Ether Price Prediction Using Deep Learning Methods
(Final Report)

Prepared by:
Toomaj Foroud

Supervised by:
Kevin Wittek
Sebastian Posth

Institute for Internet Security (ifis)
Dec. 2021

Abstract

Cryptocurrencies are innovative payment systems that operate mostly based on decentralized digital ledgers using blockchain technology. As they're getting more applications in financial transactions, more people and financial firms urge to invest in this market. However, due to high volatility and sharp price fluctuations of these assets, developing some trustable models to forecast the market behavior is critical for portfolio management and optimized trading. Large number of unpredictable and uncertain factors makes financial analysis of cryptocurrency, one of the most interesting and challenging prediction problems.

In this study the main focus is the application of deep learning methods such as Multi-Layer Perceptron (MLP), Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) in univariate and multivariate time series analysis in order to predict the Ether price, the cryptocurrency built on the open source network of Ethereum. Finding effective factors among a vast set of features such as network technical information, macro-economic and financial indicators and even trends of social media, and the way they are correlated to the price movements will be the most important issues in this study.

This report is organized as follows. At first, a literature review of this subject and then the problem definition is presented. In this part a general framework to tackle this problem is suggested that each part of this framework can be a subject of individual researches. Then, a statistical analysis of the Ether price time series and subsequently a univariate modelling using Sarima, MLP, CNN and LSTM has been performed that showed promising results. The next step was multivariate analysis of this market by considering a wide range of different variables. After, pre-processing and feature selection through the dataset, deep learning models including MLP, CNN and LSTM with different structures and transformation methods were applied to forecast the Ether price in two time horizons namely '1-day' and '1-week'. The statistical analysis of the outcomes of this step helped us to select the most successful combination of models and transformations for both prediction intervals.

The final part of this research belongs to the algorithms and methods which are applied to improve the accuracy of the prediction models. These methods are hyper-parameter tuning, stacking and filtering. The results for '1-day' prediction show that CNN with log transformation is the most accurate model with MAPE equals to 12.16±4.99 and the value of 335.34±70.63 for RMSE. Stacking could not obtain better results than CNN, but they were superior to the other models. While In case of '1-week' prediction, generally the accuracy is lower than '1-day' interval, multi-headed CNN and multi-headed MLP were most successful models with MAPE around 20%. Stacking was successful to improve slightly the accuracy of models. In This step filtering methods such as 'rolling mean', 'simple exponential smoothing' and 'kalman-filter' were applied to smooth the predictions. They could enhance the performance of models around 25% and decrease MAPE to a number around 15%. Generally, it can be stated that application of deep learning models as a hybrid with stacking and filtering methods can be considered as a promising solution in case of price prediction of cryptocurrency or any other similar assets.

Table of contents:

Introduction

Currently, cryptocurrencies are emerging as widely accepted alternative currency transfer methods. They are getting the ability to perform most financial transactions and therefore trading in this market is considered by many people as one of the most profitable investments. However, one of the main characteristics of this growing financial market is highly volatility and sharp price fluctuations, as a result developing an accurate and trustable model to forecast the market behavior is critical for portfolio management and optimized trading. During recent years, forecasting of cryptocurrency market is generally seen as one of the most interesting and also challenging prediction problems due to the huge number of unpredictable and uncertain factors involved and the significant volatility of their prices, leading in intricate time-based dependencies [1] [2] [3].

In this research we are going to investigate Ether market, the cryptocurrency built on the open source network of Ethereum, with the second highest market capitalization and one of the most promising cryptocurrencies. The main aim is to obtain a comprehensive model which is able to explain the current market movements and to forecast the future behaviors of Ether price. Clearly the expected methodology can be extended for other similar assets as well.
One of the main innovations of this study can be the adoption of cryptocurrency technical information, economic and financial indicators and even the effects of social media on the price mechanism [4] [5]. Answering the questions like "which cryptocurrencies are more correlated" and "which features have greater impact in price prediction" is of great importance in this proposed study.

Literature Review

The predictability of bitcoin price in short time horizons was discussed in [6]. Different machine learning methods have been employed and the results showed that recurrent neural networks and gradient boosting classifiers were relatively promising in this study. This research has considered a wide range of feature sets including technical, blockchain-based, sentiment-/interest-based and asset based variables. An interesting point mentioned in this paper is the bitcoin market efficiency which is a term defining the relation between market information and prices. It has been suggested that bitcoin market efficiency is increasing over the years and it means that the market prices reflect more information about the actual value of the assets and hence the price will be more a function of fundamentals that can lead to a more stable and competitive market. The various factors investigated in this research are as bellow:
Technical features: describing features that are related to historical bitcoin market data (e.g., bitcoin returns)
Blockchain-based features: denoting features related to the bitcoin blockchain (e.g., number of bitcoin transactions).
Sentiment-/interest-based features: describing features that are related to sentiment and internet search volume of bitcoin (e.g., bitcoin Twitter sentiment).
Asset-based features: are features that are related to financial markets other than the bitcoin market (e.g., gold returns, returns of the MSCI World index).

In [7] the interdependency between bitcoin and altcoin prices has been studied. Daily data of 17 cryptocurrencies and two altcoin price indices from 2013 to 2016 were used in this investigation. The results revealed the dependency between altcoin and bitcoin prices, while this relation was observed to be stronger in short-run. However, the authors couldn't find any rational connection between interdependency and the similarity of price formation mechanism. Moreover, it was concluded that in a long-run, macro-financial parameters are more effective than bitcoin price in altcoin price formation.
Besides the coin prices the exogenous factors that were considered to be effective in price formation in this research are:

- Investment and transaction demand of the currency
- Information search costs
- Transaction execution/validation mechanism
- Macroeconomic and financial developments e.g. oil and gold prices, NASDAQ composite, 10-Year Treasury Constant Maturity Rate and exchange rates of USD/EUR and CNY/USD

In the literature, only a few scholars employ all the available feature categories [8] [9]. Moreover, the significance of each feature class in different prediction models has not investigated broadly. Only in a few researches different prediction time horizons (minutely, hourly, daily …) has taken into account [10] and most studies are applying the models in just daily intervals for price forecasting.
A part of literature has examined the predictability of bitcoin prices based on various parameters like social media attention [11] [12] and bitcoin-related historical and technical indicators [13]. One group of scholars considered the period from 2014 to 2018, by counting the number of tweeted ''Bitcoin''. The results revealed that the number of tweets on Twitter can influence the next day trading volume of bitcoin [14]. Moreover, [15] studied the effect of user's comments in online platforms on price variations and number of transaction of cryptocurrencies and found a correlation between bitcoin price and the number of positive comments on social media. They reported an accuracy of 79% using Granger causality test, which suggests the importance of user opinions to the price fluctuations.
A comparison among the performance of feedforward NNs, support vector machines and LSTM networks to forecast bitcoin price directions has been done by [16]. They considered blockchain-based features and price histories from 2013 to 2019. In this study SVM was the one with the highest accuracy in shorter prediction intervals. On the other hand, LSTM was the best predictor in long term intervals. In another study, [17] tried to predict Ether price based on time series consisting of daily closing prices of this cryptocurrency. The main machine learning methods they applied were linear regression (LR) and support vector machine (SVM). By harvesting the outputs of a cross-validation method in training phase, a more robust model was expected to be created. In this setup, SVM outperformed the linear regression model with higher prediction accuracy over test data set.
In a study high-dimensional technical indicators were used to predict daily bitcoin prices applying tree-based prediction models [13]. This paper showed the practicality of technical analysis in such complicated asset market. Various machine learning approaches were examined to predict the direction of bitcoin price trends by [18]. They concluded that simple methods e.g. logistic regressions outperform complex algorithms like RNNs. However, imbalance test and training set make their results unreliable.

In [19] a review of state-of-the-art deep learning methodologies for time series analysis has been presented. Algorithms are classified as discriminative, generative and hybrids. According to the suggested definition, discriminative models also called conditional models are those methods which are depending on the observed data and trying to learn from the given statistics. The intuition of these models is based on the conditional probability of target Y given an observation X, which means they can be used to discriminate the target given an observation. Methods like SVM, ANNs, CNNs, LSTM are examples of this class of methods. On the other hand, Generative models consider the joint probability distribution of both observation X and target Y. It can be used to generate random instances regarding a set of observation and target (X, Y). Gaussian Process, Bayesian Networks, Restricted Boltzmann Machine, Deep Belief Network and Generative Adversarial Nets are some instances of such algorithms.
In he mentioned paper, a numerical study was performed on four real world and benchmark data sets, applying six different algorithms including RNN, CNN, LSTM, GANs, DBN and AE. It was observed that deep learning-based methods are good at recognizing complex structure of large data sets and distinguishing patterns from multiple time series. While all deep learning methods were relatively successful in numerical tests, LSTM suggested the most promising results. Some drawbacks of deep learning methods observed in this research, were the weak explicable out-puts and high computational times due to the determination of large amounts of weights in multiple layers [19]. While in this literature review a large number of real world applications were presented, there was no case of cryptocurrency analysis.
A multi-input deep neural network method for prediction of cryptocurrency price and trends was proposed in [20]. The model gets different cryptocurrency prices (Bitcoin, Ether and XRP) as input and uses them independently to extract appropriate information from each asset separately. While the suggested method didn't show a significant improvement in price prediction (regression problem), it was relatively superior in market directional movement prediction (classification problem).
Although innovative deep learning models are able to be applied in highly nonlinear time-series problems, they can be inefficient and erratic in cryptocurrency forecasts. More specifically, [21] [22] [23] explained that the main sources of this deficiency are: First, cryptocurrencies time-series are very similar to random walk process, which means the high complexity of the prediction, and second is autocorrelation in the errors and the lack of stationarity that leads to inefficiency of deep learning. In this regard, [1] introduced a solution to enforce the time series to become "suitable" for training a deep learning model by considering the stationarity properties and a framework to develop more reliable prediction models. Their research was mainly focused on using transformations in order to make time series stationary and examining the autocorrelation in the errors.

Problem definition

Regarding the literature of this field of study, it can be stated that price prediction of cryptocurrencies is very similar to stock market forecasting with additional complexities due to its immaturity, fast development and quite complicated technology behind it. Therefore, it seems that this problem can be treated by modifying and upgrading the traditional methods of stock market analysis. However, because

of very fast data generation and volatility in the field of cryptocurrencies, besides method revision, the automation must also be considered which forces us to think about employing machine learning models.
Generally, there are two approaches in stock market analysis called “technical” and “fundamental” methods. Technical analysis try to predict the price of a share (stock) based on the balance of demand and supply powers. In this method, the previous statistics of the share is used to predict the future changes of the price. In this way, it is supposed that the price trend depend mainly on the attitude, psychology and emotion of the traders. The key tools in technical analysis are prices, time intervals, transaction volumes and the width of changes across the industry or market. This data will interpreted by developing some representations like charts, moving averages, oscillators and indicators.
On the other hand, fundamental analysis focuses on the basic factors that affect the interest of the economy, industry and company. It aims to measure the actual value of an asset by taking into account the economy, financial and other factors to perceive the opportunities where the prices are experiencing fundamental changes. This approach mainly examines the features that can change the value of a security namely macroeconomic factor, company related factors, financial statement, competition and business concepts. In fact, it analyses the economy, the related industry, business environment and the firm itself. Hence, the main tools to be worked with in this way include financial analysis, economic conditions, future profitability and industry analysis.
By considering these two approaches, the relatively ambiguous nature of the newly emerged market of cryptocurrencies and the previous studies, the following framework in Figure 1 can be suggested to tackle the problem of cryptocurrency price prediction.

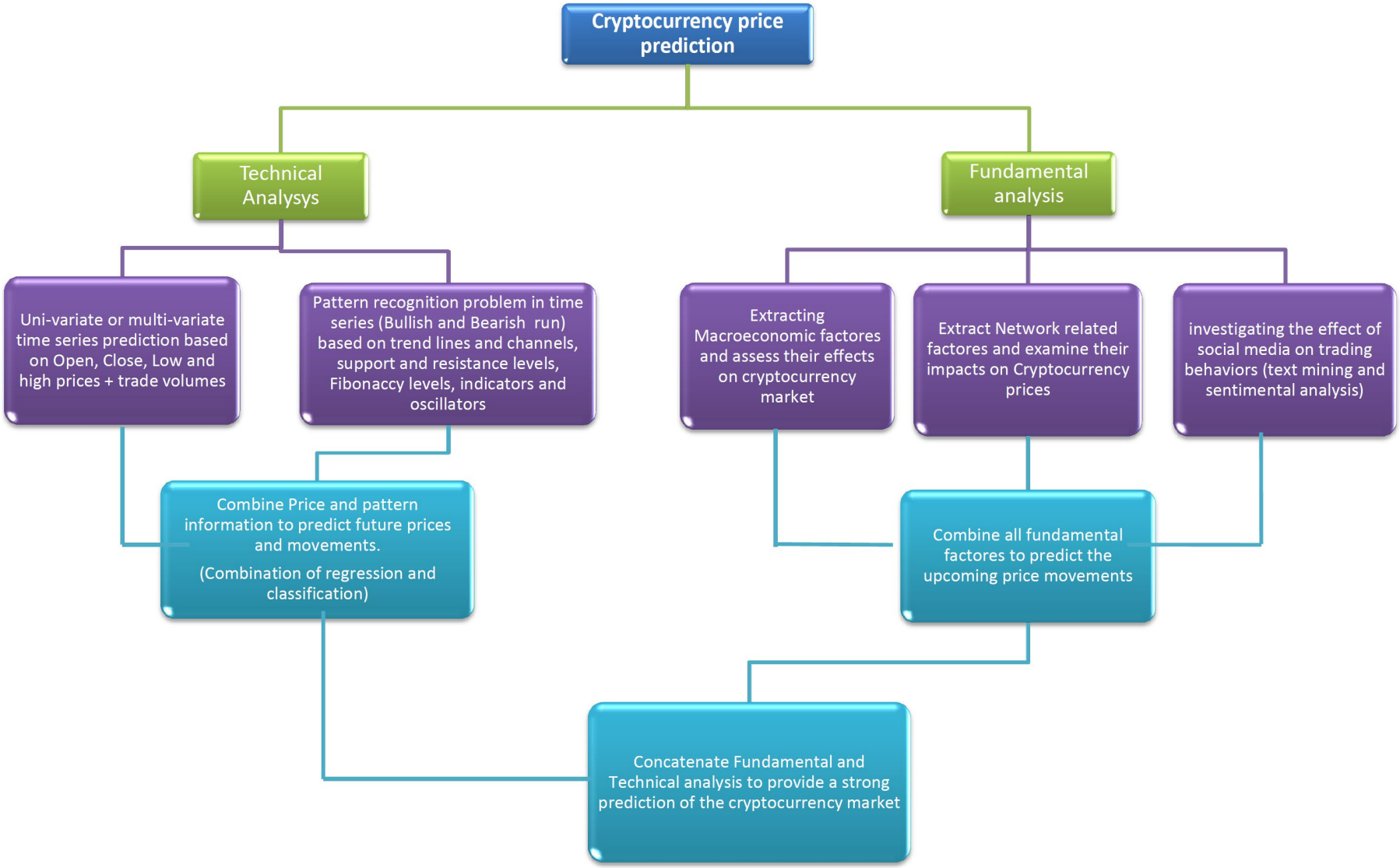

Figure 1- Suggested Framework for Ether price prediction problem

Univariate Time Series analysis of Ether Price

To make a sense about the price data we are dealing with, the information of daily, hourly and minutely price changes (including Open, Close, High & Low prices) of Ether from 2017-08-17 to 2021-04-29 were taken from www.cryptodatadownload.com. The information of the opening daily prices are summarized in Figures 2 &3.

Figure 2- Ether daily opening prices (Binance Exchange)

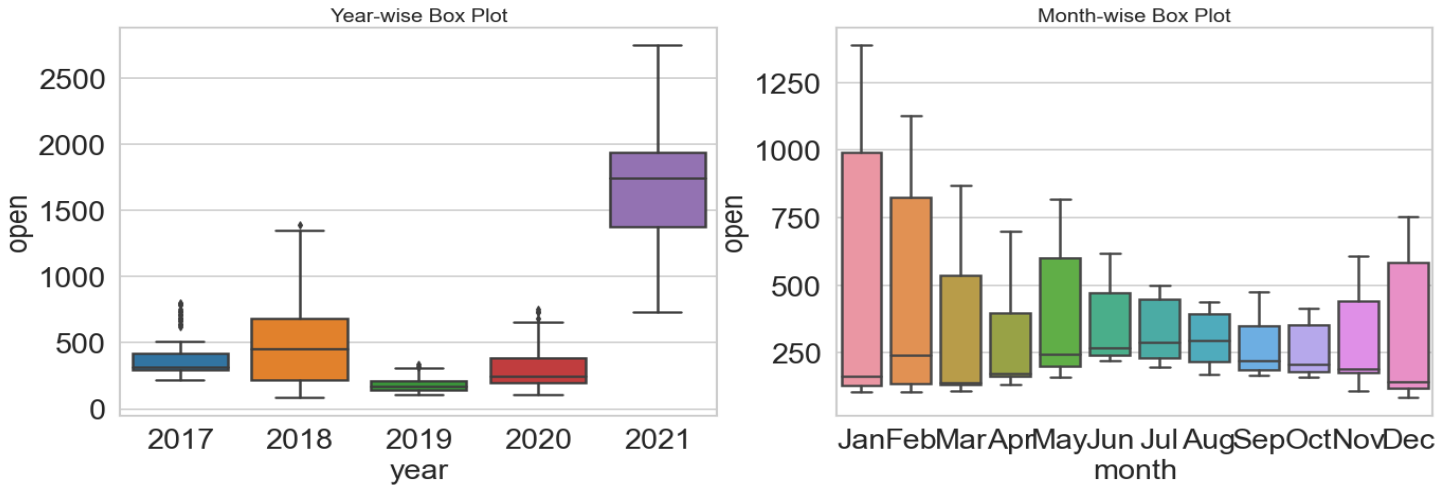

Figure 3- Box Plot of monthly and yearly price changes of Ether

As it can be observed in these graphs, the price of Ether is very volatile and there is not any obvious trend in the price movements. To check the stationarity of the data we've performed ADF and KPSS stationarity test that led to these out-puts presented in Table 1.

Table1- ADF and KPSS stationarity tests

	ADF	KPSS
Statistic	3.094	1.060
p-value	1.0	0.010
Critical values (1%)	-3.435	0.739
Critical values (5%)	-2.864	0.463
Critical values (10%)	-2.568	0.347

The results confirm the non-stationarity of this time series. Therefore, the application of conventional methods like ARIMA (Auto Regression Integrated Moving Average) can be problematic. Therefore some remedies like transformation and differencing must be applied to provide a stationary time series. In this way the effect of differencing and SARIMA modelling are investigated and the results are summarized in the following table and Figures 4-6.

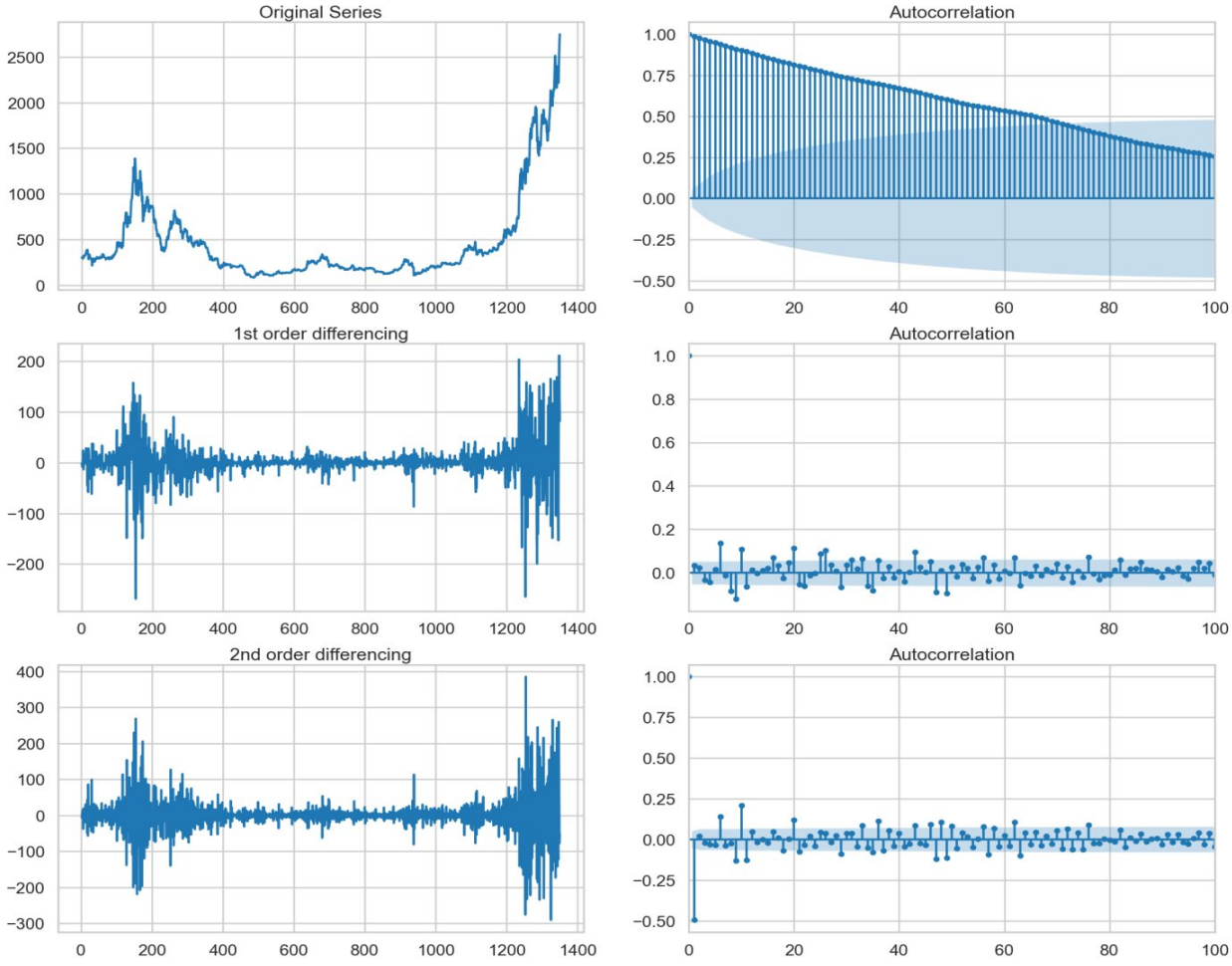

Figure 4- 1st and 2nd differencing of Ether time series and Its Autocorrelations

SARIMAX Results

Dep. Variable:	y	No. Observations:	1250
Model:	ARIMA(5, 4, 5)	Log Likelihood	-5998.844
Date:	Fri, 04 Jun 2021	AIC	12019.689
Time:	22:33:42	BIC	12076.094
Sample:	0	HQIC	12040.897
	- 1250		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-3.1420	0.019	-161.655	0.000	-3.180	-3.104
ar.L2	-4.3985	0.050	-87.913	0.000	-4.497	-4.300
ar.L3	-3.3914	0.061	-55.197	0.000	-3.512	-3.271
ar.L4	-1.4884	0.041	-36.590	0.000	-1.568	-1.409
ar.L5	-0.3531	0.013	-26.995	0.000	-0.379	-0.328
ma.L1	0.5513	0.967	0.570	0.568	-1.343	2.446
ma.L2	-1.5597	1.490	-1.047	0.295	-4.479	1.360
ma.L3	-1.5416	0.087	-17.773	0.000	-1.712	-1.372
ma.L4	0.5603	1.494	0.375	0.708	-2.368	3.489
ma.L5	0.9908	0.953	1.039	0.299	-0.878	2.860
sigma2	888.6705	850.644	1.045	0.296	-778.560	2555.901

Ljung-Box (L1) (Q):	5.26	Jarque-Bera (JB):	9695.40
Prob(Q):	0.02	Prob(JB):	0.00
Heteroskedasticity (H):	0.32	Skew:	-0.44
Prob(H) (two-sided):	0.00	Kurtosis:	16.64

Figure 5- The code output of SARIMA in Python using statmodels

Figure 6- Ether Price prediction using SARIMA (p=5, d=4, q=5)
With RMSE= 237.52

After trying SARIMA model, some deep learning methods have been investigated. MLP, LSTM and CNN are those which are used for preliminary forecasting of Ether prices. One important issue in using machine learning models especially deep learning methods is the sample extraction. After basic data preparation and analysis, it's time to provide sample sets to train and test the models. The process which has been used here to exploit appropriate samples is sliding window. By choosing a specific window length, we pass through the whole time series and divide prices in input and output variables. An example of such sample creation is depicted in Figure 7. After producing samples, considering 6 time steps as input and the seventh one as the output, they've been divided to train and test sets with 90% for training and 10% for testing. Activation function is chosen to be 'relu', the optimization algorithm for fitting process is 'adam' and the cost function is 'mse'. The results are summarized in Table 2 and Figures 8 - 11.

Figure 7- Rolling/sliding windows from original time series [24]

Table 2- Accuracy of Deep Learning methods in Ether price forecasting

Model	Architecture	RMSE
MLP	Dense layer (neuron=100) + Dense layer (neuron=1)	11.76
CNN	Conv1D (filter=64, kernel-size=2) + MaxPooling1D(pool-size=2) + Dense layer (neuron=50) + Dense layer (neuron=1)	39.27
vanilla LSTM	LSTM layer (neuron=50) + Dense layer (neuron=1)	34.69
Stacked LSTM	LSTM layer (neuron=50) + LSTM layer (neuron=50) + Dense layer (neuron=1)	37.85
Bidirectional LSTM	Bidirectional wrapper(LSTM layer (neuron=50)) + Dense layer (neuron=1)	32.38
CNN-LSTM	TimeDistributed wrapper (Conv1D (filter=64, kernel-size=1) + MaxPooling1D(pool-size=2)) + LSTM layer (neuron=50) + Dense layer (neuron=1)	90.76
ConvLSTM	ConvLSTM2D (filter=64, kernel-size=(1,2)) + Dense layer (neuron=1)	26.39
Encoder-Decode (multi-step)	LSTM layer (neuron=100) + RepeatVector + LSTM layer (neuron=100) + TimeDistributed wrapper (Dense layer (neuron=1))	58.98

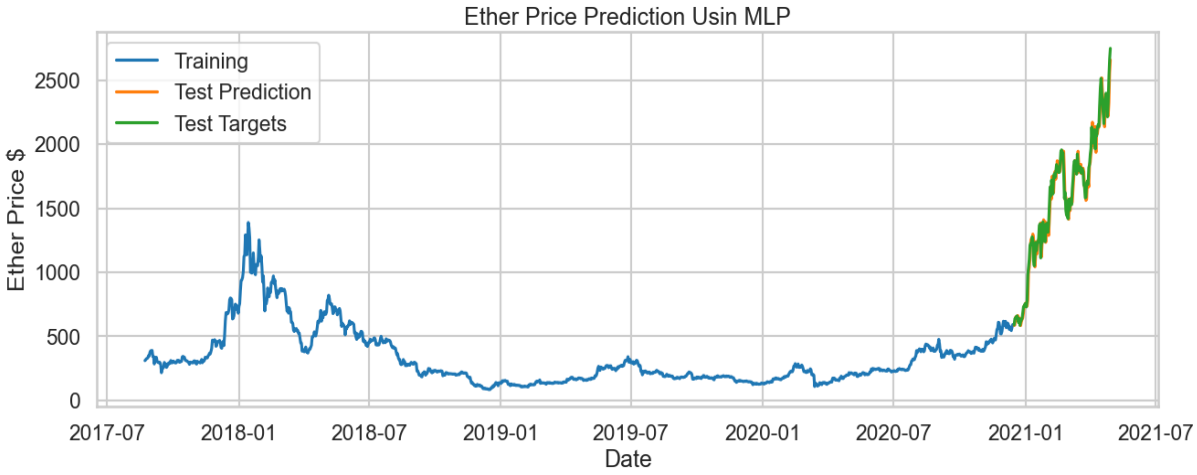

Figure 8- Ether Price prediction using MLP

Figure 9- Ether Price prediction using CNN

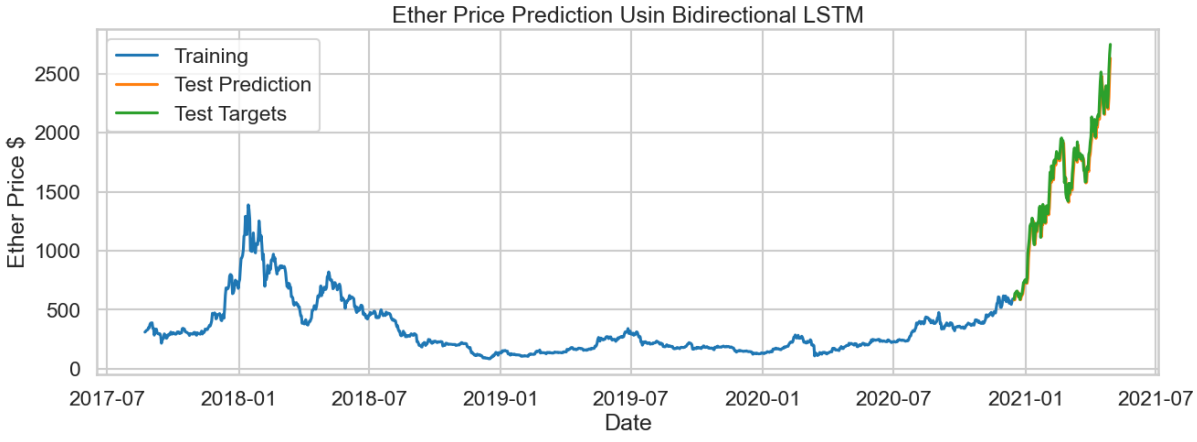

Figure 10- Ether Price prediction using Bidirectional LSTM

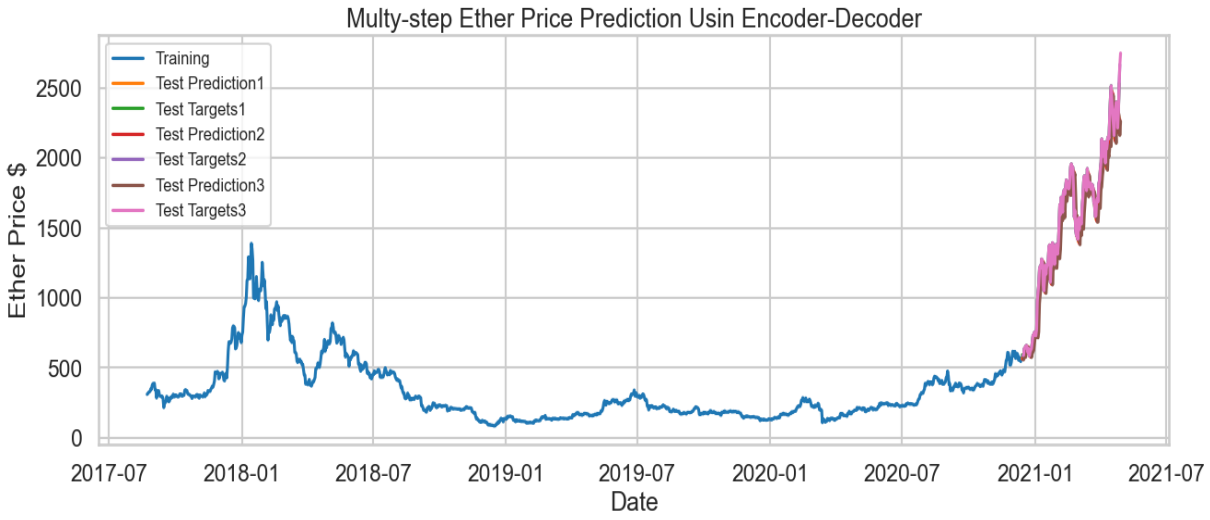

Figure 11- Multi-step Ether Price prediction using Bidirectional Encoder-Decoder

It should be noticed that the results obtained above are based on fixed, simple networks without optimization of hyper-parameters or making complex hybrids. It's worth to try different settings for initial sample selection, network parameters and designing more sophisticated combination of deep learning models to achieve higher accuracy.

Multivariate Time Series analysis of Ether Price

In the first part of this study we provided a brief literature review of the subject and summarized the general framework that can be applied for the problem of "Ether price prediction". First, we considered the problem as a univariate time series forecasting. Several algorithms including model-based ARIMA (Auto Regression Integrated Moving Average) and also some deep learning methods including MLP, LSTM and CNN were employed to predict the Ether price for the one day ahead using the prices of the last seven days. Since promising results were obtained in the first part, it was decided to extend the research for a multivariate time series prediction.

Data collection and feature selection

As it was mentioned before, a broad range of economical, technical and social behavior factors are effective in the market of cryptocurrencies. However, it is not yet completely known what are exactly the whole set of these features and how actually they affect the market movements. Therefore, here we have collected 69 different features that seem to be influential. The complete list and the main source of the data extraction are summarized in Table 3. As it can be observed from Table 3, beside Ether price, 68 features have been collected in order to build an exhaustive model for future Ether price prediction. The definitions of these variables have been provided in Table 4. The next step will be the data pre-processing.

In the pre-processing step, data were merged to make a single data frame in Python pandas library, then cleaned and scaled/normalized. The beginning time of data gathering initially considered as the 1st January 2016, as the main market movements of Ether was started around this date. Feature selection was the last part before trying to build prediction models. Feature selection helps us identifying the most relevant and effective features. The details of preprocessing steps are explained in the following.

First of all we had to handle the missing values. Two sources of missing values in our data set are the time periods before a feature was started to be measured and recorded and the second one are the holidays that financial markets were closed and no data were recorded. The first one is the main problem for cryptocurrencies which have been introduced or listed in Binance Exchange (our main source for cryptocurrency market) after the time of 1st January of 2016. Since a big part of our data set was related to such features and most of their data were collected after 2018, we decided to select the 1st January of 2018 as the beginning point of our time series. In this way a huge portion of missing values could be handled, and for those coins which were still emerged after this time, the value of zero is considered for missing 'prices', 'trade count' and 'volume' until the actual values have been recorded. Moreover, the data related to three coins including Polkadot, Solana and Chainlink were available just from 2019 and 2020, so more than one year of these time series are missed and it's been decided to remove them completely from the data frame.

Table 3- Initial features collected for multi-variate Ether price prediction

NO.	Feature	Category	Data Source	First Available Data	Statistics				
					count	mean	std	min	max
1	ADA_Price(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	4/17/2018	1986	0.123499	0.304613	0	2.2378
2	ADA_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	4/17/2018	1986	1.73E+08	2.78E+08	0	2261842102
3	ADA_Tradecount	CryptoCurrency Market	https://www.cryptodatadownload.com/	4/17/2018	1986	97613.73	308151.9	0	3500994
4	BNB_Price(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	12/18/2017	1986	31.86939	92.66938	0	653.905
5	BNB_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	12/18/2017	1986	1722753	1965826	0	20188858.2
6	BNB_Tradecount	CryptoCurrency Market	https://www.cryptodatadownload.com/	12/18/2017	1986	159561.6	441212.2	0	4491728
7	BTC_Price(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	8/17/2017	1986	9025.75	12463.6	0	63267.27
8	BTC_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	8/17/2017	1986	34590.56	38748.25	0	402201.6738
9	BTC_Tradecount	CryptoCurrency Market	https://www.cryptodatadownload.com/	12/18/2017	1986	445272.2	673786.4	0	6331062
10	DOT_Price(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	8/22/2020	1986	2.572857	8.336922	0	45.862
11	DOT_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	8/22/2020	1986	3034019	9194349	0	85164987.8
12	DOT_Tradecount	CryptoCurrency Market	https://www.cryptodatadownload.com/	8/22/2020	1986	83969.9	277331.4	0	3074880
13	EOS_Price(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	5/28/2018	1986	2.340484	2.625835	0	14.614
14	EOS_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	5/28/2018	1986	8686424	17526353	0	244242568
15	EOS_Tradecount	CryptoCurrency Market	https://www.cryptodatadownload.com/	5/28/2018	1986	72158.42	169056.9	0	2490351
16	ETC_Price(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	6/12/2018	1986	5.518298	11.3794	0	126.8135
17	ETC_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	6/12/2018	1986	1267058	2769666	0	48009367.36
18	ETC_Tradecount	CryptoCurrency	https://www.cryptodatadownload.com/	6/12/2018	1986	42629.03	191727.2	0	5008537

		Market							
19	ETH_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	8/17/2017	1986	374737.3	516271.2	0	4663239.929
20	ETH_Tradecount	CryptoCurrency Market	https://www.cryptodatadownload.com/	12/18/2017	1986	234511.8	460341.8	0	5548968
21	LINK_Price(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	1/16/2019	1986	3.91281	8.465454	0	50.4885
22	LINK_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	1/16/2019	1986	7143951	13296092	0	98883403.15
23	LINK_Tradecount	CryptoCurrency Market	https://www.cryptodatadownload.com/	1/16/2019	1986	135801.2	295021.3	0	2127313
24	LTC_Price(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	12/14/2017	1986	61.97203	71.92098	0	373.33
25	LTC_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	12/14/2017	1986	383378.2	651223	0	6731555.344
26	LTC_Tradecount	CryptoCurrency Market	https://www.cryptodatadownload.com/	12/18/2017	1986	86456.74	186542.8	0	1832583
27	SOL_Price(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	9/14/2020	1986	1.637759	6.660477	0	51.351
28	SOL_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	9/14/2020	1986	1478850	5187416	0	105584282
29	SOL_Tradecount	CryptoCurrency Market	https://www.cryptodatadownload.com/	9/14/2020	1986	31662.51	126578.9	0	1667812
30	TRX_Price(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	6/11/2018	1986	0.016133	0.02302	0	0.16256
31	TRX_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	6/11/2018	1986	6.76E+08	1.16E+09	0	14514073978
32	TRX_Tradecount	CryptoCurrency Market	https://www.cryptodatadownload.com/	6/11/2018	1986	54983.07	145997.7	0	1680016
33	XLM_Price(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	5/31/2018	1986	0.087528	0.126051	0	0.69574
34	XLM_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	5/31/2018	1986	85007467	1.95E+08	0	3305718353
35	XLM_Tradecount	CryptoCurrency Market	https://www.cryptodatadownload.com/	5/31/2018	1986	44529.83	120642.9	0	1450888
36	XRP_Price(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	5/4/2018	1986	0.21843	0.26969	0	1.815375
37	XRP_Volume(USDT)	CryptoCurrency Market	https://www.cryptodatadownload.com/	5/4/2018	1986	1.89E+08	5.04E+08	0	8608358083

38	XRP_Tradecount	CryptoCurrency Market	https://www.cryptodatadownload.com/	5/4/2018	1986	138141.5	379844.4	0	4196624
39	BTC-Dominance	CryptoCurrency Market	https://coinmarketcap.com	1/1/2016	1986	61.59579	14.36816	32.81	91.48
40	AddressCount	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	51261023	46976808	40701	155998485
41	AverageDailyTransactionFee	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	1.724904	5.159378	0	68.72
42	AvgGasPrice	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	3.92E+10	5.49E+10	7.32E+09	9.4E+11
43	BlockCountRewards	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	5948.424	682.4785	2829	6637
44	BlockDifficulty	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	2132.944	1631.806	8.147	8166.911
45	BlockReward	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	20202.62	7225.702	10304.63	36551.875
46	BlockSize	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	19843.26	14096.1	885	62116
47	BlockTime	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	14.57919	2.515794	12.83	30.31
48	DailyActiveERC20Address	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	164622.2	128900.5	0	1376408
49	DailyActiveEthAddress	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	239919.8	170876.6	2351	799580
50	EnsRegistrations	Ethereum Network Statistics	https://etherscan.io/	5/9/2017	1986	138.8434	573.6636	0	14688
51	EtherPrice	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	378.6147	561.7594	0.93	4181.28
52	GasLimit	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	7870504	3068312	500238	14988783
53	GasUsed	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	3.65E+10	2.71E+10	2.92E+08	95577766788
54	MarketCap	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	99853517	11302869	76166284	116118666.4
55	NetworkHash	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	163715.3	129902.4	526.2007	643805.8731
56	NetworkUtilization	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	0.644419	0.350656	0.019	0.9915
57	ERC20 daily token	Ethereum Network	https://etherscan.io/	1/1/2016	1986	344211.5	307073.3	0	1475639

	transfer	Statistics							
58	TransactionFee	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	1.92E+21	4.07E+21	1.57E+19	4.28E+22
59	Ethereum daily transactions	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	585411.8	410647.2	8233	1716600
60	Uncles	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	550.2487	346.9513	149	2096
61	verified-contracts	Ethereum Network Statistics	https://etherscan.io/	1/1/2016	1986	77.23615	74.1175	0	399
62	USD_EURO	Macro Economical Features	https://www.quandl.com/	1/1/2016	1986	0.87776	0.035833	0.800769	0.963855422
63	USD_YUAN	Macro Economical Features	https://www.quandl.com/	1/1/2016	1986	6.741459	0.221432	6.2649	7.1786
64	WGS7YR	Macro Economical Features	https://www.quandl.com/	1/1/2016	1986	1.826289	0.75619	0.39	3.14
65	Gold_Price	Macro Economical Features	https://www.quandl.com/	1/1/2016	1986	1418.944	236.1007	1072.7	2061.5
66	NASDAQ_Ind	Macro Economical Features	https://www.quandl.com/	1/1/2016	1986	177.639	43.87445	106.97	357.42
67	OIL_PRICE	Macro Economical Features	https://www.quandl.com/	1/1/2016	1986	54.31259	13.65289	12.22	84.09
68	GOOGLE_TREND	Social Media Features	https://trends.google.com/	1/1/2016	1986	9.417251	11.55473	1.14	100
69	TWEETS	Social Media Features	https://bitinfocharts.com	3/16/2016	1986	8446.418	8001.057	0	51682

Table 4- Definition of selected features

NO	Feature	Descriptions
1	ADA_Price(USDT)	This is the closing price of the time period for Cardano from BINANCE EXCHANGE DATA
2	ADA_Volume(USDT)	This is the volume in the base/converted in USDT amount for Cardano from BINANCE EXCHANGE DATA
3	ADA_Tradecount	This is the unique number of trades for the given time period for Cardano from BINANCE EXCHANGE DATA
4	BNB_Price(USDT)	This is the closing price of the time period for Binance Coin from BINANCE EXCHANGE DATA
5	BNB_Volume(USDT)	This is the volume in the base/converted in USDT amount for Binance Coin from BINANCE EXCHANGE DATA
6	BNB_Tradecount	This is the unique number of trades for the given time period for Binance Coin from BINANCE EXCHANGE DATA
7	BTC_Price(USDT)	This is the closing price of the time period for Bitcoin from BINANCE EXCHANGE DATA
8	BTC_Volume(USDT)	This is the volume in the base/converted in USDT amount for Bitcoin from BINANCE EXCHANGE DATA
9	BTC_Tradecount	This is the unique number of trades for the given time period for Bitcoin from BINANCE EXCHANGE DATA
10	DOT_Price(USDT)	This is the closing price of the time period for Polkadot from BINANCE EXCHANGE DATA
11	DOT_Volume(USDT)	This is the volume in the base/converted in USDT amount for Polkadot from BINANCE EXCHANGE DATA
12	DOT_Tradecount	This is the unique number of trades for the given time period for Polkadot from BINANCE EXCHANGE DATA
13	EOS_Price(USDT)	This is the closing price of the time period for EOS from BINANCE EXCHANGE DATA
14	EOS_Volume(USDT)	This is the volume in the base/converted in USDT amount for EOS from BINANCE EXCHANGE DATA
15	EOS_Tradecount	This is the unique number of trades for the given time period for EOS from BINANCE EXCHANGE DATA
16	ETC_Price(USDT)	This is the closing price of the time period for Ethereum Classic from BINANCE EXCHANGE DATA
17	ETC_Volume(USDT)	This is the volume in the base/converted in USDT amount for Ethereum Classic from BINANCE EXCHANGE DATA
18	ETC_Tradecount	This is the unique number of trades for the given time period for Ethereum Classic from BINANCE EXCHANGE DATA
19	ETH_Volume(USDT)	This is the volume in the base/converted in USDT amount for Ethereum from BINANCE EXCHANGE DATA
20	ETH_Tradecount	This is the unique number of trades for the given time period for Ethereum from BINANCE EXCHANGE DATA
21	LINK_Price(USDT)	This is the closing price of the time period for Chainlink from BINANCE EXCHANGE DATA
22	LINK_Volume(USDT)	This is the volume in the base/converted in USDT amount for Chainlink from BINANCE EXCHANGE DATA
23	LINK_Tradecount	This is the unique number of trades for the given time period for Chainlink from BINANCE EXCHANGE DATA
24	LTC_Price(USDT)	This is the closing price of the time period for Litecoin from BINANCE EXCHANGE DATA
25	LTC_Volume(USDT)	This is the volume in the base/converted in USDT amount for Litecoin from BINANCE EXCHANGE DATA
26	LTC_Tradecount	This is the unique number of trades for the given time period for Litecoin from BINANCE EXCHANGE DATA
27	SOL_Price(USDT)	This is the closing price of the time period for Solana from BINANCE EXCHANGE DATA

28	SOL_Volume(USDT)	This is the volume in the base/converted in USDT amount for Solana from BINANCE EXCHANGE DATA
29	SOL_Tradecount	This is the unique number of trades for the given time period for Solana from BINANCE EXCHANGE DATA
30	TRX_Price(USDT)	This is the closing price of the time period for Tron from BINANCE EXCHANGE DATA
31	TRX_Volume(USDT)	This is the volume in the base/converted in USDT amount for Tron from BINANCE EXCHANGE DATA
32	TRX_Tradecount	This is the unique number of trades for the given time period for Tron from BINANCE EXCHANGE DATA
33	XLM_Price(USDT)	This is the closing price of the time period for Stellar from BINANCE EXCHANGE DATA
34	XLM_Volume(USDT)	This is the volume in the base/converted in USDT amount for Stellar from BINANCE EXCHANGE DATA
35	XLM_Tradecount	This is the unique number of trades for the given time period for Stellar from BINANCE EXCHANGE DATA
36	XRP_Price(USDT)	This is the closing price of the time period for XRP from BINANCE EXCHANGE DATA
37	XRP_Volume(USDT)	This is the volume in the base/converted in USDT amount for XRP from BINANCE EXCHANGE DATA
38	XRP_Tradecount	This is the unique number of trades for the given time period for XRP from BINANCE EXCHANGE DATA
39	BTC-Dominance	The individual proportions of the Bitcoin relative to the total market capitalization of all assets.
40	AddressCount	The total distinct numbers of address on the Ethereum blockchain and the increase in the number of address daily.
41	AverageDailyTransactionFee	The daily average amount in USD spent per transaction on the Ethereum network.
42	AvgGasPrice	The daily average gas price used of the Ethereum network (Gwei).
43	BlockCountRewards	The historical number of blocks produced daily on the Ethereum network and the total block reward.
44	BlockDifficulty	The mining difficulty and the historical value of the Ethereum network.
45	BlockReward	The combination of total Ether supplied to the Ethereum network with reference to the Ethereum Block Count and Rewards Chart and Ethereum Uncle Count and Rewards Chart.
46	BlockSize	The historical average block size in bytes of the Ethereum blockchain.
47	BlockTime	The historical average time taken in seconds for a block to be included in the Ethereum blockchain.
48	DailyActiveERC20Address	The daily number of unique addresses that were active on the network as a ERC20 token sender or receiver.
49	DailyActiveEthAddress	The daily number of unique addresses that were active on the network as a sender or receiver.
50	EnsRegistrations	The number of daily registration of the Ethereum Name Service.
51	EtherPrice	The daily historical price for Ether in USD.
52	GasLimit	The historical daily average gas limit of the Ethereum network.
53	GasUsed	The historical total daily gas used of the Ethereum network.
54	MarketCap	The historical breakdown of Ether daily market capitalization and average price.
55	NetworkHash	The historical measure of the processing power of the Ethereum network.
56	NetworkUtilization	The average gas used over the gas limit in percentage.
57	ERC20 daily token transfer	The number of ERC20 tokens transferred daily.

58	TransactionFee	Historical total number of Ether paid as transaction fee for the Ethereum network.
59	Ethereum daily transactions	The total number of transactions on the Ethereum blockchain with daily individual breakdown for average difficulty, estimated hash rate, average block time and size, total block and uncle block count and total new address seen.
60	Uncles	The historical number of uncle blocks produced daily on the Ethereum network and the total uncle block reward.
61	verified-contracts	The total number of contracts verified daily.
62	USD_EURO	SPOT EXCHANGE RATE - EURO AREA from US Federal Reserve Data
63	USD_YUAN	CHINA - SPOT EXCHANGE RATE, YUAN/$ from US Federal Reserve Data
64	WGS7YR	7-Year Treasury Constant Maturity Rate
65	Gold_Price	Gold Price according to London Fixings, London Bullion Market Association (LBMA). Fixing levels are set per troy ounce.
66	NASDAQ_Ind	NASDAQ Capital Market Composite (RCMP) Index
67	OIL_PRICE	OPEC Crude Oil Price (USD)
68	GOOGLE_TREND	Google trend for "Ethereum" , "Ether" and "smart contract" modified for daily sequences (0-100)
69	TWEETS	Ethereum Tweets historical daily data

The second source of missing values which was related to holidays mostly infected the economical features like oil and gold price, Euro and Yuan exchange rate, WGS7YR and NASDAQ index. For this case the last value before the holiday was set for the missing value of the holidays and weekends.

The collected information is in a wide range of orders of magnitude (E +1 to E +22), so it will be helpful to scale all the time series in a common range to have a better comparison and analysis. The MinMax scalin in the range of (0, 1) was selected to accomplish this step. Now all the values are scaled in the range of (0, 1) and it's easier to compare them or calculate statistical information like correlation factors. It's also possible to use some transformations like log-transformation or square root-transformations to normalize data distribution of each time series. But, it wasn't applied in this part and we will investigate the effect of transformations as an option in the modelling section.

The last part of pre-processing was feature selection. Feature selection, is a critical part of data pre-processing and is required to improve model performance. The features were screened and selected using a series of different methods. Granger causality criteria, feature importance factor based on random forest, cross correlation and Variance Inflation Factor (VIF) are the main criteria which have been used to select and prune the most effective and relevant features.

Granger causality is a way to test causality between two variables in a time series. The method is a probabilistic account of causality; it uses empirical data sets to find patterns of correlation [25]. The definition of Granger causality can be explained as, whether past values of x aid in the prediction of yt, conditional on having already accounted for the effects on yt of past values of y (and perhaps of past values of other variables). Granger's causality tests the null hypothesis that the coefficients of past values in the regression equation are zero. Therefore, if the p-value obtained from the test is less than the significance level of 0.05, then, you can safely reject the null hypothesis [26].

Random forest is a machine learning method based on an ensemble of decision trees that is applicable to both regression and classification problems. Random forest employs hundreds of trees to provide better results. The training process is not complex and is useful for preliminary quick evaluations. One of the merits of this algorithm is that the features which contribute to the outputs receiving importance scores. These importance factors can be considered for selecting most relative features [27]. When training a decision tree, the effect of each feature in decreasing the impurity can be calculated. Therefore, features that decrease the impurity are more important in decision making. In random forests, this process is applied by averaging the impurity decrease across all trees [28].

Cross-correlation is an established and reliable tool to compute the degree to which the two time-series are dependent on each other considering a range of lag times. In fact cross-correlation is a vector of correlation factors between two time series when one is time shifted forward or backward relative to the other one [29].

Variance Inflation Factor (VIF) is used for computing the collinearity in a multiple regression model. It compares the difference between a model with multiple features and the same model

with a lone feature. This indicates the variability that occurs in the model due to having a feature that correlates with another feature present in the model [27]. In another word the greater the VIF is, then the explanatory variable is higher collinear with the other explanatory variables, and the parameter estimates will have large standard errors because of this. So it can be a good practice to omit the features with the highest VIF values.

In this study we've computed these four explained criteria to provide a reliable filter for selecting the most relevant features in order to reduce the computational cost and to increase the results efficiency. All the computations are performed in Python using numpy, pandas, sklearn and statmodels libraries. For Granger criteria, the p-values are computed for all features with respect to the target which is Ether price and a value greater than 0.05 is an evidence of lack of causality and a good sign for removing the feature. The random forest importance factor was calculated for all features with a lag interval of 14 days using 2000 trees. As a result the model contained 966 decision variables (60 ×14) and it selected those which their importance was greater than 50% of the mean importance of all the variables. In this way, for each feature the number of days (between 0 and 14) which are selected as effective variables were calculated. The less lagged days selected as important, the less important that feature can be. Features having less than 7 days as the important variables were candidates for elimination. The cross-correlation of all features with respect to Ether price in 14 lagged days backward and forward were computed and the maximum absolute value of correlation factors for each feature was taken into account as a feature selection criteria. Here, maximum absolute cross-correlations less than 0.4 were the cut-off for extracting less relevant features. Finally VIF for all features were calculated and the 15 features with the highest values were chosen as candidates for ignoring. The values of these factors computed for all features are summarized in Table 5.

It should be noticed that the aim of this analysis was to find the candidate features to remove from the initial data set. According to the evaluation results we decided to remove features that have one of the following conditions:

- All the features which have violated at least two criteria of these four
- All the features that have violated the random forest importance factor
- Features related to Polkadot, Solana and Chainlink due to high number of missing values.

According to these filters, totally 26 features were selected to be removed from the initial data frame (Table 5). Now we have a data set with 43 features which are going to be used in the next part to set up deep learning models to predict future price of Ether.

Table 5- Screen criteria applied for feature selection (The yellow highlight shows the violated criteria and red highlight shows the selected features to be omitted)

NO.	Features	Granger causality p-value	VIF	Random Forest Important factor	Max-abs Cros-Cor	No. of violated Criteria
1	EtherPrice	Target Variable				
2	ADA_Price(USDT)	0.00	81.84	8.00	0.92	0.00
3	ADA_Volume(USDT)	0.00	10.40	4.00	0.46	1.00
4	ADA_Tradecount	0.00	23.24	6.00	0.84	1.00
5	BNB_Price(USDT)	0.00	164.95	14.00	0.89	1.00
6	BNB_Volume(USDT)	0.00	4.86	14.00	0.38	1.00
7	BNB_Tradecount	0.00	18.02	11.00	0.83	0.00
8	BTC_Price(USDT)	0.00	243.77	12.00	0.91	1.00
9	BTC_Volume(USDT)	0.00	13.74	6.00	0.32	2.00
10	BTC_Tradecount	0.00	69.25	10.00	0.73	0.00
11	DOT_Price(USDT)	0.00	136.22	14.00	0.89	1.00
12	DOT_Volume(USDT)	0.00	17.31	6.00	0.60	1.00
13	DOT_Tradecount	0.00	41.61	7.00	0.84	0.00
14	EOS_Price(USDT)	0.01	8.66	8.00	0.22	1.00
15	EOS_Volume(USDT)	0.00	36.31	7.00	0.71	0.00
16	EOS_Tradecount	0.00	38.30	7.00	0.76	0.00
17	ETC_Price(USDT)	0.00	34.01	12.00	0.72	0.00
18	ETC_Volume(USDT)	0.00	11.12	10.00	0.54	0.00
19	ETC_Tradecount	0.00	15.37	9.00	0.59	0.00
20	ETH_Volume(USDT)	0.00	20.43	10.00	0.35	1.00
21	ETH_Tradecount	0.00	57.84	9.00	0.83	0.00
22	LINK_Price(USDT)	0.00	114.80	13.00	0.86	0.00
23	LINK_Volume(USDT)	0.06	12.31	13.00	0.21	2.00
24	LINK_Tradecount	0.00	30.94	10.00	0.62	0.00
25	LTC_Price(USDT)	0.00	40.74	9.00	0.88	0.00
26	LTC_Volume(USDT)	0.00	19.15	12.00	0.56	0.00
27	LTC_Tradecount	0.00	26.21	13.00	0.79	0.00
28	SOL_Price(USDT)	0.00	131.16	13.00	0.90	1.00
29	SOL_Volume(USDT)	0.00	7.84	10.00	0.55	0.00
30	SOL_Tradecount	0.00	18.53	9.00	0.82	0.00
31	TRX_Price(USDT)	0.00	40.66	13.00	0.75	0.00
32	TRX_Volume(USDT)	0.00	10.78	7.00	0.48	0.00
33	TRX_Tradecount	0.00	20.56	10.00	0.71	0.00
34	XLM_Price(USDT)	0.00	51.70	8.00	0.75	0.00
35	XLM_Volume(USDT)	0.00	17.60	5.00	0.43	1.00
36	XLM_Tradecount	0.00	28.20	5.00	0.71	1.00
37	XRP_Price(USDT)	0.00	13.43	13.00	0.66	0.00

38	XRP_Volume(USDT)	0.00	24.48	5.00	0.57	1.00
39	XRP_Tradecount	0.00	47.95	4.00	0.71	1.00
40	BTC-Dominance	0.00	39.05	14.00	-0.34	1.00
41	AddressCount	0.00	3698.17	14.00	0.53	1.00
42	AverageDailyTransactionFee	0.00	28.19	11.00	0.85	0.00
43	AvgGasPrice	0.00	10.17	12.00	0.51	0.00
44	BlockCountRewards	0.28	204.17	5.00	0.20	4.00
45	BlockDifficulty	0.00	9011.65	14.00	0.91	1.00
46	BlockReward	0.24	46.14	7.00	-0.07	2.00
47	BlockSize	0.00	30.25	13.00	0.74	0.00
48	BlockTime	0.41	136.91	7.00	-0.18	3.00
49	DailyActiveERC20Address	0.73	3.94	13.00	0.30	2.00
50	DailyActiveEthAddress	0.01	35.00	14.00	0.84	0.00
51	EnsRegistrations	0.98	1.09	1.00	-0.04	3.00
52	GasLimit	0.00	587.93	11.00	0.59	1.00
53	GasUsed	0.01	1396.95	12.00	0.60	1.00
54	MarketCap	0.05	3076.13	14.00	0.37	2.00
55	NetworkHash	0.00	9411.27	14.00	0.91	1.00
56	NetworkUtilization	0.08	140.09	11.00	0.47	2.00
57	20txns-ERC20 daily token transfer	0.02	23.65	11.00	0.55	0.00
58	TransactionFee	0.00	30.97	11.00	0.60	0.00
59	Ethereum daily transactions	0.01	59.92	13.00	0.74	0.00
60	Uncles	0.18	166.61	1.00	-0.05	4.00
61	verified-contracts	0.00	4.13	12.00	0.45	0.00
62	USD_EURO	0.06	27.58	13.00	-0.61	1.00
63	USD_YUAN	0.44	35.71	11.00	-0.60	1.00
64	WGS7YR	0.13	110.75	10.00	-0.19	2.00
65	Gold_Price	0.02	70.77	7.00	0.38	1.00
66	NASDAQ_Ind	0.00	49.44	12.00	0.79	0.00
67	OIL_PRICE	0.48	24.62	3.00	0.17	3.00
68	GOOGLE_TREND	0.00	36.54	11.00	0.93	0.00
69	TWEETS	0.00	4.62	13.00	0.71	0.00

Ether price prediction applying deep learning methods

After selecting 43 features as the main effective parameters for predicting Ether price, a logical connection between these variables and price changes must be stablished. In this research, we will investigate the efficiency of deep learning methods with higher level features rather than the traditional models for the following reasons. First of all, cryptocurrency prices are extremely unstable and non-stationary which is against the presumptions of traditional statistical methods of time series analysis. Secondly, there are a large number of features in the data and the proposed deep learning methodology can cope with autocorrelation, seasonality and trend effects, which is a time consuming manual tuning process in pure time-series models. Moreover, designing an efficient machine learning system requires significant knowledge about the data and the field of the problem. However, deep learning algorithms have developed as the novel advanced artificial intelligence which includes models that utilize multiple layers to represent latent features at a higher and more abstract level [30]. The underlying feature connections and representations are recognized from data rather than constructed by human engineers. As a result, deep learning models have been successfully applied in many fields relevant to time series prediction, like remote sensing, multi-sensor fusion, etc [31] [32]. They have also been applied effectively to realizing complex relationships between multiple time series. So here, it's been decided to focus on applying deep learning methods (DLM) in price prediction problem.

Different variants of 3 main classes of deep learning methods including Multi-Layer Perceptron (MLP), Convolutional Neural Networks (CNN) and Long Short Term Memory (LSTM) have been selected to be trained using selected features. In all modelling trials, the variables were normalized using MinMax scaler, also three different options for transformation were considered to be tested including without (W/O) transformation, Square-Root transformation and Log transformation. 14 days were considered as the lag time for input time series. In the other words, for each prediction, data from last 14 days were fed into the model. The prediction windows were considered for one day and one week ahead. The Architecture and the options applied in all constructed models are summarized in Tables 6-8. The Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE) and the Run-Time for each model are reported as well. It should be noticed here that 'relu' was the activation function for all dense layers. Moreover, the cost function and the training method in all models were 'mse' and 'adam' respectively. The general construction processes of these models are inspired from [33].

Table 6- The results of Ether price prediction using Multi-Layer Perceptron (MLP)

Run No	DL Model	Architecture	Transformation	Forecast Interval	Run-Time (sec.)	MAPE (%)	RMSE (USDT)
1	**Plain MLP**	Dense(700, activation='relu') + Dense(500, activation='relu') + Dense(n_out)	W/O	1 Week	346.07	46.40	1195.08
2			W/O	1 day	346.00	36.51	910.86
3			Square-Root	1 Week	321.46	47.18	1177.53
4			Square-Root	1 day	339.00	35.72	881.52
5			Log	1 Week	331.15	46.58	1095.54
6			Log	1 day	341.68	16.04	368.68
7	**Multi-Headed MLP**	n-features * Dense(50, activation ='relu') + Dense(n_out)	W/O	1 Week	162.95	17.36	544.89
8			W/O	1 day	207.41	10.29	324.89
9			Square-Root	1 Week	174.92	22.88	687.68
10			Square-Root	1 day	197.09	23.63	670.12
11			Log	1 Week	168.43	23.17	734.62
12			Log	1 day	199.72	8.83	267.52
13	**Multi-output MLP**	Dense(700, activation='relu') + Dense(500, activation='relu') + n_out * Dense(1)	W/O	1 Week	379.95	46.87	1205.37
14			W/O	1 day	340.96	33.34	875.71
15			Square-Root	1 Week	364.20	46.45	1191.60
16			Square-Root	1 day	344.35	34.45	864.07
17			Log	1 Week	373.31	31.73	718.74
18			Log	1 day	348.36	23.69	521.81
19	**Multi-headed Multi-output MLP**	n-features * Dense(50, activation='relu') + n_out * Dense(1)	W/O	1 Week	233.91	19.66	624.35
20			W/O	1 day	205.02	14.81	427.22
21			Square-Root	1 Week	224.07	23.22	685.72
22			Square-Root	1 day	196.98	20.58	606.43
23			Log	1 Week	231.89	17.89	506.01
24			Log	1 day	196.90	10.32	307.95

Table 7- The results of Ether price prediction using Convolutional Neural Networks (CNN)

Run No	DL Model	Architecture	Transformation	Forecast Interval	Run-Time (sec.)	MAPE (%)	RMSE (USDT)
1	**Plain CNN**	Conv1D(filters=64, kernel_size=2, activation='relu') + MaxPooling1D(pool_size=2) + Flatten() + Dense(500, activation='relu') + Dense(n_out)	W/O	1 Week	78.66	55.21	1366.81
2			W/O	1 day	73.70	53.66	1338.43
3			Square-Root	1 Week	69.69	53.39	1319.47
4			Square-Root	1 day	74.85	42.48	1096.23
5			Log	1 Week	71.03	24.32	631.50
6			Log	1 day	89.42	22.15	601.92
7	**Multi-headed CNN**	n-features * [Conv1D(filters=64, kernel_size=2, activation='relu') + MaxPooling1D(pool_size=2) + Flatten()] + Dense(500, activation='relu') + Dense(n_out)	W/O	1 Week	3067.70	24.42	749.55
8			W/O	1 day	3639.75	20.61	611.21
9			Square-Root	1 Week	3209.67	25.32	717.90
10			Square-Root	1 day	3089.74	26.23	650.37
11			Log	1 Week	3119.80	27.78	760.05
12			Log	1 day	3247.85	29.35	724.64
13	**Multi-output CNN**	Conv1D(filters=64, kernel_size=2, activation='relu') + MaxPooling1D(pool_size=2) + Flatten() + Dense(700, activation='relu') + n_out * Dense(1)	W/O	1 Week	224.31	55.35	1370.22
14			W/O	1 day	181.84	64.33	1561.12
15			Square-Root	1 Week	228.06	34.20	870.99
16			Square-Root	1 day	210.16	26.66	770.36
17			Log	1 Week	223.36	31.93	821.15
18			Log	1 day	182.95	31.23	792.48
19	**Multi-headed Multi-output**	n-features * [Conv1D(filters=64, kernel_size=2, activation='relu') + MaxPooling1D(pool_size=2) + Flatten()] + Dense(500, activation='relu') + n_out * Dense()	W/O	1 Week	3105.54	27.35	752.73
20			W/O	1 day	3211.35	18.82	551.84
21			Square-Root	1 Week	3250.60	27.26	759.21
22			Square-Root	1 day	3105.99	35.42	879.67
23			Log	1 Week	3103.32	30.38	813.50
24			Log	1 day	3095.42	24.22	575.96

Table 8- The results of Ether price prediction using Long Short Term Memory (LSTM)

Run No	DL Model	Architecture	Transformation	Forecast Interval	Run-Time (sec.)	MAPE (%)	RMSE (USDT)
1	**Vanilla LSTM**	LSTM(50, activation='relu') + Dense(n_out)	W/O	1 Week	33.56	47.66	1196.71
2			W/O	1 day	32.30	45.90	1207.50
3			Square-Root	1 Week	31.14	35.61	894.99
4			Square-Root	1 day	37.91	15.28	394.58
5			Log	1 Week	29.48	40.91	1050.27
6			Log	1 day	34.97	9.50	293.92
7	**Stacked LSTM**	LSTM(50, activation='relu', return_sequences=True) + LSTM(50, activation='relu') + Dense(n_out)	W/O	1 Week	54.98	55.31	1438.06
8			W/O	1 day	59.43	42.18	1114.21
9			Square-Root	1 Week	54.74	49.93	1281.18
10			Square-Root	1 day	60.21	45.34	1166.24
11			Log	1 Week	57.43	44.04	1065.32
12			Log	1 day	60.15	14.41	343.49
13	**Bidirectional LSTM**	Bidirectional(LSTM(50, activation='relu')) + Dense(n_out)	W/O	1 Week	35.81	38.83	951.80
14			W/O	1 day	40.79	33.03	862.14
15			Square-Root	1 Week	35.90	45.48	1176.45
16			Square-Root	1 day	40.00	35.82	919.80
17			Log	1 Week	35.11	33.13	888.35
18			Log	1 day	40.52	11.45	312.70
19	**CNN-LSTM**	TimeDistributed(Conv1D(filters=64, kernel_size=1, activation='relu')) + TimeDistributed(MaxPooling1D(pool_size=2)) + TimeDistributed(Flatten()) + LSTM(50, activation='relu') + Dense(n_out)	W/O	1 Week	23.28	48.77	1231.69
20			W/O	1 day	29.31	47.91	1230.87
21			Square-Root	1 Week	24.03	49.44	1284.26
22			Square-Root	1 day	28.77	15.21	372.88
23			Log	1 Week	25.37	28.94	816.78
24			Log	1 day	28.01	19.39	533.31
25	**ConvLSTM**	ConvLSTM2D(filters=64, kernel_size=(1,2), activation='relu') + Flatten() + Dense(n_out)	W/O	1 Week	39.26	60.31	1495.69
26			W/O	1 day	41.77	32.33	920.77
27			Square-Root	1 Week	40.62	54.48	1351.92
28			Square-Root	1 day	41.37	10.54	293.80
29			Log	1 Week	37.65	28.77	746.74
30			Log	1 day	44.88	20.28	584.81
31	**Encoder-Decoder LSTM**	LSTM(100, activation='relu') + RepeatVector(n_out) + LSTM(100, activation='relu', return_sequences=True) + TimeDistributed(Dense(1))	W/O	1 Week	65.28	51.40	1263.77
32			W/O	1 day	51.06	41.65	1113.92
33			Square-Root	1 Week	62.06	54.02	1377.54
34			Square-Root	1 day	49.13	34.15	922.40
35			Log	1 Week	64.49	41.18	1211.83
36			Log	1 day	51.39	9.63	299.51

These models are trained using data from time interval of 2018-01-01 to 2020-12-27 and the prices are predicted for 2020-12-28 to 2021-05-30. One important thing is that due to the stochastic nature of the algorithms, the results in each run will be different and it is better to consider running each model several times. Nevertheless, we didn't do it for all of the models and just some of them were selected to retrain quite a few times. It's been decided so, because first of all there are 84 models with different run times ranging from half a minute to about one hour, so trying to repeat all the models multiple times will be very time consuming. The second reason is that the aim of this part is to compare the performance of different assemblies of deep learning methods in order to select the best candidates to improve and optimize in the next step. Furthermore, it's been observed from those models with numerous reiterations that the general behavior of a model will not change just by retraining it, and its performance may be improved or worsened just a bit. Hence, the main purpose of comparison will not be lost by the practice applied here. In order to get a better insight about the results obtained, some statistical analyses have been performed and the findings will be discussed in the following.

At first, the results were split according to 'Forecast Interval'. In this way we can compare the results easier both for '1 Day' and '1 Week' predictions. The general comparison between two different prediction intervals is depicted in Figure 11.

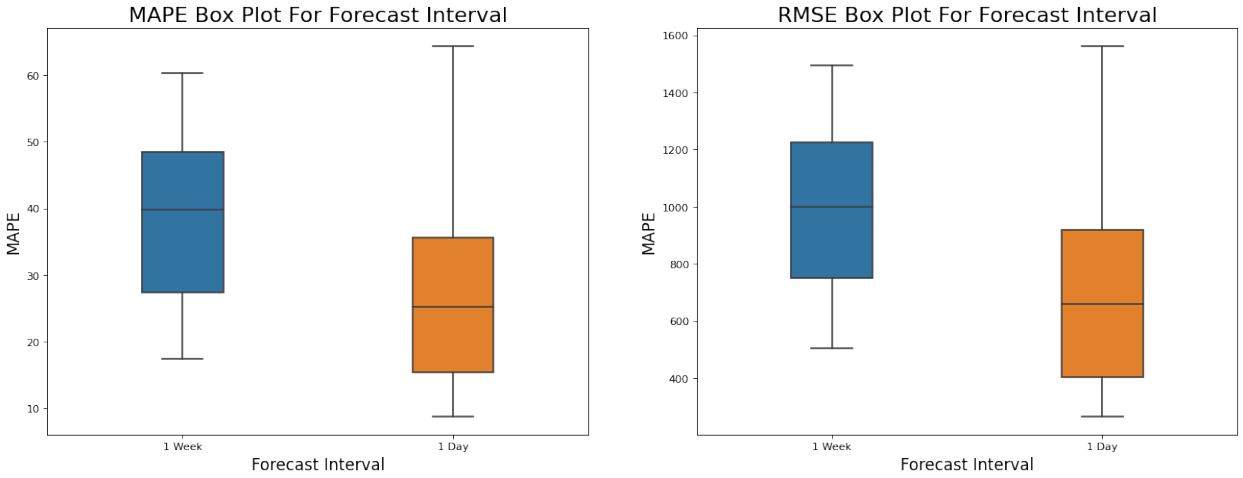

Figure 11- Comparison of RMSE and MAPE for different prediction intervals

It's evident from Figure 11 that forecasting the Ether price for one week ahead has more erroneous results with MAPE (38.44 ± 12.44) and RMSE (1000.56 ± 281.10) comparing to one day interval with MAPE (27.32 ± 13.51) and RMSE (715.90 ± 332.13). The summary of information for '1 Day' and '1 Week' groups are presented in Tables 9 &10.

Table 9- Factors which are going to be studied in model selection

	Variable	Outcome	Count
1	Model	LSTM	18
2		MLP	12
3		CNN	12
4	Transformation	Square-Root	14
5		W/O	14
6		Log	14

Table 10- Summary of MAPE statistics for '1 Day' and '1 Week' prediction intervals

group	Variable	N	Mean	SD	SE	95% Conf.	Interval
'1 Day'	MAPE	42	27.3183	13.5105	2.0847	23.1082	31.5285
'1 Week'	MAPE	42	38.4407	12.4435	1.9201	34.563	42.3184

In the following the performance of deep learning models and the effect of data transformation in each forecast interval will be investigated using two-way ANOVA. In order to prevent redundant computations, MAPE will be the evaluation metrics in statistical analysis. The summary of MAPE data are collected in table 10. In order to determine the significance of 'Model' and 'Transformation' in MAPE changes for both prediction intervals two-way ANOVA were applied and the results are presented in Tables 11 and 12.

Table 11- ANOVA table for '1 Day' prediction interval

	sum_sq	df	F	PR(>F)
Intercept	30383.38	1	250.9313	5.52E-17
C(Model, Sum)	677.3219	2	2.796944	0.075496
C(Transformation, Sum)	1790.16	2	7.392318	0.002224
C(Model, Sum):C(Transformation, Sum)	630.2784	4	1.301341	0.289771
Residual	3995.722	33		

Table 12- ANOVA table for '1 Week' prediction interval

	sum_sq	df	F	PR(>F)
Intercept	56541	1	471.0711	4.15E-21
C(Model, Sum)	1346.037	2	5.607252	0.008011
C(Transformation, Sum)	709.7955	2	2.956829	0.065886
C(Model, Sum):C(Transformation, Sum)	198.5567	4	0.413569	0.797583
Residual	3960.873	33		

As the results suggest, the combination of 'Model' and 'Transformation' was not significant, so we remove this combination and we can investigate the effect of these parameters separately using one-way ANOVA. The summary of information for the effect of 'Model' and 'Transformation' in MAPE and RMSE for two prediction intervals are shown in Tables 13 & 14.

Table 13- Statistical information for the effect of 'model' in prediction accuracy

Forecast Interval	Model	Sample	RMSE					MAPE				
			Mean	SD	SE	95% Conf.	Interval	Mean	SD	SE	95% Conf.	Interval
1 Day	CNN	12	846.19	323.81	93.48	640.45	1051.92	32.93	14.00	4.04	24.03	41.83
	LSTM	18	715.94	365.95	86.26	533.95	897.92	26.89	14.20	3.35	19.83	33.95
	MLP	12	585.57	249.56	72.04	427.00	744.13	22.35	10.54	3.04	15.65	29.05
1 Week	CNN	12	911.09	272.53	78.67	737.93	1084.25	34.74	12.36	3.57	26.89	42.60
	LSTM	18	1151.30	219.64	51.77	1042.07	1260.52	44.90	9.20	2.17	40.33	49.48
	MLP	12	863.93	281.80	81.35	684.88	1042.97	32.45	13.08	3.78	24.14	40.76

Table 14- Statistical information for the effect of 'Transformation' in prediction accuracy

Forecast Interval	Transformation	Sample	RMSE					MAPE				
			Mean	SD	SE	95% Conf.	Interval	Mean	SD	SE	95% Conf.	Interval
1 Day	Log	14	466.34	174.42	46.62	365.63	567.04	17.89	7.57	2.02	13.52	22.26
	Square-Root	14	749.18	264.98	70.82	596.18	902.17	28.68	10.63	2.84	22.54	34.82
	W/O	14	932.19	358.62	95.85	725.13	1139.25	35.38	15.42	4.12	26.48	44.29
1 Week	Log	14	847.17	195.66	52.29	734.20	960.14	32.20	8.31	2.22	27.40	37.00
	Square-Root	14	1055.46	269.78	72.10	899.69	1211.23	40.63	12.08	3.23	33.66	47.61
	W/O	14	1099.05	314.82	84.14	917.28	1280.82	42.49	14.44	3.86	34.16	50.83

Then the results of one-way ANOVA were evaluated and reported in Table 15. The criterion for evaluation is MAPE.

Table 15- one-way ANOVA results

Factors	' 1 Day" forecast						
	sum_sq	df	mean_sq	F	PR(>F)	eta_sq	omega_sq
C(Model)	677.321914	2	338.660957	1.940452	0.157258	0.090504	0.042864
Residual	6806.547069	39	174.526848	NaN	NaN	NaN	NaN
C(Transformation)	2180.546533	2	1090.27327	8.017739	0.001211	0.291366	0.250475
Residual	5303.32245	39	135.982627	NaN	NaN	NaN	NaN
	' 1 Week" Forecast						
C(Model)	1346.03707	2	673.018534	5.246963	0.009594	0.212025	0.168217
Residual	5002.46001	39	128.268205	NaN	NaN	NaN	NaN
C(Transformation)	843.030386	2	421.515193	2.985958	0.062145	0.132792	0.086399
Residual	5505.46669	39	141.165813	NaN	NaN	NaN	NaN

According to these results it can be concluded that for '1 Day' interval the effect of 'Model' was not significant but the 'Transformation' could make a significant difference. However, in '1 Week' interval 'Model' had a significant effect while different 'Transformation' methods were not significantly different in outputs. Next, by conducting post-hoc test using Bonferroni approach, it allowed to see which group(s) significantly differ from each other. Results are presented in Tables 16 and 17.

Table 16- Bonferroni post-hoc test for 'Transformation' effect on 1 Day forecast

1 Day					
group1	**group2**	**stat**	**pval**	**pval_corr**	**reject**
Log	**Square-Root**	-3.0938	0.0047	0.014	TRUE
Log	**W/O**	-3.8105	0.0008	0.0023	TRUE
Square-Root	**W/O**	-1.3395	0.192	0.576	FALSE

Table 17- Bonferroni post-hoc test for 'Model Type' effect on 1 Week forecast

1 Week					
group1	**group2**	**stat**	**pval**	**pval_corr**	**reject**
CNN	**LSTM**	-2.5819	0.0154	0.0461	TRUE
CNN	**MLP**	0.4414	0.6632	1	FALSE
LSTM	**MLP**	3.0679	0.0047	0.0142	TRUE

By plotting box-plot of MAPE and RMSE for above cases it can be seen that the results of the Bonferroni test will be confirmed by these plots. According to Table 16, for one day forecast the difference between 'Log' transformation with both 'Square-Root' and 'W/O' transformations were significant while a considerable difference between 'Square-Root' and 'W/O' transformations was not detected. In Figure 12 the superior performance of 'Log' transformation compared to the other transformations is clear while the accuracy of 'Square-Root' is just a little better than 'W/O' transformation. In case of one week forecast, the different performance between 'LSTM' model with the other two models was admitted with Bonferroni test and in figure 13 it is obvious that 'LSTM' model has significant worse accuracy comparing to 'MLP' and 'CNN' models.

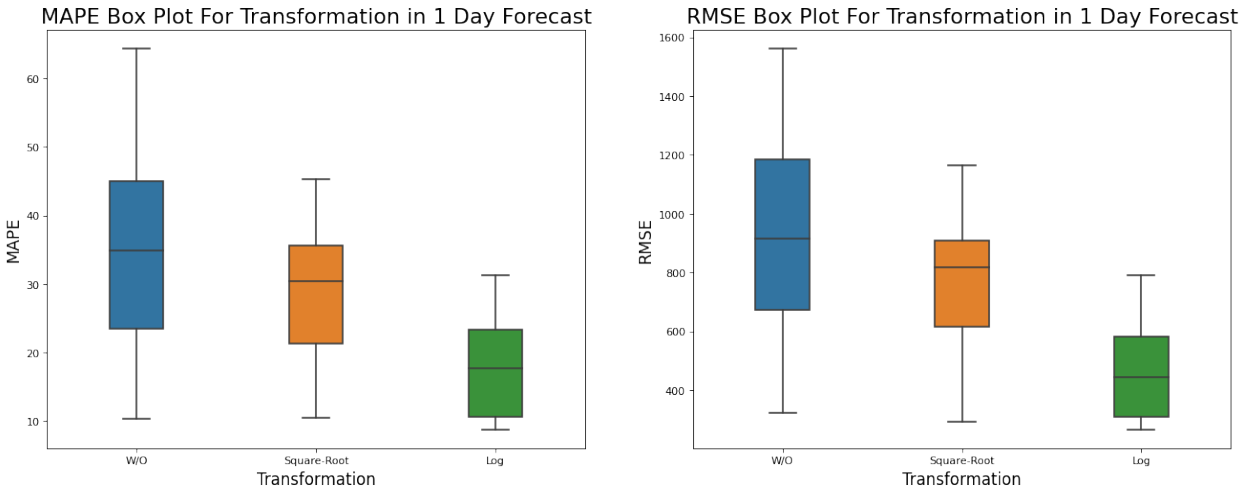

Figure 12 – Box-plot of MAPE and RMSE for ‘Transformation’ in ‘1 Day’ forecasts

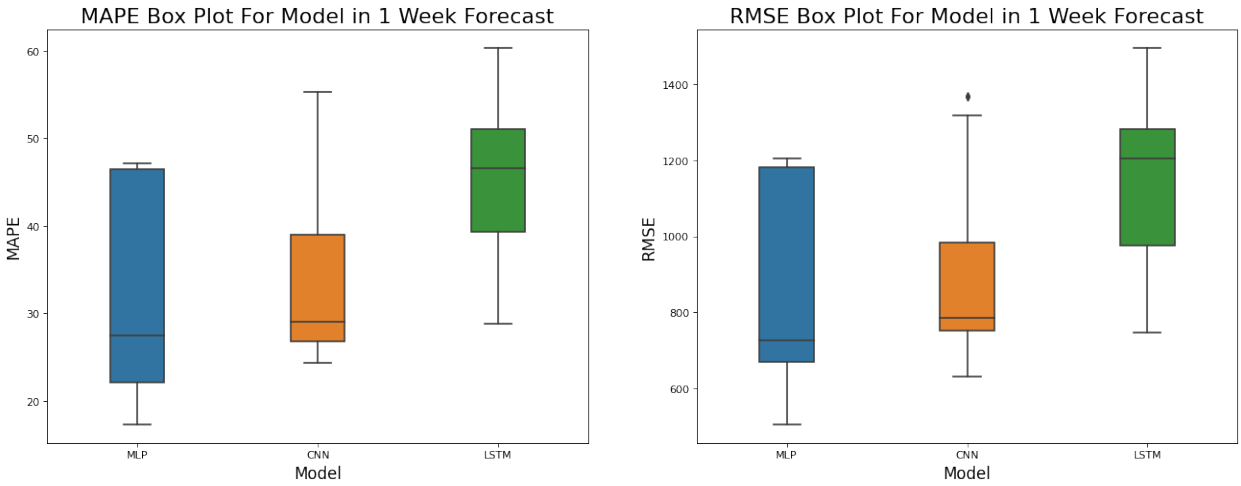

Figure 13 – Box-plot of MAPE and RMSE for ‘Model’ in ‘1 Week’ forecasts

To sum up the results of this section, it can be stated that our selected deep learning methods have different behaviors in short and long term predictions. While in ‘1 day’ forecast all the deep learning methods had relatively similar accuracy, the ‘Log’ transformation could make a significant improvement in accuracy of predictions. On the other hand for ‘1 week’ forecast none of the transformation methods could make a meaningful improvement. Although, model types had noticeable different performances as ‘LSTM’ had the worse accuracy and ‘MLP’ and ‘CNN’ acted similarly in prediction of Ether price. In order to select best candidates for ‘1 day’ prediction we should consider ‘Log’ transformation, but for the model we can check the performances in Tables 6 to 8 and select those with the lowest MAPE and RMSE. A number of candidates for one day prediction are listed in Table 18. The prediction graph of the best model is also depicted in Figure 14.

Table 18- Selected best models for 1 day Ether price forecast

No.	DL Model	Transformation	Forecast Interval	Run-Time (sec.)	MAPE (%)	RMSE (USDT)
1	**Multi-headed MLP**	Log	1 day	199.72	8.83	267.52
2	**Multi-Headed Multi-output CNN**	Log	1 day	3095.42	24.22	575.96
3	**Vanilla LSTM**	Log	1 day	34.97	9.50	293.92
4	**Stacked LSTM**	Log	1 day	60.15	14.41	343.49
5	**Bidirectional LSTM**	Log	1 day	40.52	11.45	312.70
6	**CNN-LSTM**	Log	1 day	28.01	19.39	533.31
7	**Encoder-Decoder LSTM**	Log	1 day	51.39	9.63	299.51

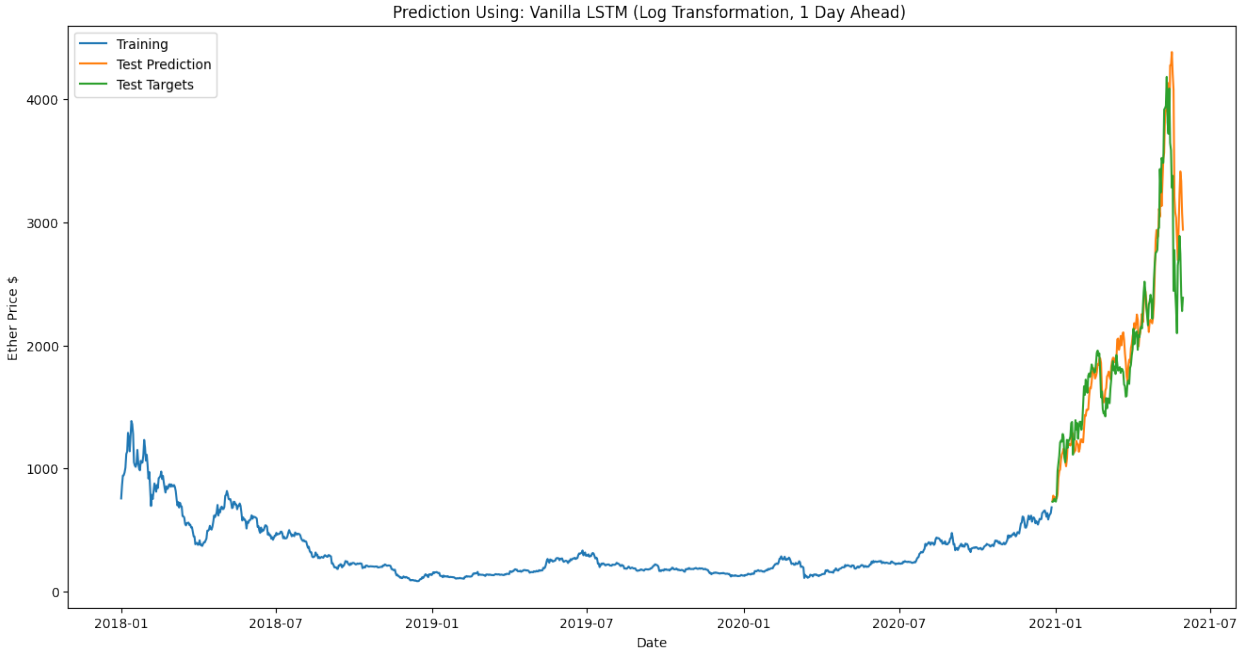

Figure 14- The best Ether price prediction model for 1 day interval using Vanilla LSTM and Log Transform

In one week prediction interval, 'LSTM' model didn't have promising outputs, so we'll select the best candidates from the other models. The transformation type is not effective here so it can be any choice. Five numbers of most accurate models in case of one week prediction are listed in Table 19. The best performance of 1 week prediction is visualized in Figure 15.

Table 19- Selected best models for one week Ether price forecast

No.	DL Model	Transformation	Forecast Interval	Run-Time (sec.)	MAPE (%)	RMSE (USDT)
1	**Multi-Headed MLP**	W/O	1 Week	162.95	17.36	544.89
2	**Multi-Headed Multi-output MLP**	Log	1 Week	231.89	17.89	506.01
3	**Plain CNN**	Log	1 Week	71.03	24.32	631.50
4	**Multi-headed CNN**	W/O	1 Week	3067.70	24.42	749.55
5	**Multi-headed Multi-output CNN**	Square-Root	1 Week	3250.60	27.26	759.21

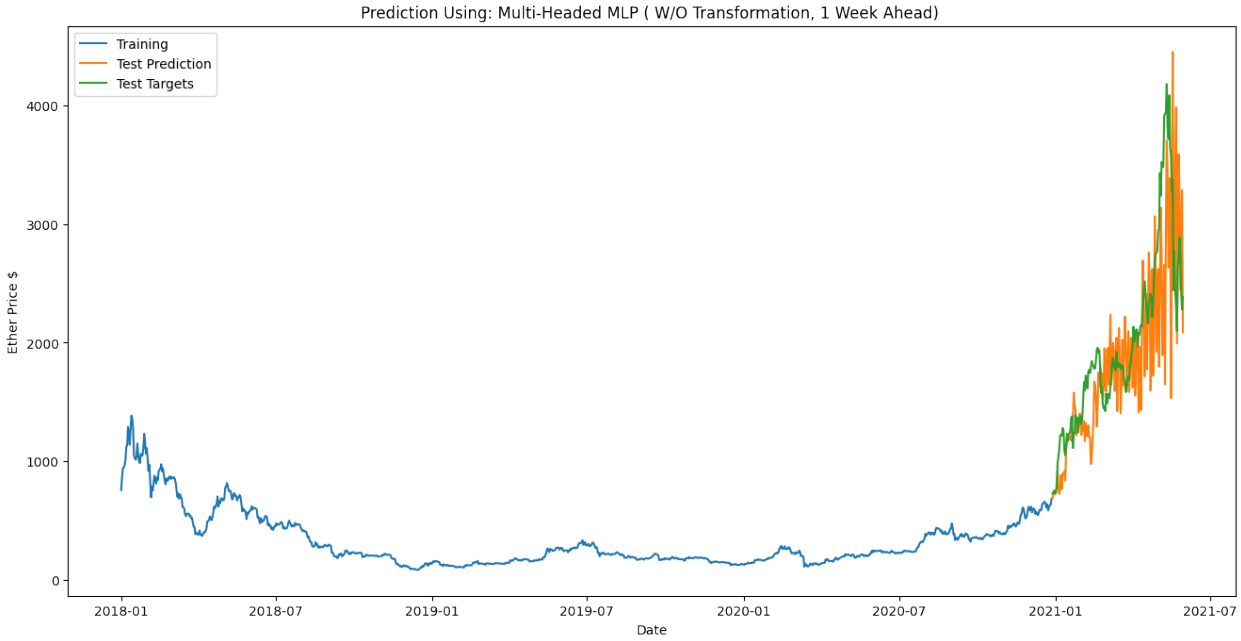

Figure 15- The best Ether price prediction model for 1 week interval using Multi-headed MLP and without Transform

The next part of this research will work on finding optimized hyper-parameters for selected best models and provide an ensemble base combination of optimized models in order to achieve the best and most reliable predictions both in short and long term intervals.

Improving deep learning models

After building different initial deep learning methods for predicting the Ether price based on various economical, technical and block-chain network factors, we investigated the effect of two transformation method including 'logarithmic' and 'squared root' and also the performance of different models for two forecasting intervals of '1 day' and '1 week' . Finally in last part, for both '1 day' and '1 week' periods, some models with and without transformation were selected based on their initial performances measured by error indicators like 'rmse' and 'mape'. In this final section, we aim to improve and enhance the predictions through some technical modifications like 'hyper-parameter tuning', 'model stacking' and 'time series filtering'. In the following the results for different prediction intervals, will be presented separately.

Improving '1-day' predictions by tuning hyper-parameters

Based on initial results obtained in previous step, six different models with best performances and also lowest running times were selected. In order to improve the predictions, we intended to change some hyper-parameters of the models like number of neurons, activation function, optimization algorithms, learning rate, number of epochs and batch size. The process of examination of the parameters was some kind of 'trial and error', not an optimized or greed search method. In this way we chose at least 3 different values for each parameter in a way that the values were higher or lower than the current applied ones, while the other parameters remained un-changed. In some cases we didn't observe any significant improvement in the results, so we just keep them as they were in the initial models. However, some other factors were effective and in these cases, more values were tested until the best ones were selected and applied to improve the initial models. The most effective factors we found here were, number of neurons, activation functions, optimization algorithms, batch size and number of epochs. It should be noted, that the effect of hyper-parameters of in model performances were evaluated based on training and evaluation sample sets that were built from the initial training set and the test samples were not applied in this process.

One important issue that we detected in our investigations was the random behavior of the neural network models. In other words, one model with the same architecture, same hyper-parameters and same train and test samples, can result in different predictions each time it's trained. The main reason of this issue is the random selection of the initial weights and also the stochastic behavior of the optimization algorithms. In order to mitigate this problem we run each model with the same parameters at least 5 times and record the best prediction results. The final modifications which have been implemented and the prediction results compare to the initial outputs are summarized in Table 20. It must be stated here that as the optimization algorithm 'adam' obtained far better results than the other ones, so it was applied in training of all models. As it can be observed from this table a significant improvements in model predictions have been achieved just by tuning the hyper-parameters using a manual, trial and error procedure. Of course by investing more time and developing more sophisticated optimized search or thorough greed search better results can be obtained.

Table 20- Initial parameters and performance of selected models for '1 day' price forecast

No.	DL Model	Modifications	Transformation	Initial Models			After Hyper-parameter tuning			Improvement %	
				Run-Time (sec.)	MAPE (%)	RMSE (USDT)	Run-Time (sec.)	MAPE (%)	RMSE (USDT)	MAPE	RMSE
1	**Multi-headed MLP**	kernel_regularizer batch_size=30, epochs=2000	Log	199.72	8.83	267.52	243.93	6.49	186.37	26.5	30.33
2	**Plain CNN**	filters=32, kernel_size=7 batch_size=30,epochs=3000	Log	89.42	22.15	601.92	90.52	12.73	431.30	42.5	28.35
3	**Vanilla LSTM**	Neurone=64 activation='sigmoid' batch_size=30, epochs=400	Log	34.97	9.50	293.92	77.79	5.89	168.54	38	42.66
4	**Stacked LSTM**	Neurone=50, 25 activation='sigmoid' batch_size=30, epochs=200	Log	60.15	14.41	343.49	59.89	9.77	240.88	32.20	29.87
5	**Bidirectional LSTM**	Neurone=16 activation='sigmoid' batch_size=30, epochs=400	Log	40.52	11.45	312.70	69.03	6.73	171.82	41.68	45.05
7	**Encoder-Decoder LSTM**	activation='sigmoid' batch_size=30, epochs=200	Log	51.39	9.63	299.51	58.99	7.26	184.45	24.61	38.42

Improving '1-day' predictions by stacking different realizations and models

Stacking or Stacked Generalization is an ensemble machine learning algorithm. It uses a meta-learning algorithm to learn how to best combine the predictions from two or more base machine learning algorithms. The benefit of stacking is that it can harness the capabilities of a range of well-performing models on a classification or regression task and make predictions that have better performance than any single model in the ensemble. It involves combining the predictions from multiple machine learning models on the same dataset, like bagging and boosting.

The architecture of a stacking model involves two or more base models, often referred to as level-0 models, and a meta-model that combines the predictions of the base models, referred to as a level-1 model.

- **Level-0 Models (*Base-Models*)**: Models fit on the training data and whose predictions are compiled.
- **Level-1 Model (*Meta-Model*)**: Model that learns how to best combine the predictions of the base models.
  The meta-model is trained on the predictions made by base models on out-of-sample data. That is, data not used to train the base models is fed to the base models, predictions are made, and these predictions, along with the expected outputs, provide the input and output pairs of the training dataset used to fit the meta-model.

The most common approach to preparing the training dataset for the meta-model is via k-fold cross-validation of the base models, where the out-of-fold predictions are used as the basis for the training dataset for the meta-model. Once the training dataset is prepared for the meta-model, the meta-model can be trained in isolation on this dataset, and the base-models can be trained on the entire original training dataset. Stacking is designed to improve modeling performance, although is not guaranteed to result in an improvement in all cases.
In our case study, we have developed and selected six different deep learning models based on neural networks. Therefore, these models can be applied for stacking. However, because of the randomness of the training process of initial models and also to make a thorough evaluation of the suggested deep learning models, a k-fold cross validation method with 10 fold was applied. In this way, 10 different training for each model were prepared using different train, test sample sets and the obtained models were saved. In fact, here for each neural network, there are 10 different models which are called here as 'realizations' and have same structure but different weights. It should be noticed that the initial test samples were not used in this evaluation and the initial training set was just split randomly in 10 subsets and for each realization 9 sets were used for the training and the other one was employed for validation purpose. As a result, now there are 60 predictive models that can be ensemble in a stacking model. For this aim, 'Linear Regression' and 'MLP' models were considered to be trained on outputs of 60 initial prediction models. The

results of the cross validation of initial and models are presented in Figure 16. According to the results multi-headed MLP (MHMLP) model has the lowest errors with 2.04±1.47 mean absolute percentage error (MAPE) and 10.68±8.35 root mean square error (RMSE). On the other hand CNN has the highest errors (RMSE=31.47±8.21, MAPE=5.21±0.66), while all the LSTM based models show relatively similar accuracy with values ranging from 3.66 to 4.47 as mean of MAPE and 18.92 to 22.24 as mean of RMSE.

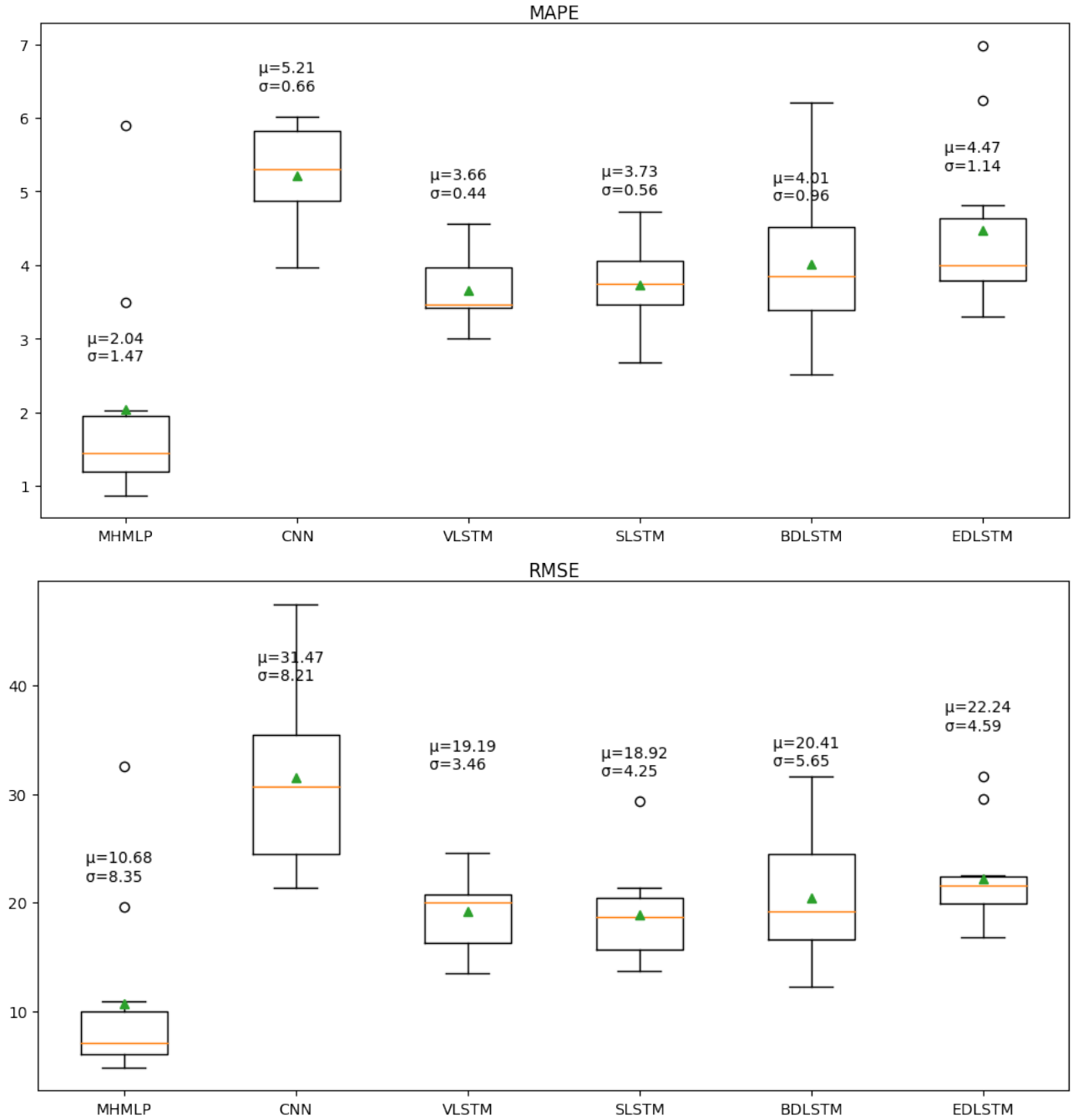

Figure 16- The evaluation of selected models using 10-fold cross validation method

An important point that we should notice, is the range of errors obtained for each model with different train, test sets. The length of box plots is a good indicator of the difference between best and worse realizations for each model and shows that while some models like multi-headed MLP (MHMLP) and encoder-decoder LSTM (EDLSTM) have relatively narrow range of variation in accuracy, models like convolutional neural networks (CNN) and bi-directional LSTM (BDLSTM) represent wider range of accuracy in predictions. Because of this observation, it's been decided to use all the 60 trained models to set up two stacked models using 'linear regression' and 'single layer MLP' (neurons = 200, activation='relu') as meta-models. The meta-models have been trained using all the initial training set and it was tested along with other initial models using test set. It should be noticed that the test set has not been used neither in training of initial models nor the meta-models, so it can be considered as a valid assessment of the prediction accuracy of the suggested models. The results are summarized in Figure 17.

According to these results, it's clear that prediction accuracies are not as good as those obtained in validation process. Despite the outputs of validation step, CNN shows the highest accuracy in prediction with MAPE equals to 12.16±4.99 and the value of 335.34±70.63 for RMSE. The higher errors in prediction can be justified as the result of sharp changes in the market trend during the test period. In fact it is obvious that the market behavior is significantly different in training and testing intervals. Moreover, different realizations of each model have a varied range of accuracy. For example in case of CNN the best prediction represents 8.22 as MAPE and 251 as RMSE. After CNN the best performance belongs to stacked Linear Regression model (SLR) with MAPE equals to 20.22 and RMSE equals to 559.01. It should be noted that for meta models as there is just one realization for each, we don't have a range of accuracies and it's just one number for each measurement in each model. Multi-head MLP (MHMLP) shows relatively similar numbers with stacked models for both RMSE and MAPE. Among LSTM based models, Bi-directional LSTM (BDLSTM) with MAPE of 26.17±7.31 and RMSE of 715.83±2013.10 has the best performance. Of course in both vanilla LSTM (VLSTM) and encoder-decoder LSTM (EDLSTM) there are realizations that have MAPE less than 10% and RMSE less than 200$. However, in order to compare the models here the mean values of different realizations of each model have been compared together. In order to have a better view of the predictions the best and worst predictions are going to be compared in Figure 18. The predictions based on stacked models are also depicted in Figure 19. Although the Stacking is not as good as the best prediction, it can perceive the general future trend of the Ether market and the volatility of the price. Due to the randomness of neural network training process, it seems that relying on just one model to predict the unseen conditions is not logical. Using stacking, may reduce the accuracy a bit compare to the best models, but as it is not obvious which model will result in the best predictions

before knowing about the future, applying the stacking method can assure us about considering all possibilities in our forecasts.

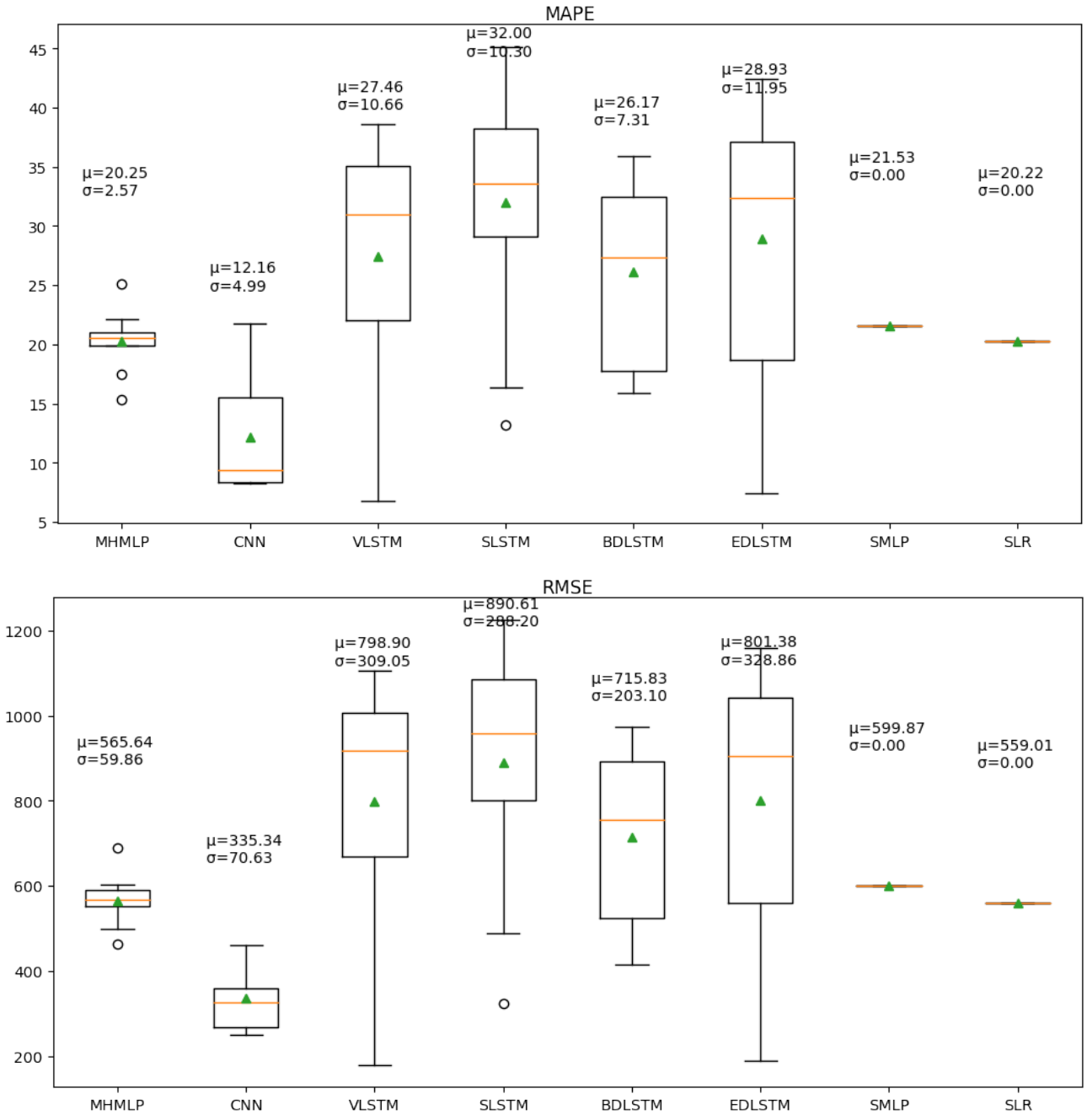

Figure 17- The evaluation of initial and stacked models using test set

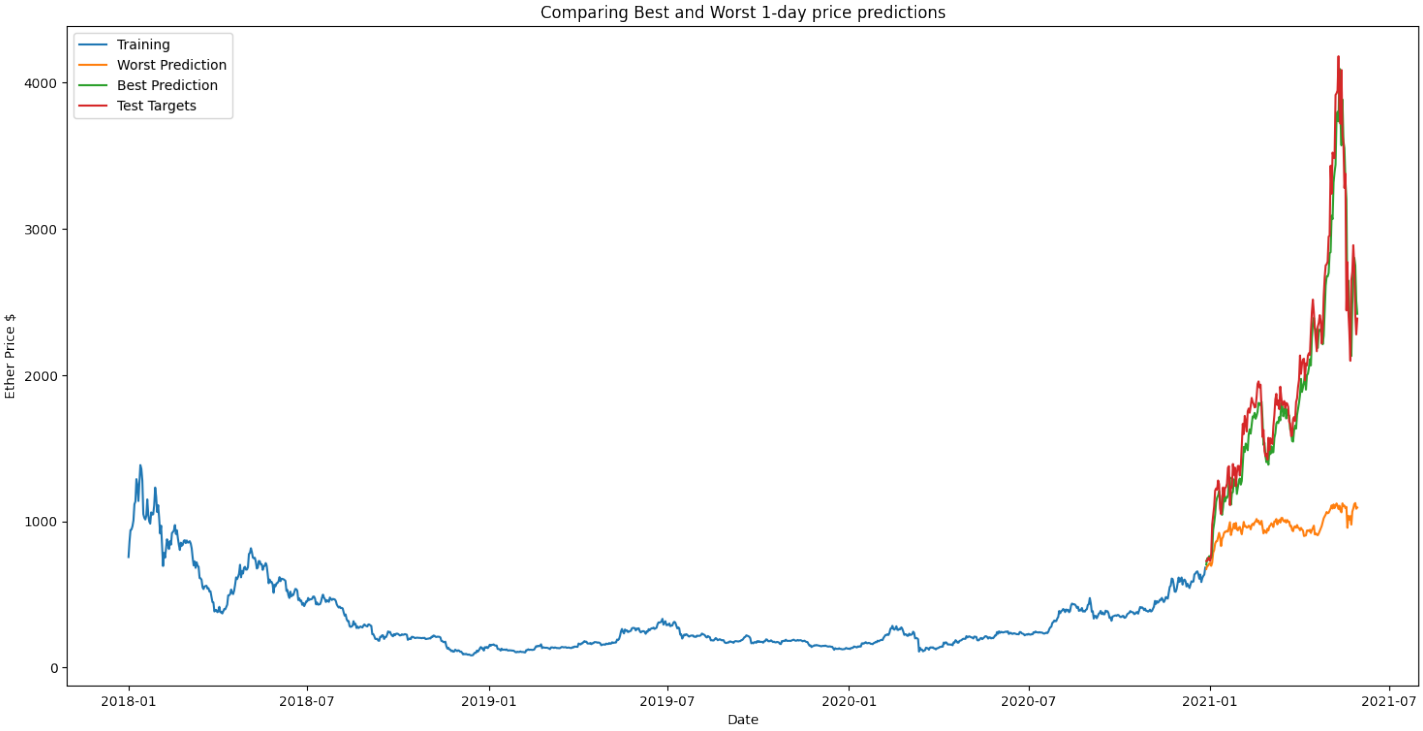

Figure 18- comparing the best and worst predictions obtained by 60 trained Deep Learning methods

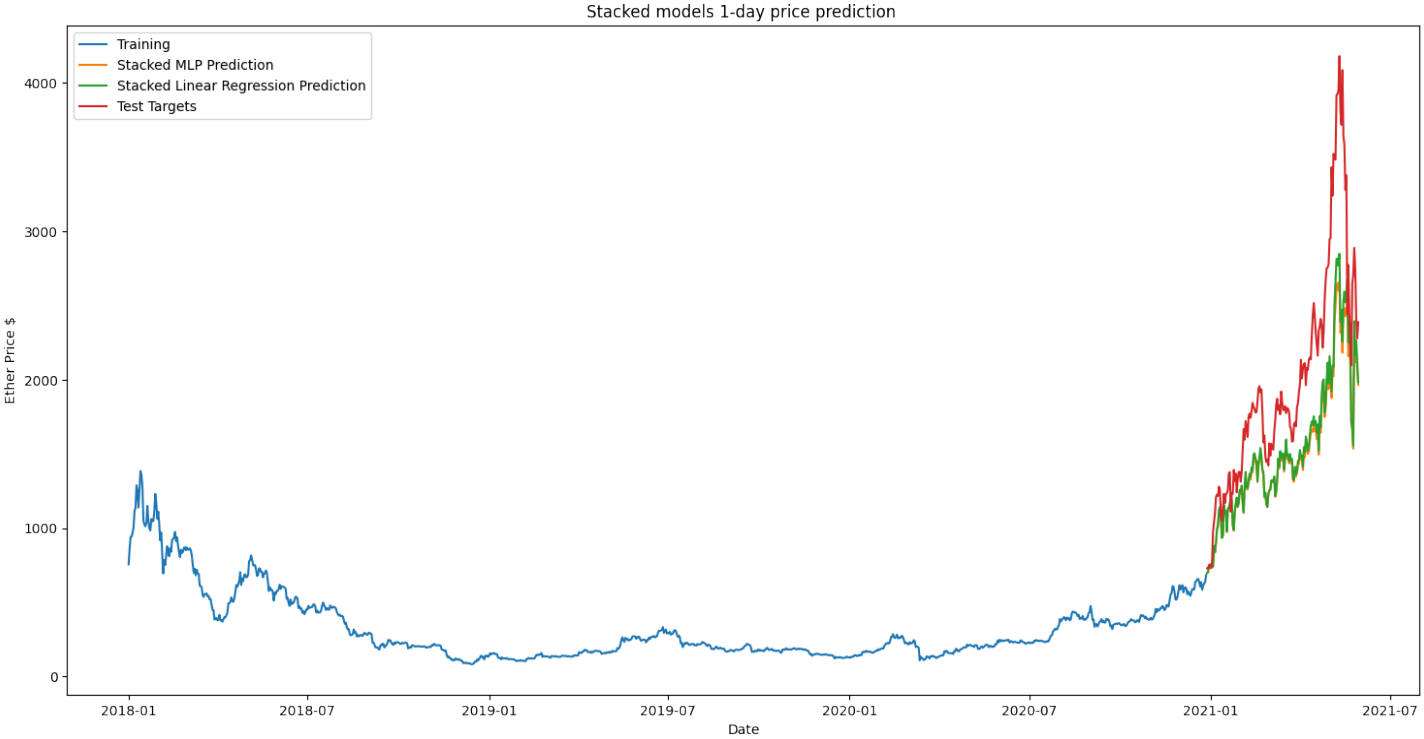

Figure 19- Prediction Ether price applying stacking method

Improving '1-week' prediction models

Generally, the prediction models for 1-week ahead, ware not as good and accurate as those for 1-day interval. However, we select best two models with the best performance in the initial stage which include multi-headed MLP (MHMLP) and multi-headed CNN (MHCNN). At first, it was tried to enhance the performance of the models by tuning the hyper-parameters, but no significant improvement were achieved. The evaluation of these models based on 10-fold cross validation is summarized in Figure 20 and the price predictions for the test samples are also illustrated in figures 21 & 22. Notice that this is the output of one of 10 trained models for each of the deep learning models.

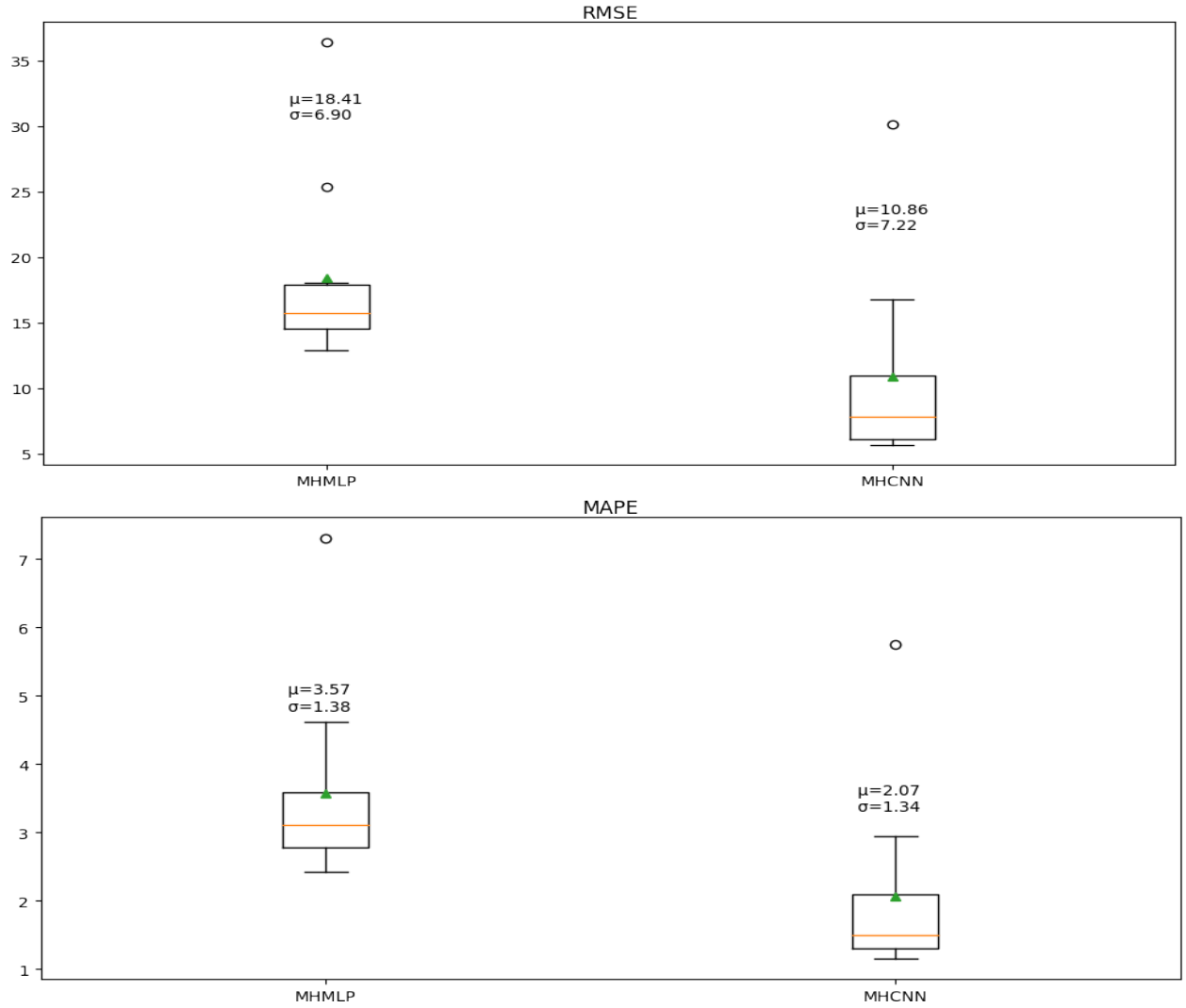

Figure 20- The evaluation of selected models for 1-week prediction, using 10-fold cross validation method

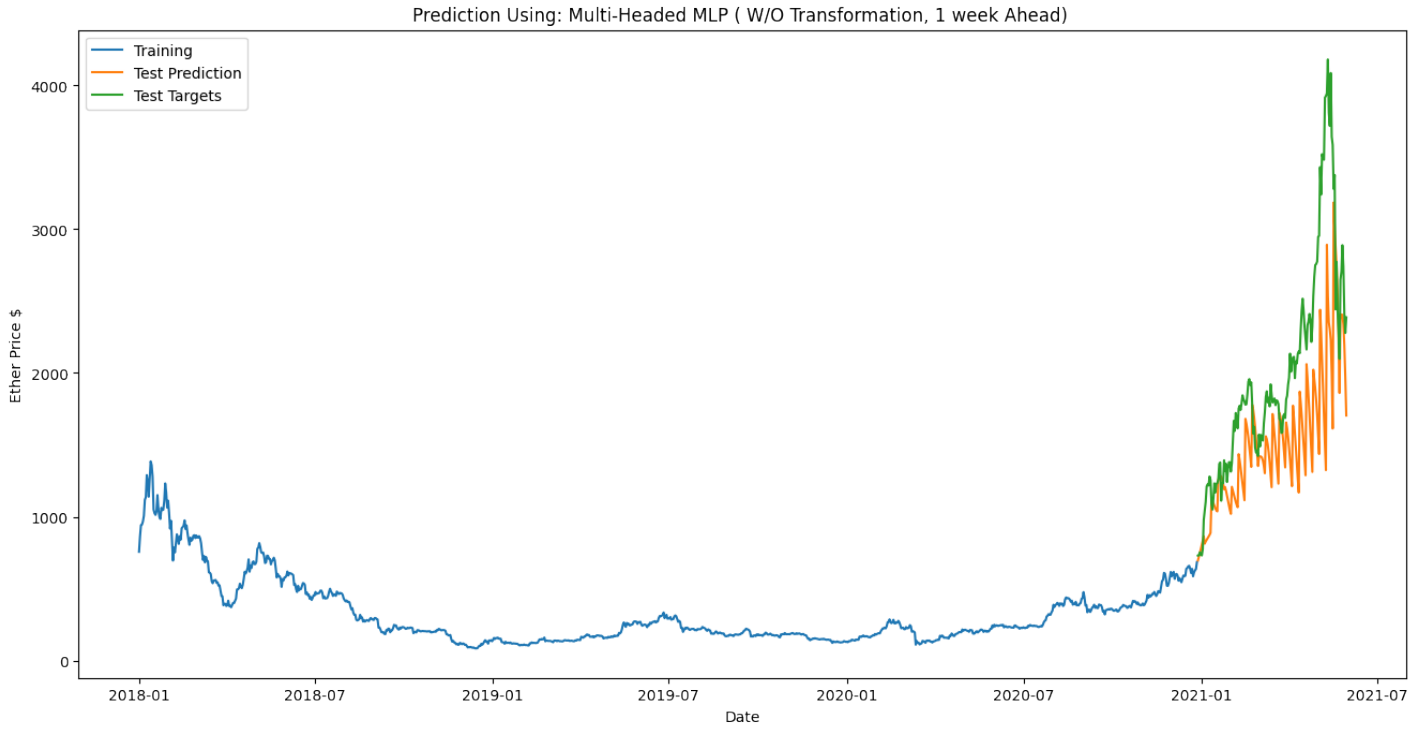

Figure 21- Ether Price prediction on 1-week ahead using MHMLP

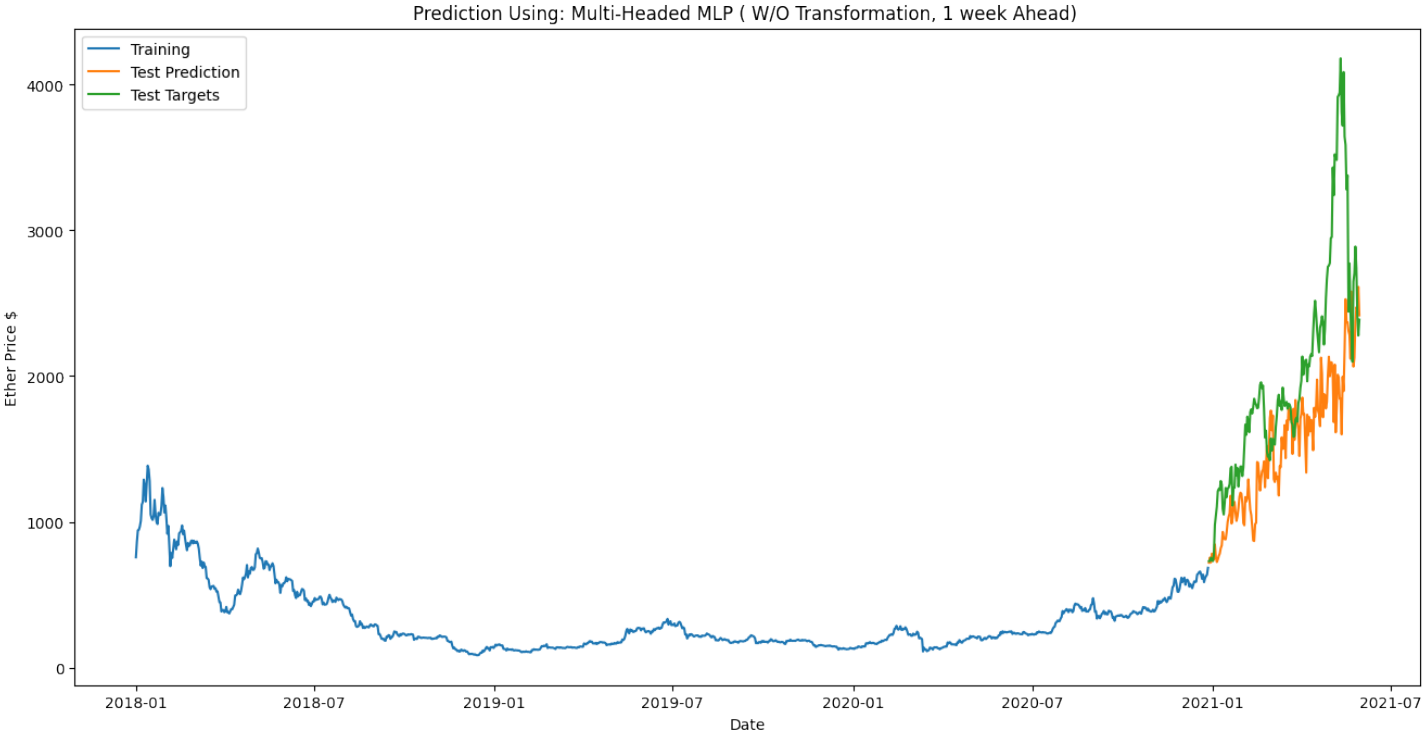

Figure 22- Ether Price prediction on 1-week ahead using MHCNN

As it can be perceived from the charts of price prediction, the volatility in price predictions are more obvious in 1-week forecast. The main reason of this behavior can be the repetition of the

weekly error patterns of the models in all the interval of prediction that act like noise or repetitive fluctuations. In order to extract the main trends and clean the polluted data, one effective way, is the smoothing of weekly forecasts based on the previous patterns. Filters and smoothing methods are effective means to remove noises and the cyclic components. More formally we can say the filter should be capable of isolating fluctuations of the data at a certain frequency. In this study we have used 'rolling mean smoothing', 'simple exponential smoothing' and 'kalman filter' to discard the values that are going out of a decided range and causing unusual fluctuation in the time series.

To apply smoothing methods, the prediction of deep learning models has been combined by these filters. In each step, the predicted prices for the next week will be add to price history and then the whole time series of Ether price will be used to set up a filter to percept the dominant trend of the market. After applying this filter and smoothing the time series, the prices of last week (predicted smoothed prices) will be extracted and reported as predictions. In fact in this process we use both deep learning models and the trend of the price history to obtain more reliable predictions with less fluctuation. To estimate the efficiency of filtering methods, we have applied them to the problem of Ether price prediction. Their results along with the outputs of initial deep learning models and also stacking models are summarized in Figure 23. From this graph we can see that both multi-head MLP (MHMLP) and multi-head CNN (MHCNN) have similar performance with relatively equal mean of RMSE and MAPE, but standard deviation of MHMLP is higher than MHCNN. It can be concluded that the MHCNN model may be more robust than MHCNN and changing the training set have lower effect on predictions. The other important observation from this chart is the slightly better performance of both stacking models (SMLP and SLR) compared to the initial deep learning models. It can a proof of ability of stacking methods to improve the predictions. And finally, the effectiveness of filtering methods on both of deep learning models is completely noticeable in this illustration. While all 3 smoothing algorithms have quite similar results, generally they have resulted in around 25% improvement in the forecasts of each initial deep learning model. To have a better view about the effect of smoothing methods in predicted prices, the forecasts of a deep learning model, with and without filters, are depicted in Figure 24. It's apparent from this graph that smoothing methods are really effective in reducing the fluctuations and extracting the actual trend of the time series.

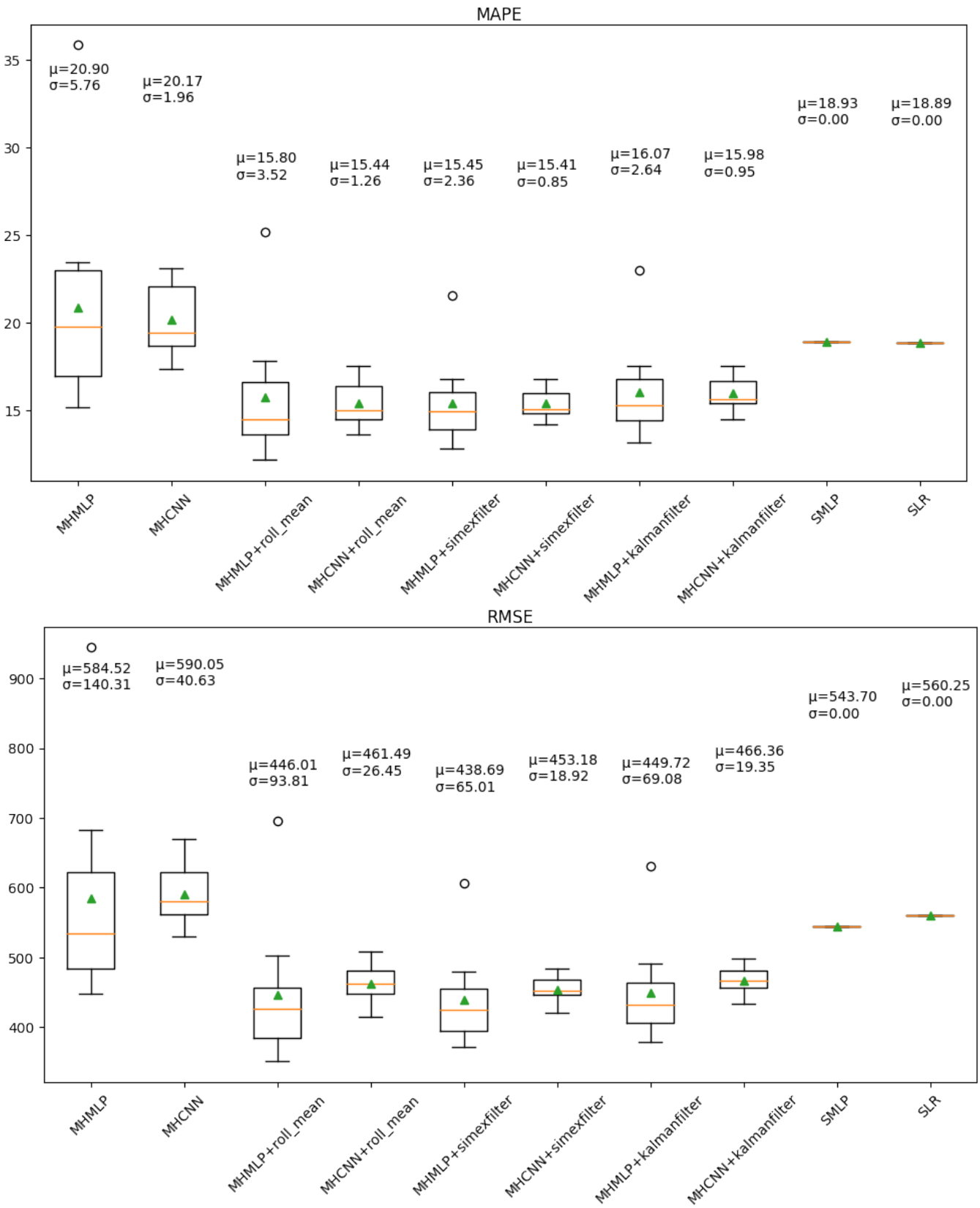

Figure 23- The prediction results of selected deep learning methods, stacking models and filtering techniques

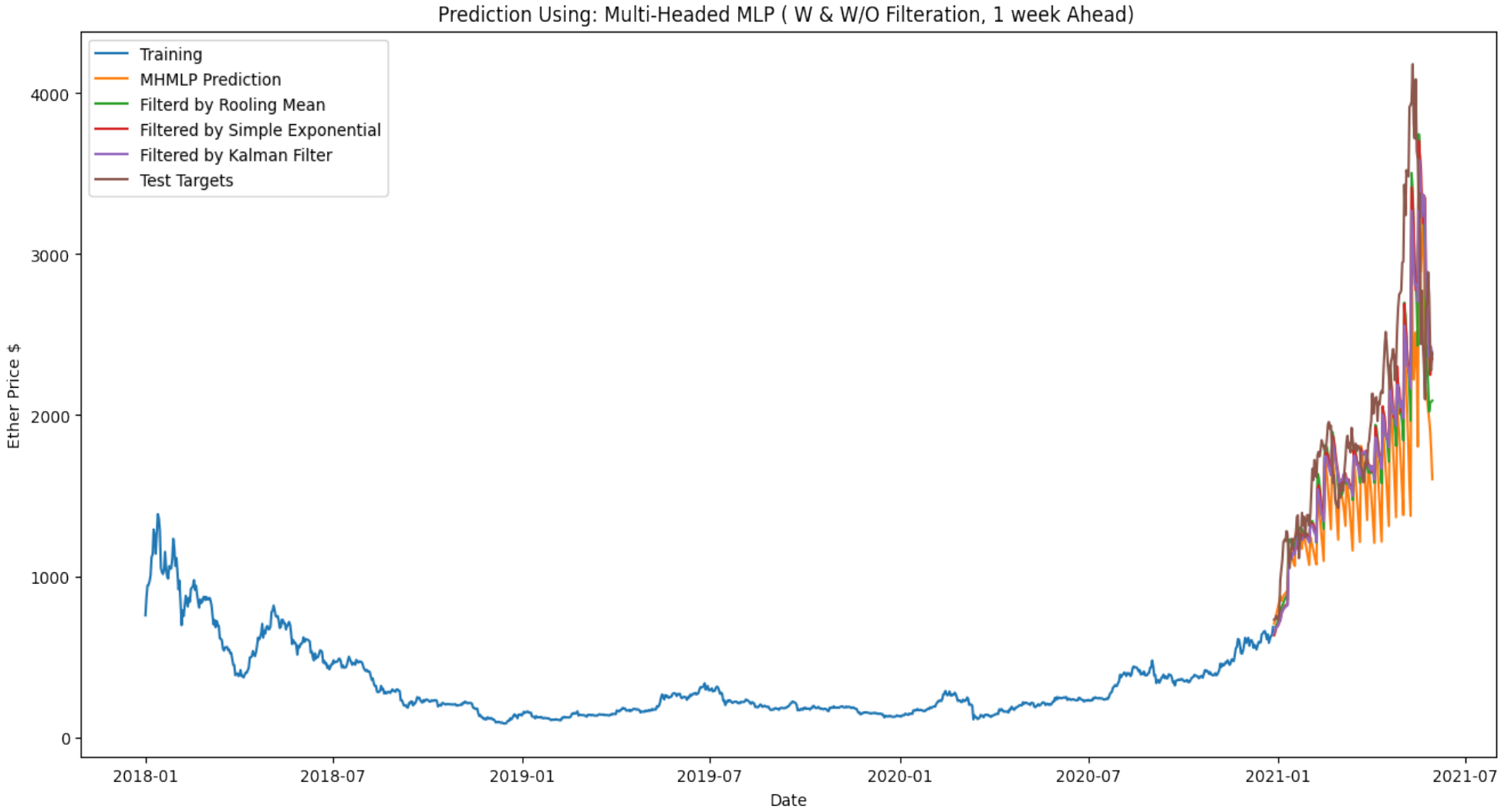

Figure 24- The effect of filtering methods on smoothing the predicted time series of Ether price (predictions are obtained by a realization of multi-head MLP model)

Conclusions and future works

In this study a set of deep learning models including Multi-Layer Perceptron (MLP), Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) were applied in both univariate and multivariate analysis of Ether price. In all models, a time series containing data of 14 days were employed as inputs in order to perform predictions in two different intervals including '1-day' and '1-week'. In multivariate analysis a database consisting of 43 variables like price and trading volume of other major cryptocurrencies, macro-economic indicators, Ethereum network features and social network activities were created and applied in deep learning modeling. Different types of transformations, parameter tuning, stacking and time series filtering have been used to enhance the prediction capability of the models. To sum up the results, it can be stated as follows:

- There are significant differences between k-fold cross validation results and prediction accuracies, because the trend of Ether price in the test interval which has been used for prediction is significantly different with the training interval which has been used for cross validation analysis.
- In '1-day' prediction, plain CNN combined with log transform was the most accurate predictors with around 12% as MAPE.
- Generally, accuracy of prediction for '1-week' interval is lower than for '1-day' interval. The best predictors in this way with about 20% as MAPE, were multi-head MLP and multi-head CNN without having any transformation.
- Stacking methods were applied for both prediction intervals. While in case of '1-day' prediction, stacking couldn't defeat the best model, it could reduce approximately 2% of the MAPE of the best '1-week' predictors.
- In '1-week' prediction, because of the repetition of weekly patterns of model forecasts, a noisy fluctuation along the prediction intervals was created. To remove this noisy variation and extract the main pattern of the market, 3 different smoothing methods including 'rolling mean', 'simple exponential smoothing' and 'kalman-filter' were employed along the deep learning models predictions. Their effective role in improving the performance is completely obvious as they reduce the MAPE of the models from around 20 % to slightly less than 15.5%.

To proceed this subject of study further ahead we suggest to:

- Try to predict the market volatility of the cryptocurrency instead of predicting the price. This will lead to a classification problem that can be a new field to assess the applicability of deep learning methods.
- Other approaches to predict the market behavior like technical analysis, pattern recognition in price charts, or even sentimental analysis through social media are all interesting and novel topics in which we can examine innovative machine learning methods.
- In our study, the approach to tune model hyper-parameters was 'trial and error' and because of time limitations other systematic methods like 'greed search' or 'optimization

algorithms' were not employed. The application of these methods in deep learning improvement, can bring out interesting results.

Bibliography

[1] L. E. Livieris, S. Stavroyiannis, E. Pintelas and P. Pintelas, "A novel validation framework to enhance deep learning models in time-series forecasting," *Neural Computing and Applications,* vol. 32, pp. 17149-17167, 2020, https://doi.org/10.1007/s00521-020-05169-y.

[2] V. Derbentsev, A. Matviychuk and V. Soloviev, "Forecasting of Cryptocurrency Prices Using Machine Learning," in *Advanced Studies of Financial Technologies and Cryptocurrency Markets*, Singapore, Springer, 2020, https://doi.org/10.1007/978-981-15-4498-9_12.

[3] R. Chowdhury, M. A. Rahman, M. S. Rahman and M. Mahdy, "Predicting and Forecasting the Price of Constituents and Index of Cryptocurrency Using Machine Learning," arXiv:1905.08444v1, 2020, 10.1016/j.physa.2020.124569.

[4] E. Chan, Algorithmic Trading: Winning Strategies and Their Rationale, Hoboken, NJ, USA: John Wiley & Sons, 2013.

[5] Y. Ma, B. Yang and Y. Su, "Technical trading index, return predictability and idiosyncratic volatility," *International Review of Economics & Finance,* vol. 69, pp. 879-900, 2020, https://doi.org/10.1016/j.iref.2020.07.006.

[6] P. Jaquart, D. Dann and C. Weinhardt, "Short-term bitcoin market prediction via machine learning," *Short-term bitcoin market prediction via machine learning,* vol. 7, pp. 45-66, 2021.

[7] P. Ciaian, M. Rajcaniova and d. Kancs, "Virtual relationships: Short- and long-run evidence from BitCoin and altcoin markets," *Journal of International Financial Markets, Institutions and Money,* vol. 52, pp. 173-195, 2018, https://doi.org/10.1016/j.intfin.2017.11.001.

[8] A. Dutta, S. Kumar and M. Basu, "A gated recurrent unit approach to bitcoin price prediction," *Journal of Risk and Financial Management,* 2020, https://doi.org/10.3390/jrfm13020023.

[9] P. Jaquart, D. Dann and C. Weinhardt, "Using machine learning to predict short-term movements of the bitcoin market," in *Proceedings of 10th International Workshop on Enterprise Applications, Markets and Services in the Finance Industry*, 2020, https://doi.org/10.1007/978-3-030-64466-6_2..

[10] N. Smuts, "What drives cryptocurrency prices? An investigation of Google trends and telegram sentiment," *ACM SIGMETRICS Performance Evaluation Review,* vol. 46, pp. 131-134, 2019, https://doi.org/10.1145/3308897.3308955.

[11] D. Philippas, H. Rjiba, K. Guesmi and S. Goutte, "Media attention and Bitcoin prices," *Finance Research Letters,* vol. 30, pp. 37-43, 2019, https://doi.org/10.1016/j.frl.2019.03.031.

[12] J. Abraham, D. Higdon, J. Nelson and J. Ibarra, "Cryptocurrency price prediction using tweet volumes and sentiment analysis," *SMU Data Science Review,* vol. 1, no. 3, 2018.

[13] J.-Z. Huang, W. Huang and J. Ni, "Predicting bitcoin returns using high-dimensional technical indicators," *The Journal of Finance and Data Science,* vol. 5, pp. 140-155, 2019, https://doi.org/10.1016/j.jfds.2018.10.001.

[14] D. Shen, A. Urquhart and P. Wang, "Does twitter predict Bitcoin," *Economics Letters,* vol. 174, pp. 118-122, 2019.

[15] Y. B. Kim, J. G. Kim, W. Kim, J. H. Im, T. H. Kim, S. j. Kang and C. H. Kim, "Predicting Fluctuations in Cryptocurrency Transactions Based on User Comments and Replies," *PLOS ONE* , vol. 11, no. 8, 2016, https://doi.org/10.1371/journal.pone.0161197.

[16] M. Mudassir, S. Bennbaia, D. Unal and M. Hammoudeh, "Time-series forecasting of Bitcoin

prices using high-dimensional features: a machine learning approach," *Neural Computing and Applications,* pp. 1-15, 2020, https://doi.org/10.1007/s00521-020-05129-6.

[17] P. M, A. Sharma, V. V, V. Bhardwaj, A. P. Sharma, R. Iqbal and R. Kumar, "Prediction of the price of Ethereum blockchain cryptocurrency in an industrial finance system," *Computers and Electrical Engineering,* vol. 81, 2020, https://doi.org/10.1016/j.compeleceng.2019.106527.

[18] Z. Chen, C. Li and W. Sun, "Bitcoin price prediction using machine learning: an approach to sample dimension engineering," *Journal of Computational and Applied Mathematics,* vol. 365, 2020, https://doi.org/10.1016/j.cam.2019.112395.

[19] Z. Han, J. Zhao, H. Leung, K. F. Ma and W. Wang, "A Review of Deep Learning Models for Time Series Prediction," *IEEE Sensors Journal,* vol. 21, no. 6, pp. 7833-7848, 2021, doi: 10.1109/JSEN.2019.2923982.

[20] L. E. Livieris, N. Kiriakidou, S. Stavroyiannis and P. Pintelas, "An Advanced CNN-LSTM Model for Cryptocurrency Forecasting," *Journal of Electronics,* vol. 10, no. 3, p. 287, 2021, https://doi.org/10.3390/electronics10030287.

[21] E. Pintelas, I. Livieris, S. Stavroyiannis, T. Kotsilieris and P. Pintelas, "Investigating the Problem of Cryptocurrency Price Prediction: A Deep Learning Approach," in *IFIP International Conference on Artificial Intelligence Applications and Innovations*, Berlin/Heidelberg, Germany, 2020.

[22] I. Livieris, E. Pintelas, S. Stavroyiannis and P. Pintelas, "Ensemble Deep Learning Models for Forecasting Cryptocurrency Time-Series.," *Algorithms,* vol. 13, no. 5, 2020, https://doi.org/10.3390/a13050121.

[23] R. Yates and D. Goodman, Probability and Stochastic Processes: A Friendly Introduction for Electrical and Computer Engineers, Hoboken, NJ, USA: John Wiley & Sons, 2014.

[24] D. Sarkar, R. Bali and T. Sharma, Practical Machine Learning With Python, A Problem-Solver's Guide to Building Real-World Intelligent Systems, Berkeley, CA: Apress, 2018, https://doi.org/10.1007/978-1-4842-3207-1.

[25] C. W. J. Granger, "Investigating Causal Relations by Econometric Models and Cross-spectral Methods," *Econometrica,* vol. 37, no. 3, pp. 424-438, 1969, doi:10.2307/1912791.

[26] S. Maitra, "Towards data science," 7 Oct 2019. [Online]. Available: https://towardsdatascience.com/granger-causality-and-vector-auto-regressive-model-for-time-series-forecasting-3226a64889a6.

[27] M. Mudassir, S. Bennbaia, D. Unal and M. Hammoudeh, "Time-series forecasting of Bitcoin prices using high-dimensional features: a machine learning approach," *Neural Computing and Applications,* 2020, https://doi.org/10.1007/s00521-020-05129-6.

[28] A. Dubey, "towards data science," 15 Dec. 2018. [Online]. Available: https://towardsdatascience.com/feature-selection-using-random-forest-26d7b747597f.

[29] U. Kumar, "towards data science," 16 Feb 2021. [Online]. Available: https://towardsdatascience.com/computing-cross-correlation-between-geophysical-time-series-488642be7bf0.

[30] Z. Han, J. Zhao, H. Leung, K. F. Ma and W. Wang, "A Review of Deep Learning Models for Time Series Prediction," *IEEE Sensors Journal,* vol. 21, no. 6, pp. 7833-7848, 2021, doi: 10.1109/JSEN.2019.2923982.

[31] L. Zhang, L. Zhang and B. Du, "Deep learning for remote sensing data: A technical tutorial on the state of the art," *IEEE Geoscience and Remote Sensing Magazine,* vol. 4, no. 2, pp. 22-40,

2016.

[32] Z. Han, Y. Liu, J. Zhao and W. Wang, "Real time prediction for converter gas tank levels based on multi-output least square support vector regressor," *Control Engineering Practice,* vol. 20, no. 12, pp. 1400-1409, 2012.

[33] J. Brownlee, Deep Learning for Time Series Forecasting Predict the Future with MLPs, CNNs and LSTMs in Python, www.machinelearningmastery.com, 2018.