days	-6.23	-0.32	19.29	90.29	18.76	17.76	58.02	-0.07	-0.33	103	527
LSTM 1008 training days	-5.71	-0.29	19.81	88.40	18.35	17.76	57.05	-0.06	-0.31	221	47

Note: Performance statistics of sensitivity analysis of training window in LSTM strategy Following changes in parameter were tested: 126 training window days, 504 training window days, 756 training window days, 1008 training window days.

Figure 24. LSTM strategy training window sensitivity analysis

Note: Sensitivity analysis of training window in LSTM strategy Following changes in parameter were tested: 126 training window days, 504 training window days, 756 training window days, 1008 training window days. Default model is using rolling 252 days training window. Hyperparameters used: number of units in hidden layers: 30, length of single input: 15, activation function for hidden layer: ReLU, loss function: Mean Squared Error, optimizer: Adam, epochs: 100, dropout rate: 0.2, starting learning rate: 0.01.

6.2.2 Sequence

Next tested parameter was the length of sequence. Figure 25 and Table 17 presents results using this changed parameter. Two tested options had worse results than LSTM model with default parameters. What is more, there can be spotted that their paths are significantly different. Having this knowledge LSTM model is not robust to changes in sequence length. There is a chance that changes in tested parameter are too big and future research should test more options.

Table 17. Performance statistics of LSTM strategy sequence sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
LSTM	5.14	0.27	19.20	50.00	15.87	7.72	11.76	0.10	0.32	447	566
LSTM 20 days sequence	1.00	0.05	19.30	52.43	16.31	8.72	14.39	0.02	0.06	334	495
LSTM 10 days sequence	-3.15	-0.16	19.44	78.52	17.91	17.76	48.54	-0.04	-0.18	103	284

Note: Performance statistics of sensitivity analysis of sequence in LSTM strategy. The following changes in parameters were tested: 20 days sequence, 10 days sequence.

Figure 25. LSTM strategy sequence sensitivity analysis

Note: Sensitivity analysis of sequence in LSTM strategy. The following changes in parameters were tested: 20 days sequence, 10 days sequence. Default model is using rolling 252 days training window. Hyperparameters used: number of units in hidden layers: 30, length of single sequence: 15, activation function for hidden layer: ReLU, loss function: Mean Squared Error, optimizer: Adam, epochs: 100, dropout rate: 0.2, starting learning rate: 0.01.

6.2.3 Epochs

Next tested parameter was the number of epochs. Figure 26 and Table 18 present results using this changed parameter. Both options performed significantly worse without any similarity to LSTM method with default parameters. There is no robustness for changes in number of epochs. Similarly to sequence parameter there is a chance that changes in tested parameter are too big and the future research should test much more options.

Table 18. Performance statistics of LSTM strategy epochs sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARCA MD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
LSTM	5.14	0.27	19.20	50.00	15.87	7.72	11.76	0.10	0.32	447	566
LSTM 50 epochs	-4.71	-0.24	19.45	85.98	18.65	20.07	62.59	-0.05	-0.25	50	497
LSTM 150 epochs	-8.37	-0.43	19.45	90.79	19.56	20.06	69.33	-0.09	-0.43	118	353

Note: Performance statistics of sensitivity analysis of epochs in LSTM strategy. The following changes in parameters were tested: 50 epochs, 150 epochs.

Figure 26. LSTM strategy epochs sensitivity analysis

Note: Sensitivity analysis of epochs in LSTM strategy. The following changes in parameters were tested: 50 epochs, 150 epochs. Default model is using rolling 252 days training window. Hyperparameters used: number of units in hidden layers: 30, length of single input: 15, activation function for hidden layer: ReLU, loss function: Mean Squared Error, optimizer: Adam, epochs: 100, dropout rate: 0.2, starting learning rate: 0.01.

6.2.4 Dropout rate

Next tested parameter was dropout rate value. Figure 27 and Table 19 presents results using this changed parameter. Both options performed significantly worse without any similarity to LSTM method with default parameters. There is no robustness for changes in the value of dropout rate.

Table 19. Performance statistics of LSTM strategy dropout rate sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARCA MD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
LSTM	5.14	0.27	19.20	50.00	15.87	7.72	11.76	0.10	0.32	447	566
LSTM 0.1 dropout rate	0.47	0.02	19.75	62.05	16.60	11.14	22.66	0.01	0.03	66	315
LSTM 0.3 dropout rate	-7.97	-0.40	19.72	90.12	20.19	20.07	72.00	-0.08	-0.39	120	214

Note: Performance statistics of sensitivity analysis of dropout rate in LSTM strategy. The following changes in parameters were tested: 0.1 dropout rate, 0.3 dropout rate.

Figure 27. LSTM strategy dropout rate sensitivity analysis

Note: Sensitivity analysis of dropout rate in LSTM strategy. The following changes in parameters were tested: 0.1 dropout rate, 0.3 dropout rate. Default model is using rolling 252 days training window. Hyperparameters used: number of units in hidden layers: 30, length of single input: 15, activation function for hidden layer: ReLU, loss function: Mean Squared Error, optimizer: Adam, epochs: 100, dropout rate: 0.2, starting learning rate: 0.01.

6.2.5 Learning rate

Next tested parameter was learning rate. Figure 28 and Table 20 presents results using this changed parameter. Decreasing value of learning rate improved the performance of the strategy, with obtaining similar path to default LSTM. Increase of learning rate value completely changed results and signals generated by strategy. This change could be too big and model could not find the minimum of loss function with optimization of parameters. LSTM model results is not robust to changes in learning rate.

Table 20. Performance statistics of LSTM strategy learning rate sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
LSTM	5.14	0.27	19.20	50.00	15.87	7.72	11.76	0.10	0.32	447	566
LSTM 0.05 learning rate	-8.55	-0.43	19.89	89.09	20.53	19.32	70.30	-0.10	-0.42	35	7
LSTM 0.001 learning rate	7.43	0.39	19.02	39.19	14.84	5.43	6.00	0.19	0.50	672	920
LSTM 0.005 learning rate	-1.29	-0.06	19.54	66.35	17.52	6.21	14.11	-0.02	-0.07	320	614
LSTM 0.0001 learning rate	4.34	0.23	18.54	50.22	14.92	6.25	8.68	0.08	0.29	946	996

Note: Performance statistics of sensitivity analysis of learning rate in LSTM strategy. The following changes in parameters were tested: 0.05 learning rate, 0.001 learning rate, 0.005 learning rate and 0.0001 learning rate.

Figure 28. LSTM strategy learning rate sensitivity analysis

Note: Sensitivity analysis of learning rate in LSTM strategy. The following changes in parameters were tested: 0.05 learning rate, 0.001 learning rate. Default model is using rolling 252 days training window. Hyperparameters used: number of units in hidden layers: 30, length of single input: 15, activation function for hidden layer: ReLU, loss function: Mean Squared Error, optimizer: Adam, epochs: 100, dropout rate: 0.2, starting learning rate: 0.01.

6.2.6 Activation function

Next tested parameter was activation function. Figure 29 and Table 21 presents results using this changed parameter. Three options are obtaining similar results with worst performing tanh activation function. LSTM might be robust to changes in activation function. The reason for this may be that LSTM model already have many activation functions in single unit, thus adding one more might not have huge impact on optimization process.

Table 21. Performance statistics of LSTM strategy activation function sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
LSTM	5.14	0.27	19.20	50.00	15.87	7.72	11.76	0.10	0.32	447	566
LSTM tanh activation	1.66	0.09	19.02	55.23	15.36	7.31	11.79	0.03	0.11	1065	534
LSTM elu activation	5.43	0.29	18.73	52.11	14.85	6.29	9.11	0.10	0.37	1159	542
LSTM selu activation	1.56	0.08	19.39	57.39	17.35	7.75	14.98	0.03	0.09	886	424
LSTM sigmoid activation	6.74	0.35	19.45	52.23	14.24	6.18	8.95	0.13	0.47	691	311

Note: Performance statistics of sensitivity analysis of activation function in LSTM strategy. The following changes in parameters were tested: tanh activation, elu activation, selu activation, sigmoid activation.

Figure 29. LSTM strategy activation function sensitivity analysis

Note: Sensitivity analysis of activation function in LSTM strategy. The following changes in parameters were tested: tanh activation, elu activation. Default model is using rolling 252 days training window. Hyperparameters used: number of units in hidden layers: 30, length of single input: 15, activation function for hidden layer: ReLU, loss function: Mean Squared Error, optimizer: Adam, epochs: 100, dropout rate: 0.2, starting learning rate: 0.01.

6.2.7 Optimizer

Next tested parameter was optimizer. Figure 30 and Table 22 present results using this changed parameter. Two tested options gave almost the same results, but utterly different than default 'ADAM' optimizer. Optimization process is crucial in terms of achieving accurate results. Change in important parameter like optimizer can affect results in every way. Thus LSTM model is not robust to optimizer change.

Table 22. Performance statistics of LSTM strategy optimizer sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
LSTM	5.14	0.27	19.20	50.00	15.87	7.72	11.76	0.10	0.32	447	566
LSTM sgd optimizer	-1.26	-0.07	18.02	72.11	16.17	11.14	24.26	-0.02	-0.08	922	1363
LSTM adadelta optimizer	-0.62	-0.03	17.54	65.53	16.71	11.13	21.38	-0.01	-0.04	936	1307

Note: Performance statistics of sensitivity analysis of activation function in LSTM strategy. The following changes in parameters were tested: sgd optimizer, adadelta optimizer.

Figure 30. LSTM strategy optimizer sensitivity analysis

Note: Sensitivity analysis of activation function in LSTM strategy. The following changes in parameters were tested: sgd optimizer, adadelta optimizer. Default model is using rolling 252 days training window. Hyperparameters used: number of units in hidden layers: 30, length of single input: 15, activation function for hidden layer: ReLU, loss function: Mean Squared Error, optimizer: Adam, epochs: 100, dropout rate: 0.2, starting learning rate: 0.01.

6.2.8 Loss function

Next tested parameter was loss function. Figure 31 and Table 23 presents results using this changed parameter. There were considered two additional loss function ‘mae’ and ‘logcosh’. Performances of this two options are significantly worse than default ‘mse’ loss function. Similarly to optimizer, loss functions are important in optimization process and changing them will change the result of optimization. Thus, LSTM model is not robust to changes in loss function.

Table 23. Performance statistics of LSTM strategy loss function sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARC AMD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
LSTM	5.14	0.27	19.20	50.00	15.87	7.72	11.76	0.10	0.32	447	566
LSTM MAE loss function	0.67	0.03	19.87	66.57	16.93	17.54	39.28	0.01	0.04	119	21
LSTM logcosh loss function	-1.62	-0.08	19.64	79.05	18.33	11.14	31.70	-0.02	-0.09	166	286

Note: Performance statistics of sensitivity analysis of loss function in LSTM strategy. The following changes in parameters were tested: mae loss, logcosh loss.

Figure 31. LSTM strategy loss function sensitivity analysis

Note: Sensitivity analysis of loss function in LSTM strategy. The following changes in parameters were tested: mae loss, logcosh loss. Default model is using rolling 252 days training window. Hyperparameters used: number of units in hidden layers: 30, length of single input: 15, activation function for hidden layer: ReLU, loss function: Mean Squared Error, optimizer: Adam, epochs: 100, dropout rate: 0.2, starting learning rate: 0.01

6.2.9 The number of units

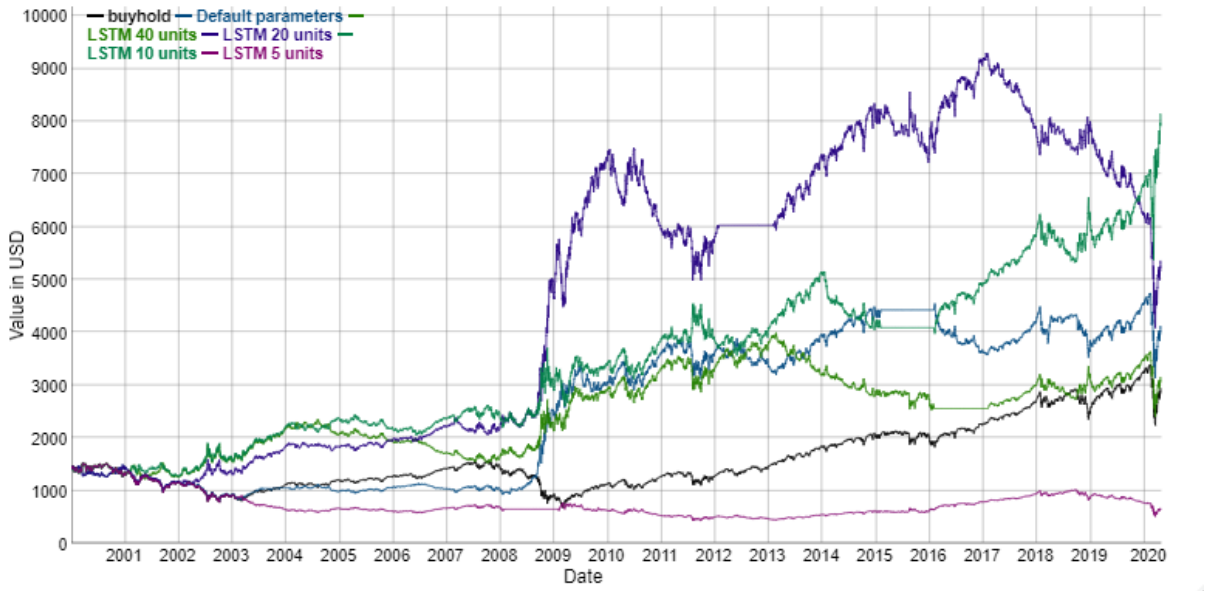
Next tested parameter was number of units. Figure 32 and Table 24 present results using this changed parameter. Three tested options performed comparably, the best performing setting was 10 units. Results show that smaller number of units could improve the final outcome. Testing more options will give more information about stability of this parameter. LSTM model was not robust to performed changes in this parameter.

Table 24. Performance statistics LSTM strategy number of units sensitivity analysis

Name	ARC	IR	aSD	MD	AMD	MLD	All Risk	ARC MD	ARCA MD	Trades	Stop Signal
BuyHold	3.23	0.16	19.95	64.33	17.34	7.155	15.92	0.05	0.18	1	0
LSTM	5.14	0.27	19.20	50.00	15.87	7.72	11.76	0.10	0.32	447	566
LSTM 40 units	3.82	0.19	19.70	44.78	15.59	7.18	9.87	0.08	0.24	174	312
LSTM 20 units	6.44	0.33	19.26	58.32	15.10	3.74	6.34	0.11	0.43	395	538
LSTM 10 units	8.53	0.44	19.41	31.97	13.39	3.12	2.59	0.27	0.64	149	302
LSTM 5 units	-3.85	-0.22	17.58	76.51	16.92	20.07	45.67	-0.05	-0.23	74	261

Note: Performance statistics of sensitivity analysis of units number in LSTM strategy. The following changes in parameters were tested: 40 units, 20 units.

Figure 32. LSTM strategy number of units sensitivity analysis



Note: Sensitivity analysis of units number in LSTM strategy. The following changes in parameters were tested: 40 units, 20 units. Default model is using rolling 252 days training window. Hyperparameters used: number of units in hidden layers: 30, length of single input: 15, activation function for hidden layer: ReLU, loss function: Mean Squared Error, optimizer: Adam, epochs: 100, dropout rate: 0.2, starting learning rate: 0.01.

ARIMA strategy was not robust to changes in parameters and obtain the best results for option with fixed starting point for training window. Moving average crossover strategy was robust to changes in parameters and obtain the best results for option with 126 trading window. Contrarian strategy was robust to changes in parameters and obtain the best results for option with 252 testing days. Momentum strategy was robust to changes in parameters and obtain the best results for option with 126 trading window. Volatility breakout strategy was robust to changes in parameters and obtain the best results for option with default parameters. Macro factor strategy was robust to changes in parameters and obtain the best results for option with 21 trading days. LSTM strategy was not robust for changes in parameters and obtain the best results for option with 10 units.

7. Conclusions

Automated trading systems gained its high popularity after computerization era, when most of the people have access to fast computing machines. What is more automated trading aim was to delete human errors from trading and risk management. Nowadays market analysts use various methods for generating signals including both those which were developed in middle of XX century as well as state of the art models.

This research shows the process of creating automated trading system based on S&P500 data for the period of last 20 years. Implemented strategies used various algorithms, including

classical methods like moving average crossover, trading breakouts and macroeconomic factors. There were also implemented statistical and machine learning approaches like ARIMA models and LSTM model. LSTM was commonly used in natural language processing, but this research tried to apply it to financial time series, due to long term dependencies and schemas which occur in this type of data. Additionally, this paper presented an approach for dynamic optimization of parameters during backtesting process by using rolling training-testing window. Every method was tested in terms of robustness to changes in parameters and evaluated by performance statistics e.g. Information Ratio, Maximum Drawdown.

This study used Buy & Hold strategy as a benchmark. Most of used strategies were not able to achieve better results than benchmark, however research introduced the method based on combining signals from different strategies to diversify risk of wrong prediction by single strategy. This method allowed to outperform benchmark doubling its compounded returns on the same level of risk. Lastly, the sensitivity analysis shown that rolling training-testing window with dynamic optimization of parameters made performance robust for changes of any parameters chosen at the beginning of this study. LSTM model results (having many hyperparameters selected without statistical optimization) were extremely volatile to changes made for crucial parameters.

At the beginning this research stated four hypotheses, which were verified during the empirical part. Main hypothesis stated that market is inefficient and it is possible to obtain abnormal returns. As it was presented the combination of strategy outperformed market significantly and sensitivity analysis showed it can be done consecutively. Second hypothesis about efficiency of signal combination also cannot be rejected, because combination of signals gave the best results by diversifying the risk of single strategy mistake. Sensitivity analysis did not allow to reject the third hypothesis as LSTM model is not robust to changes in most of the parameters. Finally, LSTM with selected hyperparameters outperformed ARIMA model as was stated in the fourth hypothesis.

This study can be extended by finding more strategies and combining them into more complex sophisticated model – researching more advanced methods of combining signals than simple weighted will probably improve results. LSTM method is new method in time series analysis and only a few papers discussed this topic. There are many ways which could improve this strategy: looking for optimization of parameters during investment process, searching for personalized loss functions created for investment purposes like maximizing information ratio, put more attention on data preprocessing.

References:

Abiad, M. A., 2003. *Early warning systems: A survey and a regime-switching approach*, IMF Working Papers 03/32, International Monetary Fund,

Boehmer, E., Fong, K., Wu, J., 2012, *International Evidence on Algorithmic Trading*. SSRN Electronic Journal,

Box, G. E., & Jenkins, G. M., 1970, *Time series analysis: Forecasting and control Holden-Day*. San Francisco, p. 498,

Brock, W., Lakonishok, J., Lebaron, B., 1992, *Simple Technical Trading Rules and the Stochastic Properties of Stock Returns*, Journal of Finance, 47, 5, pp. 1731-64,

Chappell, D., Padmore, J., Mistry, P., Ellis, C., 1996, *A threshold model for the French franc/Deutschmark exchange rate*, Journal of Forecasting, 15, pp. 155–164,

Chen, K., Zhou, Y., Dai, F., 2015, *A LSTM-based method for stock returns prediction: A case study of China stock market*. Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015, art. no. 7364089, pp. 2823-2824

Chlebus M., DyczkoM., Woźniak M., 2020, Nvidia's stock returns prediction using machine learning techniques for time series forecasting problem, Working Papers of Faculty of Economic Sciences, University of Warsaw, WP 22/2020 (328), https://www.wne.uw.edu.pl/files/6415/9481/5844/WNE_WP328.pdf

Chiu, K., Xu, L., 2003, *Finite Mixture of ARMA-GARCH Model for Stock Price Prediction*, Proc. Of 3rd International Workshop on Computational Intelligence in Economics and Finance (CIEF2003), North Carolina, USA, pp. 1112-1119,

Clements, M., Smith, J., 1999, *A Monte Carlo study of the forecasting performance of empirical SETAR models*, Journal of Applied Econometrics, 14, 2, pp. 123-141,

Contreras, J., Espinola, R., Nogales, F., Conejo, A., 2002, *ARIMA Models to Predict Next-Day Electricity Prices*. Power Engineering Review, IEEE, 22, p. 57,

Devi B., Sundar D., Alli P., 2013, *An Effective Time Series Analysis for Stock Trend Prediction Using ARIMA Model for Nifty Midcap-50*, International Journal of Data Mining & Knowledge Management Process (IJDKP), 3, 1, pp. 65-78,

Eichengreen, B., Rose, A. K., Wyplosz, C., 1996, *Contagious currency crises*, NBER Working Papers 5681, National Bureau of Economic Research,

Frankel, J., Saravelos, G., 2012, *Can leading indicators assess country vulnerability? Evidence from the 2008–09 global financial crisis*, Journal of International Economics, 87, 2, pp. 216-231,

Gerlow, M., Irwin, S., Zulauf, C., Tinker, J., 1990, *A Performance Comparison of a Technical Trading System with ARIMA and VAR Models for Soybean Complex Prices*, Ohio State University. Department of Agricultural Economics and Rural Sociology, Economics and Sociology Occasional Paper, 1790,

Gunasekarage, A., Power, D. M., 2001, *The profitability of moving average trading rules in South Asian stock markets*. Emerging Markets Review, 2, 1, pp. 17-33.,

Hawkins, J., Klau, M., 2000, *Measuring potential vulnerabilities in emerging market economies*, Bank for International Settlements, BIS Working Papers,

Hendershott, T., Jones, C., Menkveld, A., 2011 , *Does Algorithmic Trading Improve Liquidity?*, The Journal of Finance, 66, pp. 1-33,

Holmberg, U., Lönnbark, C., Lundström, C., 2013, *Assessing the profitability of intraday opening range breakout strategies*, Finance Research Letters, 10, 1, pp. 27–33,

Irwin, S. H., Uhrig, J. W., 1984. *Technical analysis–a search for the holy grail.* " Proceedings of the NCR-134 Conference on Applied Commodity Price Analysis, Forecasting, and Market Risk Management.,

Jabłecki J., Kokoszczyński R., Sakowski P., Ślepaczuk R., Wójcik P., Volatility as an Asset Class. Obvious Benefits and Hidden Risks, Peter Lang, Frankfurt am Main, 2015. DOI: 10.3726/978-3-653-04787-5

James, F., 1968, *Monthly Moving Averages - An Effective Investment Tool?*, Journal of Financial and Quantitative Analysis, 3, 3, pp. 315-326,

Kolen, J. F., Kremer, S. C., 2001, *A field guide to dynamical recurrent networks*. John Wiley & Sons,

Kość K., Sakowski P., Ślepaczuk R., 2019, Momentum and Contrarian Effects on the Cryptocurrency Market, Physica A 523, pp. 691-701, https://www.sciencedirect.com/science/article/pii/S037843711930216X?dgcid=author

Kraeger, H. Kugler, P., 1993, *Non-linearities in Foreign Exchange Market: A Different Perspective*, Journal of International Money and Finance, 12, pp. 195-208,

McQueen G., Roley V., 1990, *Stock prices, news, and business conditions*, NBER Working Papers, No. 3520,

Miffre, J., Rallis, G., 2007, *Momentum strategies in commodity futures markets.* Journal of Banking & Finance, 31, 6, pp. 1863-1886.,

Olson, D., 2004, *Have trading rule profits in the currency markets declined over time?* Journal of banking & Finance, 28, 1, pp. 85-105.,

Park, C. H., Irwin, S. H., 2007, *What do we know about the profitability of technical analysis?*, Journal of Economic Surveys, 21, 4, pp. 786-826,

Peel, D. A., Speight, A. E. H., 1994, *Testing for non-linear dependence in inter-war exchange rates*. Weltwirtschaftliches Archiv, 130, pp. 391–417,

Roondiwala, M., Patel, H., Varma, S., 2017, *Predicting stock prices using LSTM*, International Journal of Science and Research (IJSR), 6, 4, pp. 1754-1756,

Ryś P., Ślepaczuk R., 2018, Machine learning in algorithmic trading strategy optimization – design and time efficiency, Central European Economic Journal, 5(1), pp. 206-229, https://content.sciendo.com/view/journals/ceej/5/52/article-p206.xml

Sakowski P., Ślepaczuk R., Wywiał M., 2016, Can we invest based on equity risk premia and risk factors from multi-factor models?, Economics and Business Review 2(16), No. 3, pp. 78-98, http://www.ebr.edu.pl/pub/2016_3_78.pdf

Sang, C., Di Pierro, M., 2018, *Improving Trading Technical Analysis with TensorFlow Long Short-Term Memory (LSTM) Neural Network*, The Journal of Finance and Data Science, doi:10.1016/j.jfds.2018.10.003

Schiereck, D., De Bondt, W., Weber, M., 1999, *Contrarian and momentum strategies in Germany*. Financial Analysts Journal, 55, 6, pp. 104-116.,

Ślepaczuk R., Sakowski P., Zakrzewski G., 2018, *Investment strategies beating the market. What can we squeeze from the market?*, eFinanse Vol.14, no. 4, pp. 36-55, https://e-finanse.com/current-issue/?number=59&id=421

Virtanen, I., Yli-Olli, P., 1987, *Forecasting stock market prices in a thin security market*. Omega, 15, 2, pp. 145–155,

Zhang X., Liang X., Zhiyuli A., Zhang S., Xu R., Wu B.,2019, IOP Conference Series Materials Science and Engineering, 569, 5,

Zenkova M., Ślepaczuk R., 2018, Robustness of Support Vector Machines in Algorithmic Trading on Cryptocurrency Market, Central European Economic Journal, 5(1), pp. 186-205, https://content.sciendo.com/view/journals/ceej/5/52/article-p186.xml

Zoega, G., 2010, *The Financial Crisis: Joblessness and Investmentlessness*, Capitalism and Society, 5, 2.

University of Warsaw
Faculty of Economic Sciences
44/50 Długa St.
00-241 Warsaw
www.wne.uw.edu.pl