International Journal of Financial Management and Economics

P-ISSN: 2617-9210
E-ISSN: 2617-9229
IJFME 2022; 5(2): 22-29
Received: 28-05-2022
Accepted: 02-07-2022

Georgios Rigopoulos
Department of Economics, University of Athens, Athens, Greece

Corresponding Author:
Georgios Rigopoulos
Department of Economics, University of Athens, Athens, Greece

A long short-term memory algorithm-based approach for univariate time series forecasting with application to GDP forecasting

Georgios Rigopoulos

Abstract
This work presents a time series forecasting method based on Long Short-Term Memory (LSTM) network, which can be utilized for macroeconomic variable forecasting, like Gross Domestic Product. LSTM is a popular method in Artificial Neural Networks and is an active research topic, however applications in forecasting are limited. The current work focuses on one-step ahead forecast, and uses Python Keras libraries for the implementation. The method is applied to forecast Greek Gross Domestic Product and the accuracy results are high and comparable to ARIMA approach. The model we present offers a competent approach for time series and GDP forecasting, with comparable accuracy to traditional statistical approaches. It demonstrates the feasibility of the approach and can be further developed in parameter tuning and application on diverse large data sets.

Keywords: time series forecasting, long short-term memory, gross domestic product

1. Introduction
Accurate forecasting is one of the most challenging and controversial tasks in finance and economics, due to inherent characteristics of time series and the stochastic nature of the underlying processes, albeit highly desirable from professionals and policy makers. Fluctuations of indices and macroeconomic variables is rarely possible to be predicted with high degree of accuracy, and as time horizon increases it makes the forecasting almost impossible. Short term forecasting can be reasonably feasible in some cases, whenever the time series under examination poses some characteristics that allow us to extrapolate. Low levels of forecasting error and high degree of forecasting accuracy are the key requirements of a forecasting method in order to be considered as a valid candidate for usage from stakeholders. Despite the challenges, forecasting remains a very active research domain with substantial developments in the past decades. A diversity of models, either based on plain methods, or combined has been developed and policy makers try to benefit from them, in financial trading or central bank policy making, to name a few examples. Computer science developments, on the other hand, have contributed at a large extent to forecasting research, as many complex theoretical models cannot be used without the substantial computing power of modern computers and novel algorithms. Traditional statistical methodologies have been also further developed with non-linear variants, along with new models originating from data and machine learning domain.

Traditional, well known, linear statistical models, like Autoregressive Integrated Moving Average (ARIMA), have been extensively used in practice, and were dominating the forecasting practice for decades (Box, *et al*., 2015) [1]. Despite their limitations, they are still used for benchmarking. However, new nonlinear models have been developed, like the bilinear model, or the autoregressive conditional heteroscedastic model (ARCH). Those approaches lead to improved accuracy in real life applications and better explanatory power (Engle, 1982) [2]. In the past two decades, a new research stream has been developed, which is based on data and not in prior explicit theoretical modelling. This comprises a variety of machine learning methods, which among other domains, have been used for forecasting and are quite competent to traditional forecasting approaches. In some cases researchers demonstrate that they outperform. Artificial Neural Networks (ANN) is a well-known modelling approach in machine learning and is considered as a very promising alternative for

time series forecasting, capable to uncover structures and inherent patterns, like non-linearity and complexity in time series data (Martinez 2019) [3]. A specific type of ANN, is the Long Short-Term Memory (LSTM) network, which has gained prominent importance, due to its design that allows to include memory characteristics and allows to formulate prediction models and achieve high level of accuracy (Gers, *et al*., 2000; Schmidhuber, 2015) [4, 5]. LSTM has been applied in time series forecasting (Namini, 2018), with results that are comparable to traditional methods, like ARIMA, but works are limited in forecasting macroeconomic variables, like Gross Domestic Product. Another limiting factor in ANN usage for time series forecasting, is the fact that the majority of the methods are not mature enough, and they are not embedded in relevant software, like the traditional statistical ones.

Given the above and considering the limited works on time series forecasting using ANN methods and limited software solutions which support them, this work aims to present a method for time series forecasting with usage of LTSM and demonstrate its accuracy level. The focus of this work is to utilize machine learning for Gross Domestic Product (GDP) forecasting, based on the Long Short-Term Memory (LTSM), a particular type of ANN that is considered as able to capture nonlinear patterns in time series data. The method is presented along with an application in Gross Domestic Product (GDP) forecasting. The novelty of our work lies in the fact that it proposes an approach for GDP forecasting, that is not widely met in literature, is comparable to ARIMA and can be further developed in an automated process. The results look quite promising for this type of time series, however the method can be further developed, in terms of reducing complexity, parameter tuning, and further comparisons to other established statistical or machine learning methods.

The structure of the paper is as follows. Some background on time series and machine learning forecasting approaches is presented initially, and next the proposed LTSM forecasting method is introduced. Then, an illustrative application on GDP forecasting follows, along with explanation and comparison to ARIMA approach. Finally, key findings and future directions are discussed in the conclusion.

2. Background

In this work we focus on univariate time series forecasting using supervised learning. In univariate time series the observable corresponds to one variable, which is measured over time in equally divided time slots. Such observations, and measurements over time, can refer to a diverse type of underlying generating processes, from stock prices and macroeconomic variables to physical phenomena, the theoretical modelling. However, in univariate time series, the traditional statistical based forecasting is based on the assumption that future values of a time series can be estimated from past values, given that the time series exhibits some certain characteristics. So, series modelling focuses not on any exogenous variables, but solely to historical values and the effort is towards fitting the model with the highest accuracy and least prediction error.

Traditional time series forecasting is based either in ordinary regression models, where time indices are used as the independent variables, or on autoregressive moving average models, where the independent variables are the past values and the past prediction errors. The Box and Jenkins method for ARIMA and exponential smoothing are some representative traditional statistical methods for time series forecasting (Gooijer & Hyndman, 2006) [6]. In the last twenty years, there has been a rapid development of methods based on machine learning, which compete the traditional linear or nonlinear statistical based approaches (Ahmed, *et al*., 2010) [7]. Research on Artificial Neural Networks has been combined with finance and time series with some significant results. Researchers, support that some ANN structures, like LSSM can achieve a high level of capturing the dynamic structure over time and perform well in predictions (Chen *et al*., 2015) [8]. In a similar context, some works with financial data support that forecasting time series data with LSSM looks is highly comparable with ARIMA in terms of accuracy (Siami-Namini, & Namin, 2018) [9]. However, limited works present forecasting models for macroeconomic variables using methods like LSTM. Although, machine learning methods are becoming popular for prediction, there exist some deficiencies, like complexity and training (Krauss, 2016) [10].

2.1 Artificial Neural Network (ANN)

An Artificial Neural Network (ANN) is a supervised machine learning algorithm and can be considered as a construction of a set of nodes, which are connected and arranged in layers. In its general form, it can be represented as a directed graph, where the layers are placed in a sequential way and each layer follows the successive ones. Node design is based on the idea that a node receives some input value and generates some output value, based on a set of weights, which are defined for the connection of each pair of nodes, plus some bias value. The output value is generated by a node, utilizing a given activation function, which can be selected prior to the model training. The layer of nodes, which receives the initial input data, is called input layer, and the layer, which generates the final outcome of the ANN is called output layer. Between those two layers, a number of middle layers are placed, which perform the computations and pass the results to the next layer. Those are called hidden layers and can vary in number and number of nodes (Dreyfus, 2005) [11].

Given the above basic ANN conceptual design, the ANN can be built as a multiple layer structure, where the input layer uses the initial input and feeds successive hidden layer, and this is repeated until the output layer. This network, as soon as it is built, can be trained in order to be used for applications in problems like classification and forecasting (Trappenberg, 2019) [12]. The training of the network is an iterative process, where the weights between node pairs are modified in order to adapt the prediction to the expected outcome, minimizing thus the error.

In a more formal way, for a network with n nodes, we can describe the computed outcome $\hat{y}$ of node j, as a linear combination of its input values, plus a bias value, which is converted by an activation function $f(.)$ of a meaningful format, as follows:

$\hat{y} = f(z)$, where the activation function f can be of any type, but usually is either Rectified Linear Activation (ReLU), Logistic (Sigmoid), Hyperbolic Tangent (Tanh), and $z = \sum_{i=1}^{n} w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$+b, with

w_i the weight defined between node j and i and b the bias value.

The above structure, describes the initial approaches followed in ANN, where the information from the input layer is forwarded to the hidden and to the output layers, and are called feed forward networks. The number of layers and nodes can be scaled up and become large, increasing the computational effort accordingly. Such network can be trained in cycles, which are called epochs, and they refer to a training cycle with the entire training dataset. The epoch includes a forward pass, the calculation of the output value, and a backward pass, the calculation of the loss function and adjustment of weights. The number of epochs is a hyperparameter of the algorithm and can be defined and optimized, as it affects the training quality.

2.2 Long Short-Term Memory (LSTM)

A more advanced type of ANN is the Recurrent Neural Network (RNN), which uses input values from the present step but it also uses input values from the previous step, so we if we add the time dimension, we can consider that it keeps information relevant to subsequent time steps. This way, RNN can be considered as appropriate network that can be used in data with sequential nature, like time series and can be utilized for forecasting. In contrast to feed forward networks, where inputs and outputs are independent, in a RNN there is a memory capability, which makes the network perform well in forecasting. In theory, RNN can use information from long sequences, but in practice it has some deficiencies in the back propagation of time series data. This is known as the exploding or vanishing gradient problem, where terms may reach zero or infinity exponentially. It results to cost function calculation problem, which in turn reduces prediction accuracy and network efficiency. A solution to the problem is utilization of optimizers like Stochastic Gradient Decent (SGD), which can calculate the cost function for parts of the dataset and not all the training examples and speeds up the process.

The simple RNN, however, are not very efficient for long term prediction, so more advanced approaches, based on RNN model, have been introduced. The Long Short-Term Memory (LSTM) network is a special kind of RNN and is a very efficient network, explicitly designed to handle long term dependencies as a default process and not a learning task. The design is more complex compared to RNN, and is based on the concept of a set of cells, where each cell has several components to resemble memory features. A cell has three inputs, the input or activation from the previous step or time period, the new data value at current time period and the memory input from the previous step or time period. Inside the cell, the input data are transformed by activation functions, which are called gates, and comprise the place where operations are actually being performed. The gates are the input, the forget, the update and the output, each one performing a specific operation, with forget gate being a key one, as it determines whether the cell will keep a value transferred from previous time steps or it will forget an include it in the next step. This approach is fitting very efficiently with long term dependencies.

3. LSTM forecasting model

In this work we focus on univariate time series forecasting, and especially in Gross Domestic Product, where we assume that there is some level of dependency among observations in prior steps. Given that this assumption holds, the series dataset can be modelled by a Recurrent Neural Network. However, as said before, RNN is not so efficient for handling dependencies for a long period of time, while LSTM, a special type of RNN, can hold and learn more efficiently from long sequences of observations. So LSTM is considered as a more appropriate model and is utilized to model a rolling forecast approach, where, in general, a new forecast is generated when a new observation is added. The proposed model was implemented in Python using Keras libraries. Keras is an open source Python library and provides a wide set of functions for neural networks and machine learning in general.

To use LSTM for univariate time series forecasting, we introduce the following process, which comprises three phases:

A. Model definition phase: In this phase we define the parameters of the model and all the required functions or methods we are going to use. This is an initial definition, and it cannot be optimal at first, but is based on best practices and past empirical findings. The steps are the following:

1. **Split dataset in test and training sets:** Initially, the dataset is split into training and test subsets, at a ratio (a common approach is 70:30) and the order of observations in not altered, as the sequence of observations is key characteristic of the series.
2. **Normalize dataset:** Next, to normalize the input values, a scaler is selected. The scaler, standardizes the values (with $\mu = 0$ and $\sigma = 1$), and transforms the observations in values within the interval [0, 1]. After the forecasting with LSTM, the values are rescaled with the inverse normalization.
3. **Sliding window reshape:** For a supervised learning algorithm, like LSTM, the time series data need to be restructured in a form that can be used for input the algorithm. A further pre-processing step is needed then. The approach followed is the sliding window, where the input is the previous time step and the output is the next time step. In this approach, the previous time steps are the lag size or the window width or order.
4. **Define parameters**

a. **Number of layers:** A basic model comprises two layers. One LSTM layer and one Dense. The LSTM layer is a series of cells with gates (input, forget, update and output), which, as described previously, calculate pointwise operations (addition and multiplication) on the gates values to generate the output for the Dense layer. The Dense layer is the layer where the computations take place to generate output values. It computes the output utilizing the weights, the activation function and the memory.

b. **Number of neurons:** The number of neurons in each layer needs to be specified. There is no unique method for this, but it depends on the problem and the empirical findings. For a basic model, we can use one neuron for the dense layer and one for the LSTM layer.

c. **Epoch's number:** The number of epochs defines the number of training cycles using the full training dataset. It can vary based on the batch size of the data and the problem.

d. **Loss function:** A widely used loss function is the Mean Square Error, which is widely used in time series forecasting.

e. **Optimizer selection:** The optimizer selection is needed in order to minimize the loss function. Adam optimizer, is a typical selection as it is capable of handling large datasets and is appropriate to use with non-stationary data.
f. **Window size:** This parameter is mainly empirically selected, as it depends on the time series data. A too short or too long size would either add noise, or do not capture dependencies.

5. **Define regularization:** The dropout method is used in order to reduce overfitting, by randomly selecting cells and set their value to zero. This approach improves LSTM efficiency.
6. **Optimize hyperparameters:** To achieve the optimal values of the hyperparameters, we begin with some initial values and an iterative process is followed to check for their suitability. The criterion is the loss function minimization and forecast accuracy.
7. **Define validation data set:** To test the model accuracy we use a percentage of the training set as validation set to test the parameter values.
8. **Define assessment metric:** RMSE is a popular measure and it measures the residuals between the actual and predicted values. Its major characteristic is that it penalizes the large errors and it also uses the same units with the forecast values.

B. **Model fitting phase:** In this phase we train and build the LSTM model. This phase comprises the following steps:

1. **Initialization:** We use the parameter values, as defined previously, the training set, creates the layers (LSTM, Dense). Initial random weights are set at layer nodes. A low bias value is also assigned initially.
2. **Building:** The model is built and compiled with the selected loss function and optimizer function.
3. **Training:** The model is trained until it reaches the pre-defined number of epochs. During the training phase, the loss function is compared and training losses are considered. If there is no substantial improvement for a number of consecutive iterations, the training stops.

C. **Model validation:** In this phase the model is evaluated, according to training and validation errors, and the loss function. This is combined with benchmarking and forecasting accuracy comparison. Based on the assessment, some parameters can be tuned and the entire process can be repeated.

4. Application in GDP forecasting

Following the above generic LSTM model, we applied the process in a univariate time series forecasting problem, where the time series is the Greek GDP. The motivation for this approach, is the limited works on GDP forecasting using LSTM and on the other hand its potential and comparable accuracy to ARIMA and k-Nearest Neighbor models (Rigopoulos, 2022) [13]. We follow the steps presented previously, using Python and Keras Library.

To build the time series for the Greek GDP, we used publicly available data from the World Bank database in constant USD2015 values for the period 1971-2020.The line plot below (Fig. 1) depicts the evolution of GDP through this period (in USD millions). It is obvious that there is an increasing trend until 2005 and a decreasing trend after this, indicating a non-stationary process. We applied the LSTM model in both the original series, as well as the time series after taking the second difference (Fig. 2), which results in a stationary series (Rigopoulos, 2022) [14], in order to compare the LSTM accuracy. The process we followed is detailed below and the parameters provided, are the ones that resulted into an optimal accuracy, after executing several experiments.

Fig 1: Greek GDP in millions (constant $2015)

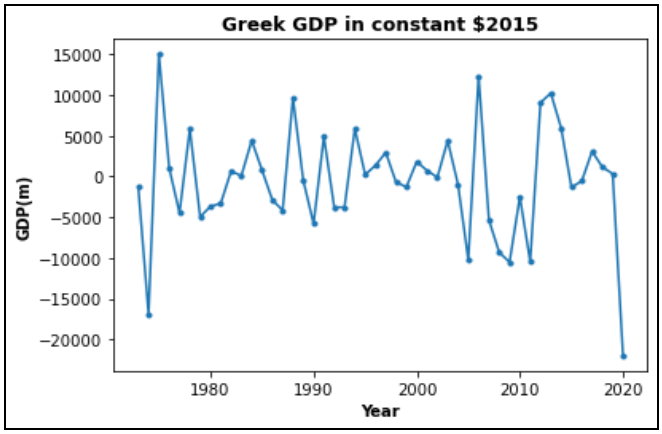

Fig 2: Second difference of Greek GDP in millions (constant $2015)

A. **Model definition phase:** We defined the parameters at an initial level, using some experimentation with the dataset and following default values:

1. **Split dataset in test and training sets:** We split the dataset into training and test subsets, at a ratio (a common approach is 70:30) not altering the order of observations.
2. **Normalize dataset:** We use the min-max scaler for this. It standardizes the values (with $\mu = 0$ and $\sigma = 1$), and transforms the observations in values within the interval [0, 1]. After the forecasting with LSTM, we rescale the values with the inverse min-max normalization.
3. **Sliding window reshape:** For the specific series, we selected a window of four, given the low number of observations and not seasonality.
4. **Define parameters:**

a. **Number of layers:** We use two layers. One LSTM layer and one Dense.
b. **Number of neurons:** For the model, we can use one neuron for the dense layer and four for the LSTM layer.
c. **Epoch's number:** We define the number of epochs to 50. After each epoch, the loss function is computed, and if there is not a change in ten subsequent epochs, the

iteration terminates.

d. **Loss function:** As loss function we use the Mean Square Error.
e. **Optimizer selection:** Adam optimizer is selected.
f. **Window size:** The window size is defined to three.

5. **Define regularization:** The dropout method is used in order to reduce overfitting, by randomly selecting cells and set their value to zero.
6. **Optimize hyperparameters:** We use loss function minimization and forecast accuracy to review the results.
7. **Define validation data set:** A 10% of the training set is used as validation set. to test the parameter values.
8. **Define assessment metric:** RMSE is used as a metric for assessment.

B. **Model fitting phase:** We trained and built the LSTM model:

1. **Initialization:** We used the parameter values, as defined previously, the training set, creates the layers (LSTM, Dense).
2. **Building:** The model was built and compiled with the selected loss function and optimizer function.
3. **Training:** The model was trained until it reaches the pre-defined number of epochs, or there was no further improvement in the loss function.

C. **Model validation:** We performed a series of experiments with different values of hyperparameters and the models were examined for forecasting accuracy. Based on the assessment the entire process was repeated and parameters were tuned, until the results that are provided below.

5. Results and Discussion

In this section, we present the results of the LSTM process application for Greek GDP forecasting for the two time series, the initial (non-stationary) and the second differences (stationary). The parameters we used were identical in all repetitions, in order to be able to compare the outcomes. In Table 1, we present the RMSE values of LSRTM model for both datasets, and we also include the values from work in the same dataset, using ARIMA and k Nearest Neighbor methods (Rigopoulos, 2022) [14]. The RMSE values indicate that the proposed LSTM method is quite competitive to ARIMA method and outperforms the k-NN method. Even if more elaborate experiments in financial time series (Namini, & Namin, 2018) [15] suggest that LSTM outperforms ARIMA, in our case, we see that ARIMA performs better. This is related to the time series on the one hand, and on the other hand on the scope of the present work. Our aim is to introduce the method for GPP forecasting and tune the parameters at a sufficient level. Optimizing the network to outperform ARIMA is not a straightforward process and it depends on the time series. Another aspect of the results, is that the method performs higher in stationary series.

Table 1: RMSE values of various forecasting models

	RMSE
LSTM on Initial dataset	17669.988278
LSTM on Second difference	12341.161289
ARIMA	8589.909
k-NN	23316.767881

In Figures 3 and 4, we can see the training performance of LSTM on the two datasets, and it in figures 5 and 6, we can see one-step ahead forecasts for the two datasets, as well. Those results, depict the relatively low accuracy level, but reasonably acceptable, but there is a number of elements in the LSTM process, that can be considered for improvement.

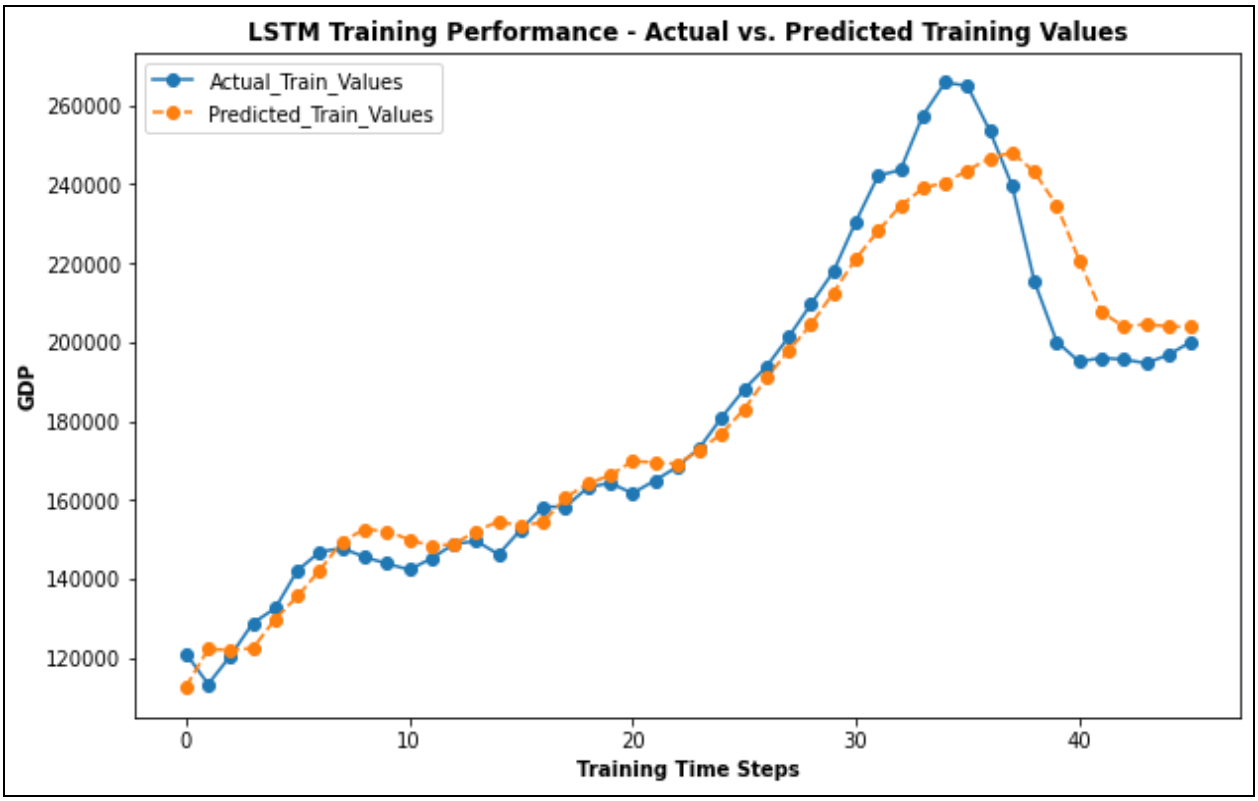

Fig 3: Training performance on initial dataset

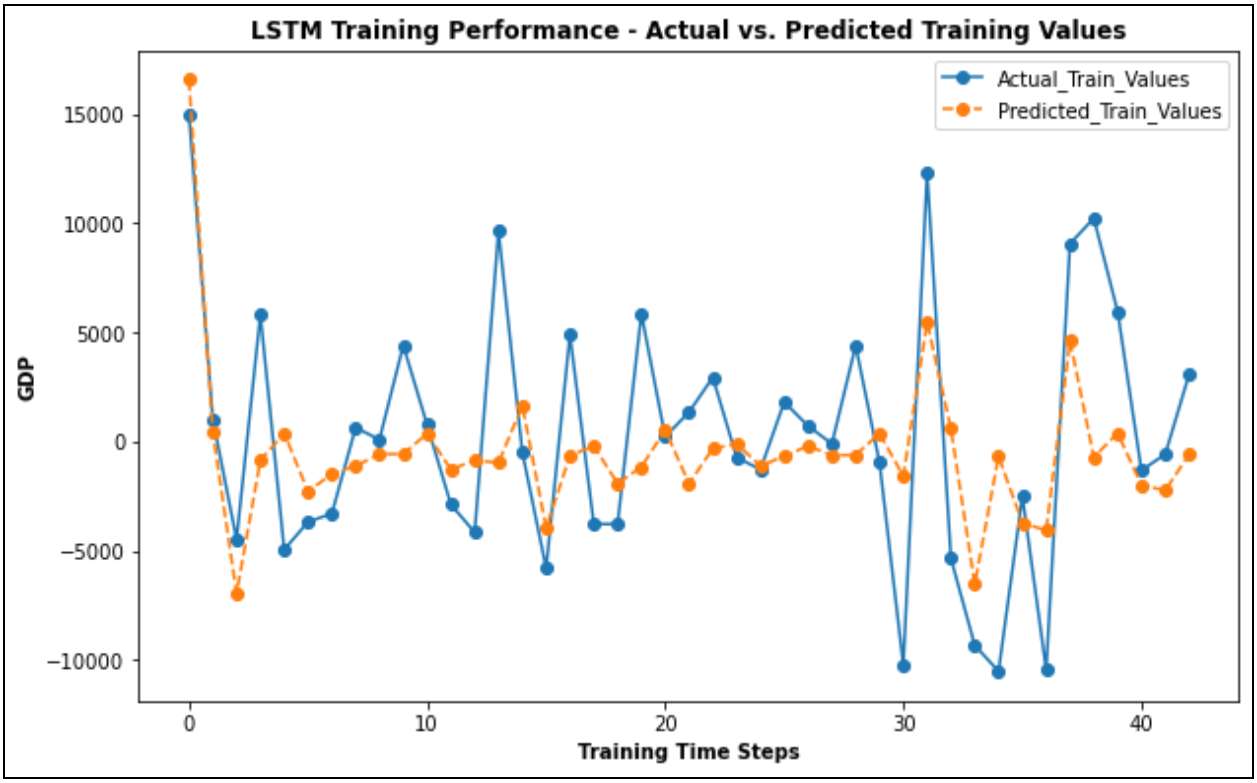

Fig 4: Training performance on the second difference dataset

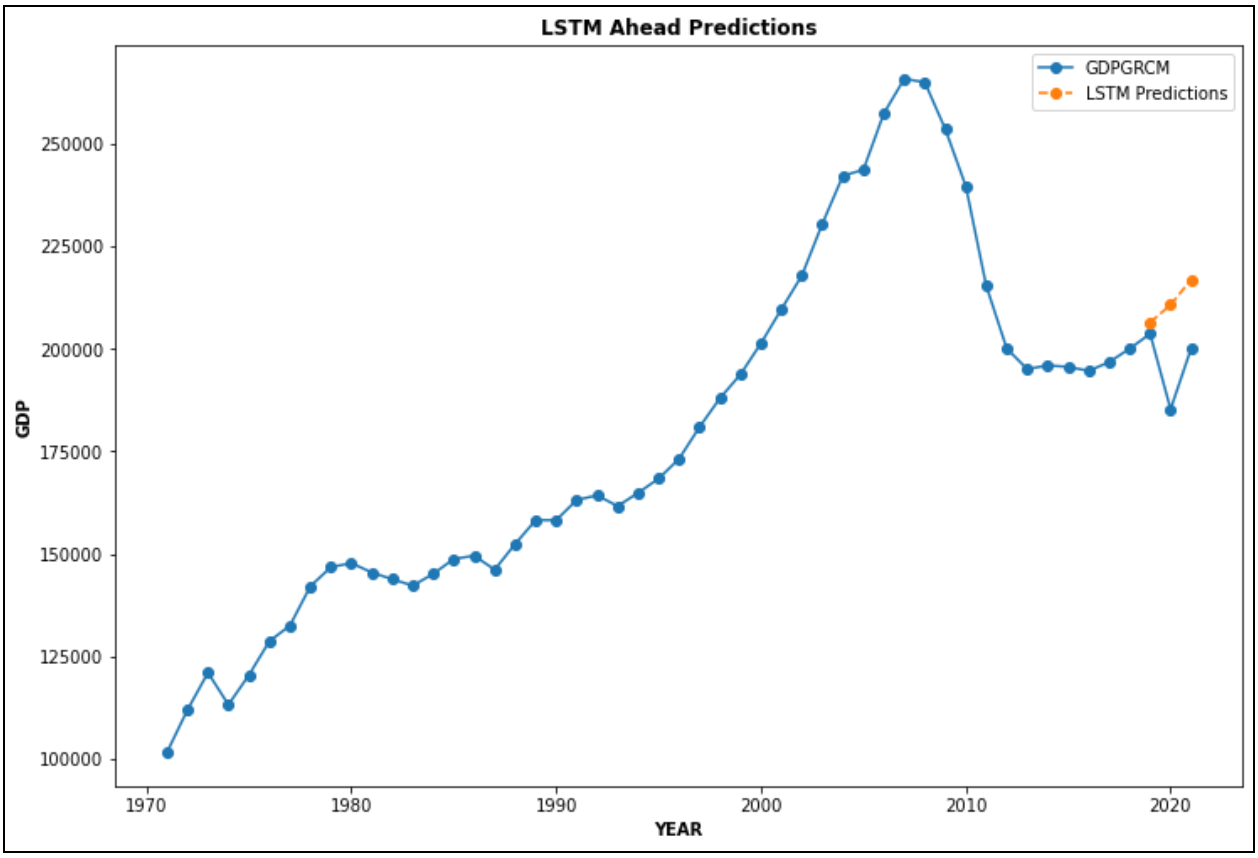

Fig 5: One step ahead forecasting on initial dataset

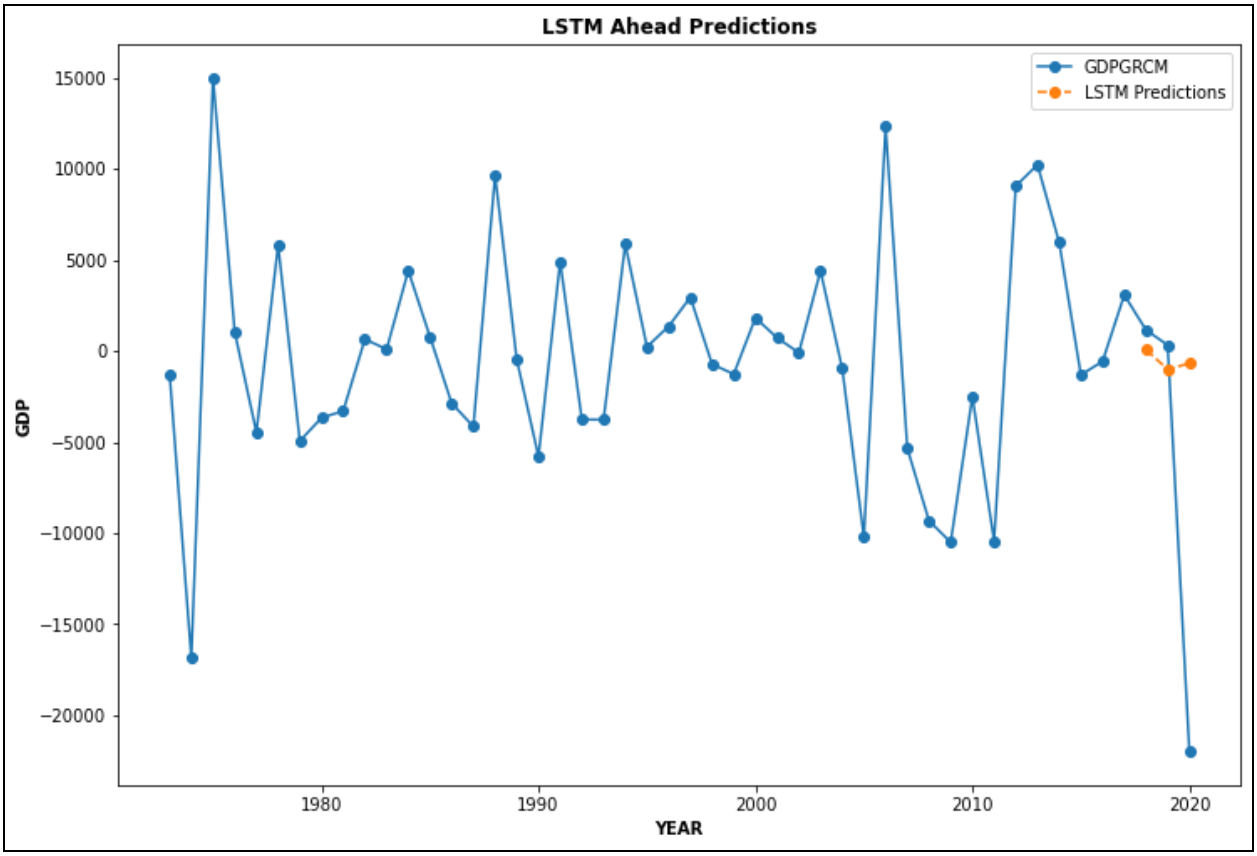

Fig 6: One step ahead forecasting on second difference dataset

As LSTM requires a number of hyperparameters to be optimal, it is not always to tune them in the way that optimizes the RMSE in an automatic way. On the other hand, each dataset has its own characteristics and a general formula cannot be applied. A variety of option can be followed in order to achieve the best outcome, including, epochs number, window size, activation functions among the rest of the parameters.

6. Conclusion

The model presented is an initial approach to model GDP using LSTM method, and can be improved in future work at parameter tuning. Also, further assessment with larger and more diverse datasets is needed, as well as comparison with established models, like ARIMA and GARCH. However, the findings from this work demonstrate the feasibility of the approach and the comparable accuracy to traditional statistical approaches.

7. References

1. Box GE, Jenkins GM, Reinsel GC, Ljung GM. Time series analysis: forecasting and control. John Wiley & Sons; c2015.
2. Engle RF. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. Econometrica. 1982;50(4):987-1007.
3. Martínez F, Frías MP, Pérez MD, Rivera AJ. A methodology for applying k-nearest neighbor to time series forecasting. Artificial Intelligence Review; c2019. 52(3).
4. Gers FA, Schmidhuber J, Cummins F. Learning to Forget: Continual Prediction with LSTM, in Neural Computation. 2000;12(10):2451-2471.
5. Schmidhuber J. Deep learning in neural networks: An Overview, in Neural Networks. 2015;61:85-117.
6. De Gooijer JG, Hyndman RJ. 25 years of time series forecasting. International Journal of Forecasting. 2006;22(3):443-473.
7. Ahmed NK, Atiya AF, El-Gayar N, El-Shishiny H. An empirical comparison of machine learning models for time series forecasting. Econometric Reviews. 2010 Aug;29(5-6):594-621.
8. Kai Chen, Yi Zhou, Fangyan Dai. A LSTM-based method for stock returns prediction: A case study of China stock market. In: IEEE International Conference on Big Data (Big Data); c2015. p. 2823-2824.
9. Sima Siami-Namini, Akbar Siami Namin. Forecasting Economics and Financial Time Series: ARIMA vs. LSTM; c2018. In: ArXiv abs/1803.06386.
10. Krauss XA, Do N. Huck, Deep neural networks, gradient boosted trees, random forests: Statistical arbitrage on the S&P 500, FAU Discussion Papers in Economics 03/2016, Friedrich-Alexander University Erlangen-Nuremberg, Institute for Economics; c2016.
11. Dreyfus G. Modeling with Neural Networks: Principles and Model Design Methodology. Springer Berlin Heidelberg; c2005. p. 85-201.
12. Thomas P. Trappenberg. Neural networks and Keras. Oxford University Press; c2019.
13. Rigopoulos G. GDP Modeling Using Autoregressive Integrated Moving Average (ARIMA): A Case for Greek GDP, International Journal of Business Marketing and Management (IJBMM). 2022 July-Aug;7(4):66-75. ISSN: 2456-4559.
14. Rigopoulos G. Univariate Time Series Forecasting Using k-Nearest Neighbors Algorithm: A Case for GDP, International Journal of Scientific Research and

Management (IJSRM), 2022, 10(8).
15. Siami-Namini S, Tavakoli N, Namin AS. A comparison of ARIMA and LSTM in forecasting time series. In 17th IEEE international conference on machine learning and applications (ICMLA). IEEE; c2018. p. 1394-1401.