Corrected Triple Correction Method, CNN and transfer learning for prediction the realized volatility of Bitcoin and E-mini S&P500

Manevich Vyacheslav[1 [0000−0002−8706−1820]] and Ignatov Dmitriy[2 [0000−0002−6584−8534]]

Russia, Moscow, National Research University Higher School of Economics (HSE University); Postal address: 11 Pokrovsky Boulevard, Moscow, Russia, 109028.
https://www.hse.ru

Abstract. The article compares ARMA models, boosting, neural network models, HAR_RV models and proposes a new method for predicting one day ahead realized volatility of financial series. HAR_RV models are taken as compared classical volatility prediction models. In addition, the phenomenon of transfer learning for boosting and neural network models is investigated. Bitcoin and E-mini S&P 500 are chosen as examples. The realized volatility is calculated based on intraday (intraday - 24 hours) data. The calculation is based on the closing values of the internal five-minute intervals. Comparisons are made both within and between the two intervals. The intervals considered are 01.01.2018 - 01.01.2022 and 01.01.2018 - 02.04.2023. Since there were structural changes in the markets during these intervals, the models are estimated in sliding windows of 399 days length. For each time series, we compare three-parameter enumeration boostings, about 10 different neural network architectures, ARMA models, the newly proposed CTCM method, and various training transfer and training sample expansion options. It is shown that ARMA and HAR_RV models are generally inferior to other listed methods and models. The CTCM model and neural networks of CNN architecture are the most suitable for financial time series forecasting and show the best results. Although transfer learning shows no improvement in terms of forecast precision and yields little decline, it requires more extensive and detailed study. The smallest MAPEs for Bitcoin and E-mini S&P 500 realized volatility forecasts are achieved by the newly proposed CTCM model and are 21.075%, 25.311% on the first interval and 21.996%, 26.549% on the second interval, respectively.

Keywords: Time Series · Forecasting · Bitcoin · E-mini S&P 500 · Cryptocurrency · Realized Volatility · Corrected Triple Correction Method · Transfer Learning · Convolution Neural Network.

1 Introduction

This article is the first step of a large study to identify the best prediction models and/or to create a unified methodology for selecting the most appropriate

models without a long trying process. The main contribution of the work in the investigated field is the new proposed method (TCM and CTCM), which has shown itself to be comparable in forecast precision with neural network models and boosting, the study of the transfer learning effect for financial time series in different configurations on two different assets as well as the systematization of other methods and approaches, which are briefly described in Section 3.5.

The paper examines the logarithms of the realized volatilities of Bitcoin and the S&P 500 futures. Of course, a study of just two assets does not allow us to draw fully representative conclusions about the best or worst models, which can be unambiguously extended to many other time series. However, the purpose of this article is not to choose the best model, but to consider the applicability of various types of models on the example of two major representatives of the cryptocurrency and classic stock exchanges. The conclusions drawn in this study on these two assets are an important starting point for the following work. Namely, that it is impossible to offer a "panacea" in methods. Sometimes even the most classical models like AR(1) can outperform neural network models in prediction precision.

In [13] it was confirmed the presence of volatility spillover between the two mentioned assets, which are among the main representatives of cryptocurrencies and classic stock exchanges. Neural network models, among others, allow us to exploit this connectivity in training. For example, by using two time series simultaneously training or by applying variations of transfer learning.

The article compares a large number of models – classical (ARIMA, HAR, regression) and machine learning (boosting), different neural network architectures, approaches to learning – transfer learning, feature generation for sample expansion, and also proposes new models TCM and CTCM, which is described in section 3.9. The best models are described in more detail in the methodology and their uncertainties are given in the results, other methods and models are briefly described in Section 3.5. All presented methods and models are chosen for consideration as they are classical in the field of econometrics, machine learning or neural networks. Transfer learning are considered because they are poorly understood for economic time-series forecasting and, as we know from other machine learning problems, they can sometimes improve the results.

The MAPE (Mean absolute percentage error) metric is used to compare the quality of predictive power of models. It is explicitly interpretable and allows us to estimate specific errors in percentages. This metric also makes it possible to estimate the possible practical applicability of models and to compare prediction precision on different data series and for different periods.

2 Related Work

As shown by researchers ([5]; [20]; [17]) high volatility of futures prices creates big obstacles for volatility forecasting, in the case of using oil and electricity futures volatilities.

[9] proposed a hybrid ANN-GARCH model to predict Euro/Dollar and Yen/Dollar volatility. The authors considered a small number of layers and neurons for the ANN part and concludes about the best type of architecture. The hybrid approach was shown to be efficient and reduce errors.

Two years later, [10] proposed the application of the developed ANN-GARCH model to Bitcoin volatility. Unfortunately, the authors didn't provide the results obtained for the hybrid MAPE model, but the relative errors for different types of GARCH models are large, which is also consistent with our results.

[6] investigated the applicability of LSTM models in forecasting using the S&P 500 returns from 1992 to 2015 as an example. The authors showed the effectiveness of LSTM relative to RNN and random forest (in some cases).

[8] used hybrid models to predict stock price volatility. They combined a machine learning model, namely the Long-Term Short-Term Memory (LSTM) model with some of the GARCH models. The proposed methodology improved the prediction performance compared to the GARCH models.

[15] compared RNN (recurrent neural network) models, GARCH and EWMA on Bitcoin data. The authors show that RNNs are better in average prediction efficiency, but they don't capture Bitcoin outliers well enough.

[18] used the Dow Jones Industrial Average (DJIA) and Nasdaq Composite (IXIC) to investigate a large number of hybrid models based on ARMAX, GARCH and LSTM models, and the applicability of wavelet transformations. The results show an overall improvement in stock index prediction using the AWT-LSTM-ARMAX-FIEGARCH model with the Student's t distribution. A robust test proves that this model has better prediction precision at different time horizons (1-, 10-, 15-, 20-, 30-, and 60-day ahead) for both stock indices. Also, AWT-LSTM improves the ability of the HAR_RV (3) X-RV model in predicting realized stock volatility for the specified time horizons.

[16] used data from the Chinese oil futures market and the Australian electricity futures market to find that the precision of XGBoost volatility forecasting is significantly superior to GARCH-jump and HAR-jump models in both markets.

[11] used LSTM and CNN-LSTM models to predict gold price volatility. The study showed improved prediction results when using a combination of CNN and LSTM layers.

[12] applied a model based on LSTM layers and ensembles to account for the features of the cryptocurrencies under study, Bitcoin, Etherium and Ripple. The proposed model can leverage mixed cryptocurrency data, reduce overshoot and provide lower computing cost compared to a traditional deep neural network in terms of fewer weights (and lower computation time).

[19] combined different methods and proposed a hybrid DWT-ARIMA-GSXGB model to achieve improved prediction precision, approximation and generalization abilities. First, a discrete wavelet transform is applied to split the data set into approximation and error parts. Then the ARIMA (0, 1, 1), ARIMA (1, 1, 0), ARIMA (2, 1, 1) and ARIMA (3, 1, 0) models process approximate par-

tial data, and the advanced xgboost (GSXGB) model process erroneous partial data. Finally, the prediction results are combined using wavelet reconstruction.

[2] use a new set of traits by creating a six-function set (High, Low, Volume, Open, HiLo, OpSe) rather than the traditional four-function set (High, Low, Volume, Open). The study is conducted on 4 assets: Apple, ExxonMobil, Tesla, Snapchat. MLP, GRU, LSTM, Bi-LSTM, CNN and CNN-LSTM are compared to predict the adjusted closing price of the stock. The authors found that the LSTM model gave the most precise results, although all models showed comparative results in which no one model performed consistently better. It is also important to note that the addition of the new series had an overall positive effect on the performance of the forecasting models. This work shows the applicability of expanding the number of features to improve the predictive power of the models, but this approach does not always give the best results.

[14] investigated the effect of transfer learning on the predictive ability of models and their convergence rate during training on different data and 4 architectures. The transfer was performed both within a single application domain (seismology) and between different application domains (seismology, speech, medicine, and finance). The conclusion from the work is that transfer learning tends to either enhance or have no negative effect on the predictive efficiency of the model.

Thus, a new TCM/CTCM method is given in this paper, which is different from all the given methods, as well as showing better precision than the other considered methods.

3 Data

3.1 Data

The study considers two periods: 01.01.2018 - 01.01.2022 and 01.01.2018 - 02.04.2023. All data is taken from finam.ru[1]. Values at the close of internal five-minute intervals are used. For the forecast we consider an interval of 399 days in a sliding window of length 5. In other words, every fifth value is predicted. This approach makes it possible to reduce the running time of the program and to obtain error estimates.

Exchange instruments are used for analysis:

1. Bitcoin (btc) is the cryptocurrency with the largest capitalization, we can say that it is the "main" cryptocurrency at the moment. It is traded 24/7.
2. The E-mini S&P500 Index (snp) is a futures contract traded on the Chicago Mercantile Exchange (CME) representing one fifth of the value of a standard S&P 500 index futures contract. The S&P500 Index includes: 400 industrial corporations, 20 transportation corporations, 40 financial corporations and 40 utilities. Almost all the companies are traded on the NYSE, but there are also those traded on the AMEX and Nasdaq. The base asset for this futures

[1] https://www.finam.ru/profile/cryptocurrencies/btc-usd/export

is the value of the S&P 500 stock index. This futures is valuable to research as it is related to one of the largest S&P 500 indices and trades almost all day. It is traded from 6:00 Sunday to 5:00 Friday (Chicago Stock Exchange time) with a daily break from 5:00 to 6:00.

3.2 Realized volatility

The five-minute realized volatility is used as the realized volatility. The choice of such time intervals is due to the study of [3]. They showed that the values of the realized volatility calculated by five-minute intervals are the most optimal in terms of precision and microstructure errors. Previously, the same form of volatility calculation was used in the articles of [13] and [1].

The realized volatility on day t is presented as:

$$RV_{t,j} = \sqrt{\sum_{j=1}^{N} r_{t,j}^2}, \tag{1}$$

where $r_{t,j} = log(p_{t,j}) - log(p_{t,j-1})$ – the yield, $p_{t,j}$ – the price of the asset on day t at the end of the intraday interval j of length T (seconds), $j = 1 \dots N$ with the total number of intervals for one day equal to N. The first five minutes are 00:00 – 00:05. The last five minutes are 23:55 – 00:00. Calculation is done by closing prices of five-minute periods.

The RV described above is the square root of the realized variance. In HAR_RV models, realized variance is predicted and then the square roots are taken from them to obtain realized volatilities.

In the article, to obtain comparable results in the presence of gaps in the data, the realized volatility is calculated as follows:

- If there are less than 5 hours of data in the day, the day is deleted;
- If observations are missing at the beginning and/or at the end of the day, the realized volatility is calculated using the available K five-minute intervals, and then scaled to the daily data. Since the day contains 288 five-minute intervals, the numerator is 288:

$$RV_t = \sqrt{\frac{288}{K}\sum_{j=1}^{K} r_{t,j}^2}. \tag{1.1}$$

- In the case of missing data within a day, for example, between moments j_1 and j_2, the appropriate sum is replaced by the square of the yield for the missed period.

Thus, the comparability of the daily realized volatility values of the futures on different days with the daily realized volatility values of Bitcoin is obtained.

In [13] it was obtained that the realized volatilities are of lognormal nature and it is the HAR-ln(RV) models that show the best prediction precision. In

this article, it is the logarithms of the realized volatilities that are fed into the input of the models, and then the exponents from the forecasts are taken. Thus, we can take into account the lognormal character of the realized volatilities and obtain the highest-quality forecasts of the studied values.

4 Methodology

4.1 Workflow

A large number of models and approaches were investigated and compared in the article, the tables in the "Results" section present the best of them by MAPE indices (see (2)). Also in section 3.5 we present models which did not show good prediction precision, but which were also built and investigated. The article provides, according to the authors, an exhaustive comparison (in terms of prediction precision) of neural network models and boosting (with/without Transfer Learning) with similar predictions of classical HAR_RV models, as well as with the CTCM (item 3.9) and $ARIMA(p, 0, q)$ models, where $p \in [1, 10], q \in [0, 9]$. All models are trained in a sliding window of 399 days with a sliding step of 5, followed by a 1-step-ahead forecast. Thus, all models make predictions for the same days, using the same data.

For repeatability, numpy.random.seed(7) is used because the neural networks give slightly different results with each training. Each model in each window is trained 10 times (each time a prediction is made), the prediction for each step is averaged, and then MAPE is calculated, which allows to achieve relative repeatability of the results.

Since the realized volatility is predicted, and [1] confirmed the logarithmic nature of the realized volatility, the logarithms of the realized volatility are input into the models, and exponents are taken from the resulting predictions.

After the MAPEs for the neural networks are calculated, the errors are compared with the MAPEs of the selected best HAR-ln(RV) models, conclusions are made about the best models and the efficiency/non-efficiency of the classical models.

4.2 Boosting

This article describes forecasting with the help of the "xgboost" [25] package and the function "XGBRegressor" in Python. A search is carried out according to 3 parameters, $n\_estimators = 100$, 5 elements in each sample:

1. $eta = [0, 1]$ in steps of 0.1;
2. $gamma = [0, 1]$ in steps of 0.1;
3. $max\_depth = [1, 11]$ in steps of 1.

In the following tables, the results of the boosts are named "xgboost_(eta)_(gamma)_(max_depth)", where instead of parameters are their values in the model.

Catboost [22] and adaboost [21] are also considered. Their results are almost identical (and sometimes inferior) to those of xgboost and are not given in the tables below.

4.3 Architectures of neural networks

As a result of the search, it was found that the best performance of the models is achieved with the "*tanh*" activation function – the tables show the results of the models with this activation function.

For all presented and mentioned below neural network models, the loss function of training is the classical MAE. Empirically on the given data sets, it is obtained that with loss="mae" the best error rates are obtained. The batch_size parameter, as well as the batch lengths (lags), are chosen to be 5. For training in each sliding window 200 epochs are taken with control of training by unchanging loss on 10 consecutive epochs, optimizer= "adam".

The article focuses on neural networks with one gate and one or two feature-rows (features) on the gate (in the endnotes "TS_1" and "TS_2", respectively). Many architectures such as LSTM [7], BiLSTM, CNN-LSTM, MLP, RNN, CNN and Encoder-Decoder are investigated. The most efficient of them, in terms of MAPE and running time, were the CNNs. At the same time, the number of layers has little effect on the error, which can be attributed to the features of the data and training, not only to the predictive abilities of the models themselves. The architectures of the two models that showed the best results in MAPE for both data series are described below.

1. First CNN with 1 layer (M_1_CNN):
   filters=4features, kernel_size=lags;
   dense_layer_output: features neurons.
2. Second CNN with 1 layer (M_2_CNN):
   filters=features, kernel_size=lags;
   dense_layer_output: features neurons.

4.4 Transfer Learning

This section deals with Transfer Learning (TL) only for the above two architectures: M_1_CNN and M_2_CNN, as well as for boosting. The following TL variants are investigated, with appropriate labels at the end of the model names:

1. Training on RV Bitcoin and RV E-mini S&P500 forecasting – "tbs";
2. RV E-mini S&P500 Training and RV Bitcoin Forecasting – "tsb";
3. Training on RV Bitcoin and RV E-mini S&P500 followed by forecasting both assets – "tbbss".

4.5 Other methods

A large number of models, architectures, and approaches have been studied in the course of writing this article. Not all of them have shown effectiveness and good

results. The following models and approaches were investigated in the article, but not included among the best, so their results are not presented in tables and graphs:

1. Classic regression, similar to the AR(1) process. Errors are 4-5% higher than in CNN. Running time is many times less than CNN.
2. RNN. MAPE is 4-5% higher than CNN. The running time is 1.5 times more than that of CNN.
3. LSTM. MAPE is 9-10% higher than CNN. The running time is 2-3 times longer run time than the CNN.
4. BiLSTM. MAPE is 9-10% higher than CNN. The running time is 2 times longer run time than the CNN.
5. LSTM – CNN. MAPE is 4-5% higher than CNN. The running time is 2-3 times longer run time than the CNN.
6. BiLSTM – CNN. MAPE is 4-5% higher than CNN. The running time is 2 times longer run time than the CNN.
7. Encoder – decoder based LSTM. The model is architecturally similar to the classical CNN, it is % less precision than CNN. The running time is 1.5-2 times longer run time than the CNN.
8. MLP (2 to 15 layers with 10/16/25/32/64 neurons) – MAPE is 5-6% higher than CNN. The running time is 1.5 times longer run time than the CNN.
9. Library TabNet [23]. Prediction results made with this library with enumeration of different internal parameters are 5-6% inferior to selected CNN models. Running time is comparable to CNN.
10. The methods listed above and below with artificial expansion of the training sample using the tsfresh library [24] are inferior to the corresponding models without artificial expansion of the training sample set by 4-5%. More than 4,000 features were extracted in each sliding window step, and then only 22 relevant ones were selected. The features extracted in each sliding window were fed into the models as well as into "tbbss"/"tssbb" type transfer learning methods. In the case of the SARIMAX model, where the extracted relevant traits are taken as exogenous variables, the MAPE was increased by 4-5%. MAPE increased by 4-5% relative to CNN.
11. Multiple regression with regressors, which are features extracted using tsfresh. This approach is inferior to the classical regression AR(1) by 2-3% in MAPE, and also takes 10-14 times longer, because it takes about 10-14 seconds to extract the features in one sliding window (for a window length of 399 values).
12. Catboost, adaboost, based on decision trees. These types of boosting are usually inferior to xgboost by 1-2%, so the following tables show the models from the xgboost package.

In all cases of neural network architectures, 1-3 layer models with 5, 20, 32, 64 and 128 neurons per layer (in different combinations) and activators "linear", "tanh", "relu" for each layer were considered. The best results were obtained with the "tanh" activator function on all layers.

4.6 Benchmark HAR

The article presents the results of HAR_RV (5, 21) [4] forecasts in the logarithmic specification as a benchmark. Earlier in [1] it is shown that in terms of prediction precision there is not much difference between the simplest HAR_RV (5, 21) and the model selected by enumerating different HAR_RV model specifications. The paper also shows that the realized volatilities are best predicted exactly by HAR_RV models with logarithmic specification. It is important to point out that in these models the variance is predicted and then the square root is extracted from it, since the task is exactly to predict the realized volatility.

Specification HAR_RV (5,21) [4]:

$$RV_{t+1} = \beta_0 + \beta_1 RV_t + \beta_2 \frac{1}{5}\sum_{j=0}^{4} RV_{t-j} + \beta_3 \frac{1}{21}\sum_{j=0}^{20} RV_{t-j}. \tag{2}$$

4.7 The Triple Correction Method

This method was developed for this article and is based on the following ideas:

– The idea of considering a correction for past step deviation (with some weight):

$$E_j = y_j - \hat{y}_j = \varepsilon_j;$$

– Attempt to consider and reflect the internal dependencies of the original data series and take into account the process changes as a difference of consecutive values (taken with some weight), to correct the algorithm:

$$I_j = y_j - y_{j-1};$$

– Attempt to reflect the deviation of the changes in the raw data series from the changes in the series obtained during the algorithm (taken with some weight) to correct the algorithm:

$$EI_j = ((y_j - y_{j-1}) - (\hat{y}_j - \hat{y}_{j-1})).$$

Let us combine all of them:

$$\hat{y}_{t+1} = \alpha_0 y_t + \alpha_1 (y_t - y_{t-1}) + \alpha_2 (y_t - \hat{y}_t) + \alpha_3 ((y_t - y_{t-1}) - (\hat{y}_t - \hat{y}_{t-1})) =$$

$$= \alpha_0 y_t + \alpha_1 I_t + \alpha_2 E_t + \alpha_3 EI_t =$$

$$= \alpha_0 x_{0,t} + \alpha_1 x_{1,t} + \alpha_2 x_{2,t} + \alpha_3 x_{3,t},$$

where $\hat{y}_{t+1}$ – the predicted value at time $t+1$, y_j – the real "historical" data, $\hat{y}_j$ is the predicted data ($j = 0, \ldots, t$), $\hat{y}_{0,1} = y_{0,1}$, α_k – the adaptation parameters, $k = 0, 1, 2, 3$.

The vector of optimal α_i is found by the following algorithm:

$$\sum_{j=0}^{t-1} (y_{j+1} - \hat{y}_{j+1})^2 = \sum_{j=0}^{t-1} (y_{j+1} - \alpha_0 x_{0,j} + \alpha_1 x_{1,j} + \alpha_2 x_{2,j} + \alpha_3 x_{3,j})^2 \to min;$$

$$\sum_{j=0}^{t-1} x_{i,j}(y_{j+1} - \alpha_0 x_{0,j} + \alpha_1 x_{1,j} + \alpha_2 x_{2,j} + \alpha_3 x_{3,j}) = 0, where\ i = 0, \ldots, 3;$$

Matrix form:

$$Y_{t-1}^T Y_{t-1} \alpha = Y_{t-1}^T X_{0,t}$$

$$\alpha = (Y_{t-1}^T Y_{t-1})^{-1} Y_{t-1}^T X_{0,t},$$

where $Y_{t-1} = [X_{0,t-1}, X_{1,t-1}, X_{2,t-1}, X_{3,t-1}]$, and Y_{t-1}^T is the transpose of matrix Y_{t-1}. When solving the system in the first 4 iterations (due to the sparse matrix Y), regularization is applied with the addition of 1 on the main diagonal.

$$X_{i,t-1} = \begin{bmatrix} x_{i,0} \\ x_{i,1} \\ \vdots \\ x_{i,t-1} \end{bmatrix}, X_{0,t} = \begin{bmatrix} x_{0,1} \\ x_{0,2} \\ \vdots \\ x_{0,t} \end{bmatrix}, \alpha = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix}$$

Thus, iteratively, up to the last known value at step t, all 4 coefficients α_i are obtained, which are then substituted into the original equation to obtain the prediction $\hat{y}_{t+1}$.

Thus, an iterative adaptive method with four adaptation parameters is obtained – the "Triple Correction Method" or abbreviated "TCM". This is the form in which it is further used in the implemented algorithms.

We also propose a modification of this method the "Corrected Triple Correction Method" (CTCM). The modification is that at the last iteration the obtained predicted $t+1$ value is multiplied by $(1 + percentile(MPE_t, 80))$, where $percentile(MPE_t, 80)$ is the 80th percentile of all $MPE_j = \frac{RV_j - h_j}{RV_j}$, $j = 0, \ldots, t$. The 80th percentile is taken from empirical considerations and is in general a parameter of the CTCM model.

As a result, for CTCM the forecast at the moment $t+1$ is calculated by the formula:

$$\hat{y}_{t+1} = (\alpha_0 x_{0,t} + \alpha_1 x_{1,t} + \alpha_2 x_{2,t} + \alpha_3 x_{3,t}) \dot{(} 1 + percentile(MPE_t, 80))$$

Thus, the correction allows to partially account for the percentage error made in the previous fitting steps and improve the prediction result. It is important to note that uncertainties for the adjustment are counted with the sign. Both positive and negative MPE values should be taken into account.

4.8 Methodology for model comparison

Errors are calculated for two time intervals 01.01.2018 - 01.01.2022 and 01.01.2018 - 02.04.2023. Thus, we compare the predictive ability of the models not only in sliding windows, but also in intervals with fundamentally different patterns of time series behavior.

The error metrics used is:

$$MAPE = \frac{100\%}{K}\sum_{j=1}^{K}\frac{|RV_j - h_j|}{RV_j}, \tag{3}$$

$$MAE = \frac{1}{K}\sum_{j=1}^{K}|RV_j - h_j|, \tag{4}$$

$$MSE = \frac{1}{K}\sum_{j=1}^{K}(RV_j - h_j)^2, \tag{5}$$

where h_j – forecast at time j, RV_j – value of realized volatility on day j. In this article, all forecasts are made 1 step ahead 211 times, i.e. $K = 211$.

The target error metric is MAPE, as it allows us to compare models in terms of practical applicability. Also this error is more indicative at different intervals under condition of jumps and outliers of predicted series.

5 Results

The following tables and graphs show forecasting errors of RV Bitcoin and RV E-mini S&P-500 by different methods from 01.01.2018 to 01.01.2022 (column 2022) and from 01.01.2018 to 02.04.2023 (column 2023). Errors are ordered by increasing MAPE in column 2022. Boostings and ARMA are presented in two specifications, as one specification is selected as the best by MAPE and the other by MAE.

No transfer learning is applied to the ARIMA models. Also, the results for both processes showed that reducing the length of the windows has a negative effect on the prediction results. Thus, the sliding window length of 399 values in this article is the best.

5.1 Bitcoin

Tab. 1 shows the errors of Bitcoin RV forecasts. The CTCM model was the best in both periods, outperforming neural networks, ARMA, and boosting. Among neural networks CNN with one input and one output was the best, yielding slightly to CTCM. The errors of the MAPE forecasts, which also include 2022 and the first quarter, are larger by about 1% for all of the above methods. The year 2022 and early 2023 for RV Bitcoin are characterized by significant global shocks, which reduces the predictability of the time series. Moreover, all models have lower MAE and MSE over a wider time horizon in contrast to MAPE. It is also interesting that the best models for one metric are not the best models for the other. However, the target metric in this paper is MAPE, so the models are ordered by this error. Thus, the best MAPE model for both time intervals was CTCM with an error of 21.075% to 2022 and 21.996% to Q1 2023.

	2022			2023		
	MAPE	MAE	MSE	MAPE	MAE	MSE
CTCM	**21.075**	0.01	0.00044	**21.996**	0.009	0.00033
M_2_CNN_TS_1	21.399	0.01	0.00045	22.792	0.009	0.00033
M_1_CNN_TS_1	21.57	0.01	0.00044	22.728	0.009	0.00033
M_1_CNN_TS_2	22.071	0.01	0.00043	23.164	0.009	0.00032
M_2_CNN_TS_2	22.206	0.01	0.00044	23.206	0.009	0.00033
ARMA (7, 9)	23.853	0.01	0.00044	25.276	0.010	0.00033
ARMA (1, 1)	24.014	0.01	0.00042	24.825	0.009	**0.00031**
boost_0.2_0.4_2	24.353	0.01	0.00044	25.568	0.010	0.00034
HAR_RV (5, 21)	25.372	0.011	0.0005	26.33	0.01	0.00037
boost_0.6_0.3_5	25.419	0.010	**0.00039**	27.128	0.010	0.00032
TCM	25.588	0.011	0.00043	25.739	0.01	0.00032

Table 1: Model errors for RV Bitcoin.

Fig. 1 shows graphs of predictions on delayed intervals of realized Bitcoin volatility, the HAR_RV benchmark model, the CNN single-layer model, CTCM, ARMA and boostings, which showed the best results among all ARMAs and boostings. As can be seen from the graphs, the HAR_RV model actually flattens the original data series. The CTCM improves the TCM prediction, but, in sum, slightly underestimates the peaks relative to the TCM. For example, this can be observed in May-June 2021. Boosting and ARMA overestimate relative to quiet periods and underestimate peaks. Neural network models show similar behavior to CTCM, which is logical given the proximity of the errors. The interval, which includes 2022 and Q1 2023, is characterized by many fluctuations that are not well predicted, which increases MAPE for all methods.

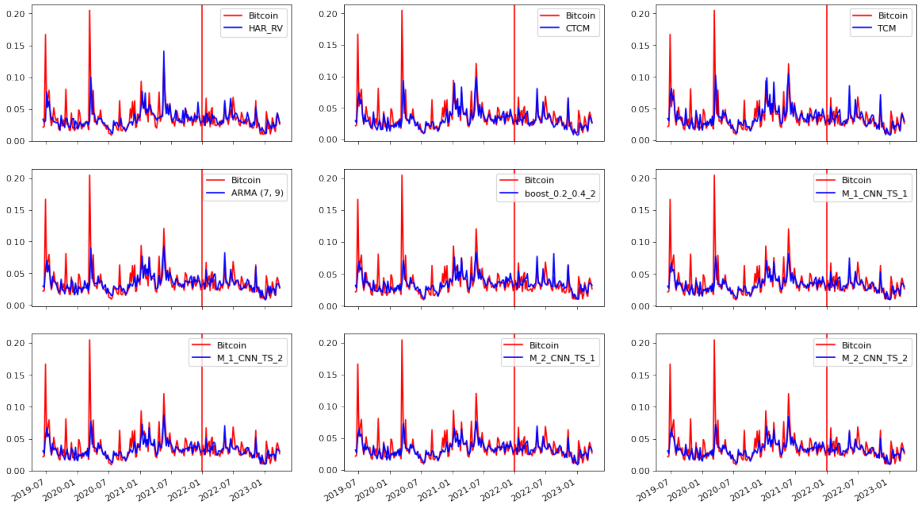

Fig. 1: The red line is Bitcoin RV, the others is predictions of the corresponding models.

5.2 E-mini S&P500

For the RV E-mini S&P-500 the best MAPE is obtained by the CTCM model and is 25.311% (Tab. 2). The best MAPE and MAE boosting models match and outperform ARMA and neural network models on the target metric. Also, it is important to note that, like for RV Bitcoin, the MAPE in the second interval is larger than the MAPE in the first. This suggests the difficulty of forecasting in 2022 due to the characteristics of global economic and geopolitics. At the same time, the HAR_RV model is significantly inferior to all of the above models.

	2022			2023		
	MAPE	MAE	MSE	MAPE	MAE	MSE
CTCM	**25.311**	0.003	0.00002	**26.549**	0.003	0.00002
boost_0.2_0.1_8	26.430	0.002	0.00002	28.338	0.003	0.00002
boost_0.2_0.1_8	26.430	0.002	0.00002	28.338	0.003	0.00002
ARMA (5, 5)	26.536	0.003	0.00002	27.001	0.003	0.00002
ARMA (3, 1)	26.761	0.003	0.00003	27.028	0.003	0.00002
TCM	26.936	0.002	0.00002	27.583	0.003	0.00002
M_1_CNN_TS_1	27.108	0.003	0.00003	27.739	0.003	0.00003
M_2_CNN_TS_1	27.513	0.003	0.00003	27.858	0.003	0.00003
M_1_CNN_TS_2	27.527	0.003	0.00002	27.982	0.003	0.00002
M_2_CNN_TS_2	27.971	0.003	0.00003	28.166	0.003	0.00003
HAR_RV (5, 21)	30.52	0.003	0.00003	30.059	0.003	0.00003

Table 2: Model errors for RV E-mini S&P 500.

Fig. 2 shows graphs of forecasts on pending intervals of realized volatility E-mini S&P500, reference model HAR, one-layer model CNN, CTCM, ARMA and boosting, which showed the best results among all ARMA and boosting. As you can see from the charts, the HAR, CTCM and TCM models for the futures behave very similarly to these same models for RV Bitcoin. However, for this series the best ARMA and boostings behave very similarly, unlike neural networks, which in this case tend to underestimate peaks and average small fluctuations. The interval from early 2022 to Q1 2023 is characterized by increasing volatility and fluctuations, which led to an increase in MAPE for all methods except HAR.

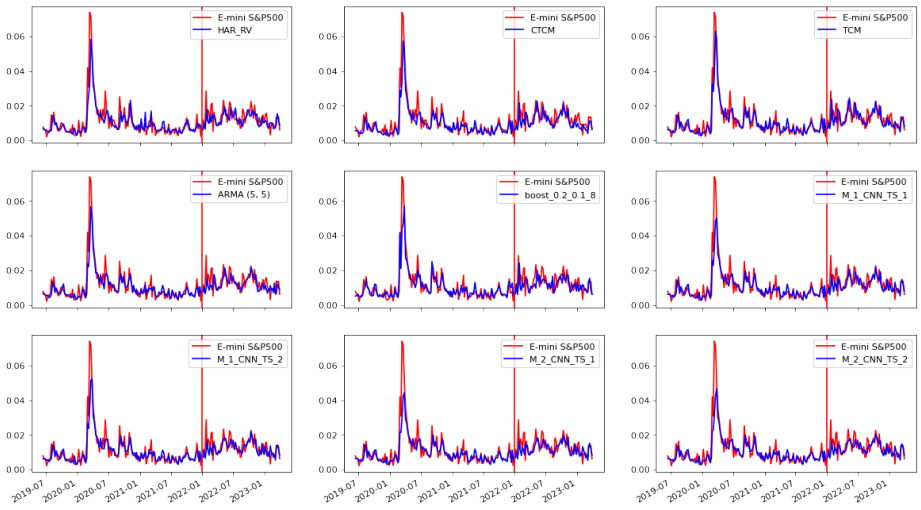

Fig. 2: The red line is RV E-mini S&P 500, the others is the forecasts of the corresponding models.

5.3 Transfer Learning

Tab. 3 and Tab. 4 show the MAPE, MAE, MSE of the corresponding models using TL.

Interesting conclusions for the transfer learning of RV Bitcoin:

1. In the case of RV Bitcoin, the best MAPEs at both time intervals are achieved using tbbss transfer learning for neural networks. These results are inferior to what was obtained without transfer learning.
2. In the case of tsb transfer for RV Bitcoin, we observe, in general, a deterioration of prediction quality relative to methods without transfer.
3. In the case of neural networks and tbbss transfer, the deterioration may be due to the fact that these two assets are poorly suited for cross-transfer learning within the features of the neural network architecture. At the same

time, the tbbss transfer shows an improvement in the predictive ability of the boostings. The transfer learning will be investigated in more detail and extensively in future works.

	2022			2023		
	MAPE	MAE	MSE	MAPE	MAE	MSE
M_1_CNN_TS_1_tbbss	**22.785**	0.01	0.00043	**24.093**	0.009	0.00032
M_2_CNN_TS_1_tbbss	23.060	0.01	0.00043	24.272	0.009	0.00033
boost_0.1_0.9_3_tbbss	23.485	0.010	0.00045	24.170	0.010	0.00033
boost_0.2_0.6_8_tbbss	23.784	0.010	**0.00040**	24.968	0.009	**0.00031**
M_1_CNN_TS_1_tsb	25.121	0.011	0.00043	26.302	0.01	0.00033
M_2_CNN_TS_1_tsb	25.341	0.011	0.00045	26.519	0.01	0.00034
boost_0.2_0.5_6_tsb	28.966	0.013	0.00063	32.764	0.014	0.00054
boost_0.8_0.4_7_tsb	29.742	0.013	0.00063	34.590	0.014	0.00054

Table 3: Model TL errors for RV Bitcoin.

Interesting conclusions for the transfer learning of RV E-mini S&P500:

1. The best MAPE error for the RV E-mini S&P500 is obtained by a single layer neural network model with tbbss-type transfer learning and is 26.456%.
2. neural network models show less MAPE than boostings.
3. tbbss-type transfers for both neural networks and boostings show higher precision on the target metric than the tbs transfer.
4. Tbs transitions for boostings show significant degradation of prediction precision for all metrics. This may be due to the fact that Bitcoin is a much more volatile asset. It may also indicate some peculiarities and limitations of transfer learning for boostings, which will be investigated in further work.

	2022			2023		
	MAPE	MAE	MSE	MAPE	MAE	MSE
M_1_CNN_TS_1_tbbss	**26.456**	0.003	0.00003	27.042	0.003	0.00003
M_2_CNN_TS_1_tbbss	26.725	0.003	0.00004	**26.998**	0.003	0.00003
M_1_CNN_TS_1_tbs	26.914	0.003	0.00004	29.225	0.004	0.00004
M_2_CNN_TS_1_tbs	27.68	0.003	0.00005	30.279	0.004	0.00004
boost_0.2_0.2_10_tbbss	28.658	0.003	**0.00002**	30.628	0.003	0.00003
boost_0.7_0.4_4_tbbss	29.708	0.003	**0.00002**	31.084	0.003	**0.00002**
boost_0.9_0.7_1_tbs	93.682	0.006	0.00006	121.266	0.011	0.00029

Table 4: Model TL errors for RV E-mini S&P 500.

6 Discussion and conclusion

The results of this work can be summarized in two main theses:

1. The proposed CTCM model showed results outperforming neural networks, boostings, HAR_RV and ARMA models both with and without transfer learning. This model requires a more detailed study not a large number of assets, which will be done in future works. Also, these methods are easy to handle and have low run time.
2. The paper also clearly shows the difference in the forecast accuracy, and, accordingly, in the behavior of the studied time series. As we know, the second period is very saturated with geopolitical and economic events, which significantly affected both Bitcoin and S&P500. Thus, it is important to take into account when learning (or transferring) not only the proximity of the processes themselves, but also the proximity of events and shocks that occurred to the assets at certain points in time.

All this means that one way to improve forecasting results is not only to develop a new model specification, but also to take into account in it various shocks related to the external environment and other assets.

7 Future work

In future works it is planned to investigate the developed TCM/CTMC methods in more detail on a larger number of assets, as well as the phenomenon of learning transfer. TCM and CTMC have shown results that outperform other methods under consideration, which may indicate their practical applicability and quality. This will be tested on a much larger number of assets.

The transfers, although they did not improve the precision of forecasts for neural networks, still improved the precision of boostings, which may indicate their potential applicability with a proper selection of assets and methods. Special attention will be paid to the method of selecting the assets from which the transfer is performed, as this should largely affect the quality of the forecast.

Acknowledgements

The article was prepared within the framework of the HSE University Basic Research Program.

References

1. Aganin A.D., Manevich V.A., Peresetsky A.A., Pogorelova P.V. Comparison of Cryptocurrency and Stock Market Volatility Forecast Models. HSE Economic Journal. 2023; 27(1): 49–77. (In Russ.). DOI: 10.17323/1813-8691-2023-27-1-49-77.
2. Alkhatib K., Huthaifa K. Khazaleh, Hamzah Ali Alkhazaleh, Anas Ratib Alsoud, Laith A. (2022). A New Stock Price Forecasting Method Using Active Deep Learning Approach. Journal of Open Innovation: Technology, Market, and Complexity, Volume 8, Issue 2. DOI: 10.3390/joitmc8020096.

3. Bollerslev T. (1986). Generalized autoregressive conditional heteroskedasticity. Journal of Econometrics, 31, 307–327. DOI: 10.1016/0304-4076(86)90063-1.
4. Corsi F. (2003). A Simple Long Memory Model of Realized Volatility. Manuscript, University of Southern Switzerland.
5. Dutta A., Bouri E., Saeed T. (2021). News-based equity market uncertainty and crude oil volatility. Energy, 222, Article 119930. DOI: 10.1016/j.energy.2021.119930.
6. Fischer T., & Krauss C. (2018). Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research, 270(2), 654–669. DOI: 10.1016/j.ejor.2017.11.054.
7. Hochreiter S., & Schmidhuber J. (1997). Long Short-Term Memory. Neural Computation; 9 (8): 1735–1780. DOI:10.1162/neco.1997.9.8.1735
8. Kim H. Y., & Won C. H. (2018). Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models. Expert Systems with Applications, 103, 25–37. DOI: 10.1016/j.eswa.2018.03.002.
9. Kristjanpoller W., & Minutolo, M. C. (2016). Forecasting volatility of oil price using anartificial neural network-garch model. Expert Systems with Applications, 65, 233–241. DOI: 10.1016/j.eswa.2016.08.045
10. Kristjanpoller, W., & Minutolo, M. C. (2018). A hybrid volatility forecasting framework integrating garch, artificial neural network, technical analysis and principal components analysis. Expert Systems with Applications, 109, 1–11. DOI: 10.1016/j.eswa.2018.05.011
11. Livieris I. E., Pintelas E., Pintelas P. (2020). A CNN–LSTM model for gold price time-series forecasting. Neural Computing and Applications. DOI: 10.1007/s00521-020-04867-x.
12. Livieris I. E., Kiriakidou N., Stavroyiannis S., Pintelas P. (2021). An Advanced CNN-LSTM Model for Cryptocurrency Forecasting. Electronics, 10(3), 287. DOI: 10.3390/electronics10030287.
13. Manevich V. A., Peresetsky A. A., Pogorelova P. V. (2022). Stock market and cryptocurrency market volatility. Applied Econometrics. 65, 65–76. DOI: 10.22394/1993-7601-2022-65-65-76. (in Russian).
14. Otovic E., Njirjak M., Jozinovic D., Mausa G., Michelini A., Stajduhar I. (2022). Intra-domain and cross-domain transfer learning for time series data—How transferable are the features? Knowledge-Based Systems, Volume 239. DOI: 10.1016/j.knosys.2021.107976.
15. Shen Z., Wan Q. and Leatham D. J. (2021). Bitcoin return volatility forecasting: A comparative study between GARCH and RNN. Journal of Risk and Financial Management, 14(7), 337. DOI:10.3390/jrfm14070337.
16. Shusheng D., Tianxiang C., Yongmin Z. (2022). Futures volatility forecasting based on big data analytics with incorporating an order imbalance effect. International Review of Financial AnalysisVolume 83, October 2022, 102255. DOI: 10.1016/j.irfa.2022.102255.
17. Singhal, D., & Swarup, K. (2011). Electricity price forecasting using artificial neural networks. International Journal of Electrical Power & Energy Systems, 33(3), 550–555. DOI: 10.1016/j.ijepes.2010.12.009.
18. Zolfaghari, M., & Gholami, S. (2021). A hybrid approach of adaptive wavelet transform, long short-term memory and ARIMA-GARCH family models for the stock index prediction. Expert Systems with Applications, 182, 115149. DOI: 10.1016/j.eswa.2021.115149.
19. Wang Y. and Guo Y. (2020). Forecasting method of stock market volatility in time seriesdata based on mixed model of arima and xgboost. China Communications, 17(3),205–221. DOI: 10.23919/JCC.2020.03.017

20. Wang, L., Ma, F., Hao, J., & Gao, X. (2021). Forecasting crude oil volatility with geopolitical risk: Do time-varying switching probabilities play a role? International Review of Financial Analysis, 76, Article 101756. DOI: 10.1016/j.irfa.2021.101756
21. Adaboost
22. Catboost
23. TabNet
24. Tsfresh
25. Xgboost