

RESEARCH ARTICLE

Deep learning models for inflation forecasting

Alexandre Fernandes Theoharidis¹  | Diogo Abry Guillén² | Hedibert Lopes³

¹Inspere Institute of Education and Research and Compass Group, São Paulo, Brazil

²Inspere Institute of Education and Research and The Central Bank of Brazil, São Paulo, Brazil

³Inspere Institute of Education and Research, São Paulo, Brazil

Correspondence

Alexandre Fernandes Theoharidis, Inspere Institute of Education and Research and Compass Group, São Paulo, Brazil.
Email: alex.theoharidis@gmail.com

[Correction added on 26 April 2023, after first online publication: The reference citations have been corrected.]

Abstract

We propose a hybrid deep learning model that merges Variational Autoencoders and Convolutional LSTM Networks (VAE-ConvLSTM) to forecast inflation. Using a public macroeconomic database that comprises 134 monthly US time series from January 1978 to December 2019, the proposed model is compared against several popular econometric and machine learning benchmarks, including Ridge regression, LASSO regression, Random Forests, Bayesian methods, VECM, and multilayer perceptron. We find that VAE-ConvLSTM outperforms the competing models in terms of consistency and out-of-sample performance. The robustness of such conclusion is ensured via cross-validation and Monte-Carlo simulations using different training, validation, and test samples. Our results suggest that macroeconomic forecasting could take advantage of deep learning models when tackling nonlinearities and nonstationarity, potentially delivering superior performance in comparison to traditional econometric approaches based on linear, stationary models.

KEYWORDS

autoencoders, convolutional networks, deep learning, inflation forecasting, LSTM networks, machine learning

1 | INTRODUCTION

In modern macroeconomics, the relevance of inflation forecasting cannot be overstated, given its prominent role in many practical situations. For instance, the estimation of DSGE models, which are pervasively employed by central banks in their decisions regarding monetary policy, requires a thorough understanding of inflation dynamics, without which it becomes unfeasible to derive the links with other macroeconomic variables and, thus, make accurate predictions. Additionally, inflation forecasts are crucial for firms when assessing the profitability of long-term investments. Finally, banks and households also rely on such predictions when celebrating contracts set in nominal values, such as debts.

As the pertaining literature shows, forecasting inflation is challenging and no consensus exists regarding which is the best econometric approach; see References 1 and 2 for a comprehensive discussion. Undoubtedly, improving upon simple univariate econometric models is daunting due to several factors. The main hindrance lies on the nonlinear dynamics displayed by inflation, undermining the use of the standard linear Phillips Curve, despite its theoretical appeal. Supportive evidence is reported by References 3 and 4, among others.

In this paper, we delve into two important aspects of the problem. First, with respect to nonlinearities, many of its sources have been identified and documented, such as nominal rigidity,⁵ zero lower bound for interest rates,⁶ economic uncertainty,⁷ and fixed costs.⁸ Second, the choice of variables which can be systematically used for prediction,

yielding reliable out-of-sample forecasts, is also paramount. In the era of big data, multiple options are promptly available. Without tools and criteria to filter them and achieve parsimony, one becomes prone to data mining biases and overfitting.¹

In an attempt to address the aforementioned obstacles, this work investigates whether deep learning methods can generate more accurate out-of-sample inflation forecasts than standard models reviewed in the related literature. The selection of deep learning to overcome the limitations of previous approaches reflects the encouraging and abundant findings in real situations where nonlinearities are ubiquitous and input variables abound, such as pattern classification,¹⁰ speech recognition,¹¹ and image processing;¹² see References 13 and 14 for additional examples.

Despite the potential of deep learning, applications to inflation modeling are scarce. In a seminal work,⁸ we were among the first to survey and carefully examine several machine learning models, evaluating their performance in forecasting inflation, but focusing on standard models (with random forest outperforming), even though recent advances in deep learning have brought forth a myriad of promising architectures for time series modeling, which are yet to be meticulously tested and appraised in the context of forecasting macroeconomic variables². Previously, authors such as References 19, 20, and 21 have focused solely on assessing individual models against straightforward benchmarks.

Among these alternative architectures, it is worth mentioning LSTM (Long Short-Term Memory) networks, a versatile deep learning model that has been successfully applied in sequential processing such as textual interpretation. Introduced by²² and extended by References 23, 24, and 25, LSTM networks form a special subset of RNN (Recurrent Neural Networks) whose architecture is well-adapted to capture temporal dependencies in data. This feature explains their power for text processing and similar tasks.

In time series modeling, where the dependent variable is assumed to be explained by its past values and other independent variables, LSTM networks should be suitable. Here, a particular variation of LSTM network introduced by Reference 26, called Convolutional LSTM network, or ConvLSTM for short, is the focus due to its desirable flexibility and ability to extract spatiotemporal features from the inputs. Empirical studies using ConvLSTM and yielding encouraging results are provided by References 26-29, and 30.

In Macroeconomics, not only the literature lacks an assessment of the performance of recent advances in forecasting models based on machine learning, but also no consensus regarding the superiority of nonlinear models exists. For instance,³¹ argue that accounting for nonlinearities does not necessarily provide statistical gains when forecasting the US CPI. Meanwhile,³² conclude that, although machine learning models are more accurate under certain conditions, more parsimonious time series models may outperform in some scenarios. Thus, additional investigation is justified, increasing the value of our paper.

Furthermore, deep learning methods can also be used for dimension reduction and denoising. An example is the autoencoder, which is a type of neural network trained in an unsupervised manner for data encoding and decoding. When applied to time series, it can be seen as roughly equivalent to a nonlinear version of Principal Component Analysis (PCA). However, as Reference 33 argue, unlike PCA, autoencoders can also detect repetitive structure in the data. In various domains, these networks have achieved positive results, outperforming alternative dimension reduction techniques; see References 33 and 34. For the purpose of dimension reduction and denoising, we employ the Variational Autoencoder (VAE); see Reference 35.

Therefore, in a nutshell, this paper presents a deep learning approach for forecasting inflation accurately. Namely, we combine ConvLSTM networks and VAEs with this objective, and the resulting model (VAE-ConvLSTM) is estimated using the dataset compiled by Reference 9. Our results demonstrate that the proposed model is superior to several benchmarks in terms of out-of-sample accuracy for multiple forecasting periods and that deep learning models should be considered when forecasting inflation.

The contributions of this paper are threefold. First, by scrutinizing deep learning methods and applications in inflation forecasting, there is an opportunity to advance the literature related to the use of machine learning for macroeconomic prediction, which is a burgeoning and ever-changing field of research. In particular, it is possible to assess the potential of LSTM networks and their extensions, namely ConvLSTM, in modeling macroeconomic time series through the development of a novel deep learning model. For the preceding reasons, it is plausible to believe that deep learning could yield

¹As illustration, in an effort to compile and standardize macroeconomic time series for academic research,⁹ provide a monthly database comprised by more than 100 variables that can be neatly applied to forecast inflation, as Reference 8 and us show. However, it is conceivable that not all of these variables are beneficial for out-of-sample prediction.

²For demonstration, multiple examples, with positive results, can be found in the surveys conducted by References 15-17, and 18. Thus, a study of more modern approaches and their application for inflation forecasting is justified.

compelling results in several contexts that demand forecasting macroeconomic variables in general, without constraining itself to inflation.

Second, it is expected that, by using deep learning to model inflation, some knowledge regarding the sources of nonlinearities within this variable can be acquired. As previously contended, there is still no consensus with respect to the statistical gains of incorporating nonlinearities in models used for inflation forecasting. Actually, through the application of autoencoders, it becomes feasible to inspect the factor structure in the macroeconomic dataset to diagnosis whether nonlinear interactions are existent. Consequently, the present research may shed some light in these questions as well.

Finally, by examining denoising and dimension reduction techniques, this work intends to measure the value of data preprocessing for macroeconomic forecasting so as to avoid data mining biases when time series are plentiful. Reference 36 coined the term *data-rich environment* precisely to describe these situations where both the number of variables and the length of time series are large and close to each other. It is reasonable to believe that, for macroeconomic time series, which are typically observed in low frequency, generating few observations, this topic is critical.

2 | METHODS

In this section, we present a novel approach to improve inflation forecasting. For the sake of brevity, we do not discuss well-known topics relating to the fundamentals of inflation, trivial linear econometric models, nor rudimentary, readily available machine learning models. For a comprehensive overview of these methods in a time series application, we defer to Reference 8, who investigate and compare each of them in terms of out-of-sample performance.

2.1 | The VAE-ConvLSTM model

Recently, several empirical papers have reported encouraging results regarding the use of deep learning techniques in problems involving time series forecasting. This success is explained mainly by the fact that neural networks are capable of capturing nonlinearities in the input data, enhancing their forecasting performance. Some studies illustrating the power and versatility of those models are provided by Reference 37, who apply a MLP (Multilayer Perceptron) to forecast water demand;³⁸ employing MLPs to model exchange rates;³⁹ who combine LSTM networks and GARCH models to analyze the volatility of selected stock indices;⁴⁰ modeling nonstationary time series through ANNs (Artificial Neural Networks);²⁶ who utilize ConvLSTM to predict rainfall;³⁰ and applying ConvLSTM to forecast energy generation. The interested reader may find exhaustive surveys on the subject in the References 14-16, and 17.

The central idea of the model proposed herein is to build on the adaptability and robustness of LSTM networks in capturing temporal dependencies to formulate a model that is well-suited to address the nonlinearities observed in inflation time series, as verified by References 5,41, and 42. LSTM networks are a reasonable building block since they are designed to model time-dependent data, succeeding in multiple tasks. For instance, References 11,43,44 provide applications for comprehension of handwritten sentences, speech recognition, and text classification, respectively. Time series analysis with LSTM networks is discussed by Reference 39 and others.

So as to improve the forecasting capabilities of the LSTM networks, convolutional networks are added to take advantage of hierarchical patterns that may exist in data, creating the ConvLSTM architecture as proposed by Reference 26. Finally, VAEs are included for dimension reduction, forming an architecture henceforth called VAE-ConvLSTM. The forecasting process and the full network, with every layer, are shown in Figures 1 and 2. The architecture and hyperparameters are debated subsequently.

The core references that inspired the development of this model are References 26-28 and 30. The positive results provided therein when applying ConvLSTM and similar architectures in multiple forecasting contexts encouraged the present work. In that sense, as argued and demonstrated by Reference 26, who are credited for the idealization of

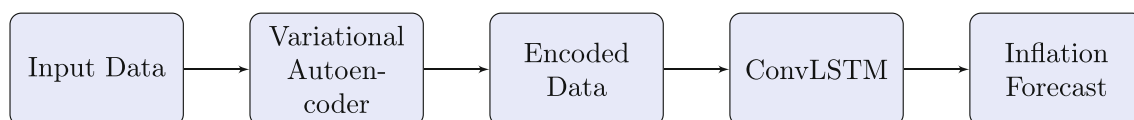


FIGURE 1 Flowchart exhibiting the proposed forecasting process.

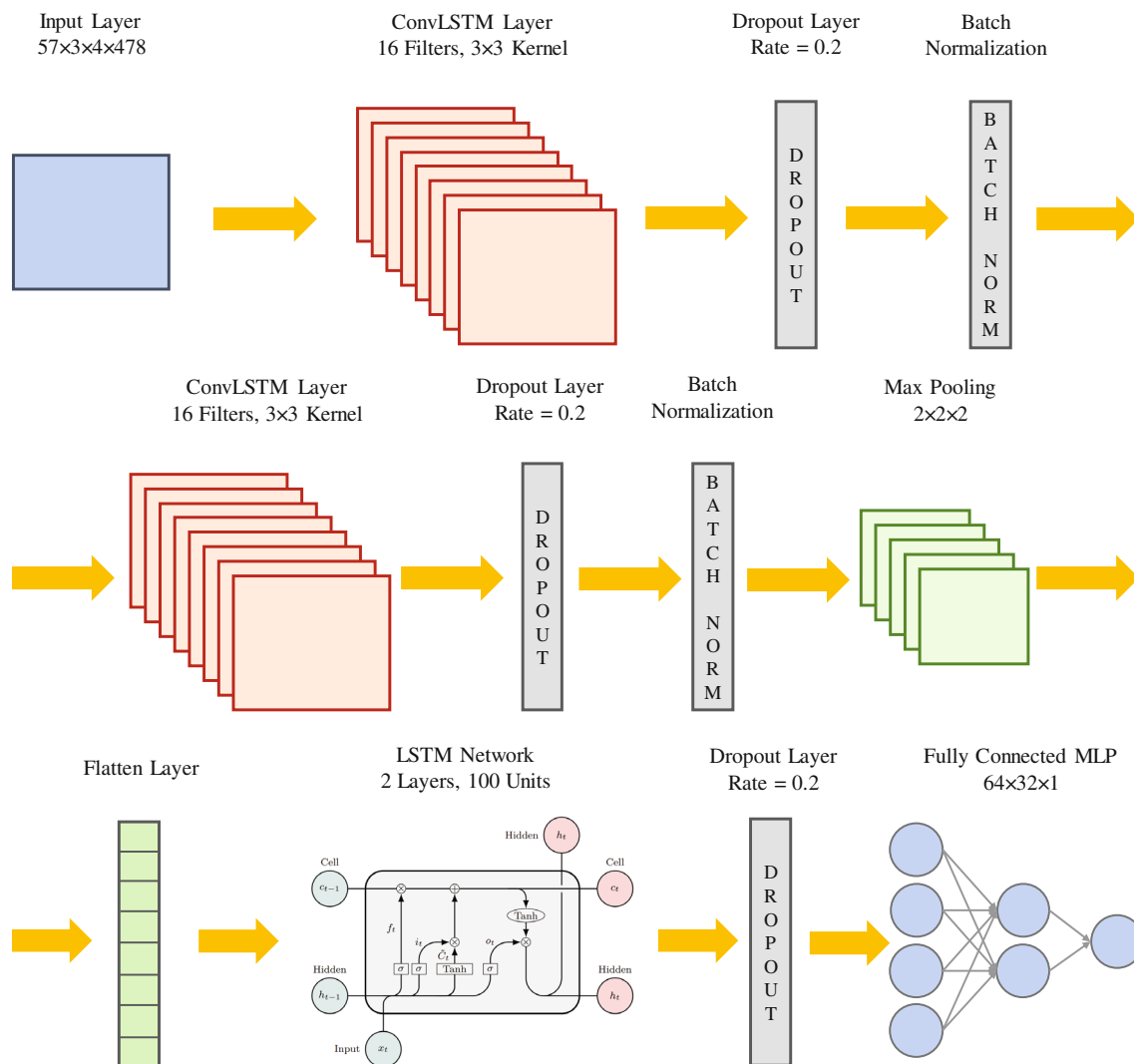


FIGURE 2 Flowchart representing the layers of the convolutional LSTM (ConvLSTM) model. The input layer receives the encoded data supplied by the variational autoencoders (VAE; a $57 \times 3 \times 4 \times 478$ tensor) and transfers the data to a sequence of ConvLSTM, dropout (with a 0.2 dropout rate), and batch normalization layers. The convolutional layers have 16 filters and 3×3 kernel (filter size). Next, a 3D max pooling layer ($2 \times 2 \times 2$) summarizes the data, which is then flattened and inserted in a long short-term memory (LSTM) network formed by two layers with 100 units each, whose output is processed by a fully connected deep multilayer perceptron (MLP), generating the inflation forecasts.

ConvLSTM, the choice of this version over the original LSTM network is justified by the fact that, although the latter has proven powerful when handling temporal serial correlation, it contains undesirable redundancy for spatial data. In that sense, experiments show that ConvLSTM networks are more effective at capturing spatiotemporal correlations, consistently outperforming LSTM networks.

Moreover, with the accelerated development and gradual maturity of deep learning, some practitioners began to realize that the local connection and global sharing features of convolutional neural networks can greatly diminish the required parameters and training time of the model. This approach is explored by Reference 45 in financial trading, where the authors select 15 distinct technical indicators and compute their values for a 15-day period to convert price series into 2D images, which are later processed by a deep CNN.

It is worth emphasizing that the choice of a hybrid model that merges convolutional and LSTM networks is not only supported by the empirical results reported in the literature regarding the successful applications of ConvLSTM for time series modeling and forecasting. Supplementary studies show that, in general, hybrid models tend to deliver more robust performance than a single model in several situations. For instance, Reference 46 demonstrate the effectiveness of utilizing CNNs and LSTMs to classify scene images with multi-views and multi-levels of abstraction. This combination

outperforms several state-of-the-art methods in their experiments. In distinct contexts, similar results are provided by Reference 47, who merge ANNs and GARCH models to predict the price return volatility of gold spot and future prices; by Reference 48, who combine genetic algorithms and ANNs for groundwater level prediction; and by Reference 49, whose model for time series forecasting stems from a hybridization of artificial neural networks and ARIMA models.

Mathematically, a *ConvLSTM network* can be formulated as:

$$i_t = \sigma(w_{xi} \times x_t + w_{hi} \times h_{t-1} + w_{ci} \circ c_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma(w_{xf} \times x_t + w_{hf} \times h_{t-1} + w_{cf} \circ c_{t-1} + b_f) \quad (2)$$

$$o_t = \sigma(w_{xo} \times x_t + w_{ho} \times h_{t-1} + w_{co} \circ c_{t-1} + b_o) \quad (3)$$

$$\tilde{c}_t = \tanh(w_{xc} \times x_t + w_{hc} \times h_{t-1} + b_c) \quad (4)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \quad (5)$$

$$h_t = o_t \circ \tanh(c_t) \quad (6)$$

where σ is the sigmoid function, and w_{xi} , w_{xf} , w_{xo} , w_{xc} , w_{hi} , w_{hf} , w_{ho} , and w_{hc} are 2D convolution kernels. The input x_t , the cell state c_t , the hidden state h_t , the forget gate f_t , the input gate i_t , and the output gate o_t are all 3D tensors. The symbol \times denotes the convolution operator, and \circ is the Hadamard product. The main difference between a conventional LSTM network and ConvLSTM lies on the convolutions carried out when updating states and gates in the latter architecture. This structure allows the network to be more effective at capturing spatiotemporal correlations, consistently outperforming standard LSTM networks; see Reference 26.

In the network proposed, VAEs serve to reduce the dimension of the input data, since the dataset provided by Reference 9, which is employed to fit the ConvLSTM model as discussed subsequently, contains several highly correlated time series, implying that some redundancy is expected. As an illustration of the relevance of dimension reduction, in a real-time forecasting exercise similar to ours,⁵⁰ find that factors extracted from as few as 40 pre-screened series often yield satisfactory or even better results than using all the 147 series available in that occasion. Besides, weighting the data based on their properties when conceiving the factors also lead to more accurate forecasts, and VAEs have been successfully applied to transform, encode, and extract features from time series, as shown by Reference 51, among others.

Therefore, the architecture begins with a variational autoencoder acting analogously to a nonlinear PCA, mixing the original series and producing a smaller dataset that can be used to model and forecast inflation. The comparison is justified by the fact that, similarly to the conventional PCA, autoencoders allow to extract principal components by encoding data for later decoding with the intent of replicating the original data. The nonlinear aspect stems from the fact that, unlike PCA, the components found are not linearly uncorrelated. Such feature is not a drawback, for neural networks are designed to handle and learn from these nonlinearities.

Moving forward, the encoded data conceived by the VAE is transferred to the ConvLSTM network. The flowcharts in Figure 2 unveil the inner layers of the architecture, displaying the existence of:

1. Two ConvLSTM layers, which are responsible for extracting spatiotemporal features and transforming the time series, generating almost a timeline that shows when different features appear in the time series. Each of these are immediately followed by dropout and batch normalization layers, both of which have the purpose of improving out-of-sample accuracy, as argued by References 52 and 13. The $57 \times 3 \times 4 \times 478$ tensor these layers receive from the encoding layer of the VAE network follows the number of encoded variables (which resemble principal components) obtained from the dataset (57), the spatiotemporal organization of the lags (12 lags arranged in 3×4 dimensions to be read by the convolutional layers³) of each encoded variable, and the number of inflation observations (478) included, adjusted by the number of lags and sample sizes;
2. A summarizing max pooling layer that receives as input the outputs of the convolutional layers. By definition, its purpose is to transform the output of the previous layer prior to letting the data continue to flow through the network. In practical terms, the pooling function replaces the output at a certain location with a summary statistic computed

³Of all the configurations tested, this one produced the most satisfactory results. It must be highlighted that the conclusions of this paper are not sensitive to this data arrangement.

using nearby outputs. The consequence is that the representation becomes almost invariant to small translations of the input, which is a desirable property whenever the presence of a certain feature is more important than its position in the data. The $2 \times 2 \times 2$ configuration used in our architecture was determined in an ad hoc manner and delivered attractive results;

3. A flatten layer that converts tensors into vectors that can be read sequentially by a LSTM cell;
4. A conventional, simple, stacked LSTM network that receives the flattened tensors and captures temporal dependencies. These LSTM layers also encompass dropout layers (adding them once again here improved the out-of-sample performance in our tests); and
5. A MLP that condenses the output of the LSTM network into a inflation forecast.

2.2 | Network training and hyperparameter selection

As highlighted by Reference 13, the central challenge in machine learning is that models must perform reasonably well on new, previously unseen inputs. A well-designed training strategy is decisive to attain this objective⁴. A popular and effective solution is to split the input data into training and test samples, implemented as follows. First, 20% of the observations were reserved for testing. Of the remaining data, 90% of the observations, or 72% of the complete dataset, were applied for training and the rest, for validation. These percentages match the guidelines available in the literature.

Since we are dealing with a time-series modeling problem, we implemented *k*-Fold Cross-Validation for increased reliability, as advocated by References 54 and 55. The approach entails randomly dividing the set of observations used for fitting the model into *k* groups, or folds, of equal size. The first fold is treated as a validation set, and the model is fitted on the remaining (*k* - 1) folds. Typically, *k*-fold cross-validation is carried out using *k* = 5 or *k* = 10, as these values have been shown empirically to yield test error rate estimates that suffer neither from excessively high bias nor from unacceptably high variance. For this study, *k* = 10 has been adopted.

Furthermore, Monte Carlo simulation supplemented cross-validation so as to enhance the reliability of the empirical analysis. More specifically, the splitting of the dataset into training, validation, and test sets, plus the *k*-Fold Cross-Validation, have been repeated 100 times with different, randomly-generated seeds, yielding distinct splits of the dataset each time. This strategy was deemed important because, despite the power of cross-validation, the test set remains unchanged during the process. Simulation also increased the number of times the out-of-sample performance of each model was measured, allowing the computation of confidence intervals for these metrics.

When dealing with time series, the preceding splitting process must be adapted. Indeed, random splits will certainly disrupt the autocorrelation pattern of the series, rendering them unusable for fitting any model. That is, when splitting the samples, the observation of each variable must be accompanied by its corresponding lags. If samples are randomized and split without grouping together observations and the corresponding lags, it would be impossible to know, for a given date, the observations that precede the ones associated with this particular date and we could run the risk of incurring in look-ahead bias.

An alternative method employed here involves running the procedure in blocks or, equivalently, including the lags of the time series as new variables, which was the option chosen. This way, a split does not break the autocorrelations, for every observation of a given variable in a particular instant will be accompanied by the lagged observations of this same variable. To illustrate how the procedure works, one should think the inputs as organized in a table with observed values ordered by date. The columns contain the variables. Since lags of each variable also appear as input variables, for a given date, the corresponding line in the table contains not only the observed values of each variable, but also the corresponding lagged values necessary to estimate inflation, as required by our model and the benchmarks. Here, we included the statistically significant lags as indicated by the autocorrelation functions.

Another consideration is the integration of the training procedures of the VAEs and the ConvLSTM. In this initial version, although there could be gains integrating the training of these two networks, we decided to follow a more naive approach and train the networks independently. That is, the encoded data is produced according to its ability to recover the input data without considering its predictive power for inflation forecasting. Despite the potential improvement stemming

⁴Regarding the selection of the optimization algorithm for training, ADAM, together with Nesterov momentum and a quadratic loss function, has also generated the best performance out-of-sample in the context of inflation forecasting, justifying their selection when fitting the neural networks models considered here. Nesterov momentum accelerates the convergence of gradient-based algorithms, explaining its inclusion; see Reference 53 for further details.

from this integration, the current results are already promising, explaining why the exploration of this training strategy is left for future studies.

Inspired by the configurations and hyperparameters tested by Reference 45, 16 filters and a filter (kernel) size of 3×3 provide the convolution capability with closest neighbors' (upper, lower, right, left, upper left, upper right, lower left, and lower right) information while processing the current layer. Thus, sharp variations within the matrix can be captured and a decent number of lags will be assessed each time the filter is run (i.e., recent past inflation is included, but older observation is discarded). In our experience, these values delivered a satisfactory performance and the sensitivity to them seemed manageable and negligible.

It is worth mentioning that, although there is no clear rule in the literature guiding how to optimize the filter size, few filters cannot infer enough features to improve the learning ability of the network. On the other hand, an excessive number of filters may decrease the efficiency of the feature extraction process, since too much data will be analyzed simultaneously and, thus, the filter may become blurry and fail to identify the underlying features. Also, adhering to the best practices in the literature, zero padding is adopted, meaning that the size of the input is automatically adjusted to avoid shrinking the spatial extent of the network rapidly and/or using small kernels, harming the generalization power of the network; see Reference 13.

In the model proposed for inflation forecasting, dropout layers have been added exclusively after the ConvLSTM and the LSTM layers. This decision is justified by the fact that the core of the forecasting process happens at those stages. The succeeding MLP layers essentially convert their output into a inflation estimate, but the knowledge is concentrated in the previous engines. However, to test this reasoning, other configurations have been implemented, with additional dropout layers, and the results were virtually the same in terms of out-of-sample performance. Moreover, the calibration of the dropout rate was carried out empirically, since there is no consensus in the literature and the optimal choice varies according to the context considered. For the purposes of this paper, a value of 0.2 performed reasonably well, and results did not appear to be overly sensitive to this hyperparameter.

The number of LSTM units, hidden layers, and nodes per layer in the MLP (see Figure 2) were defined in an ad hoc manner. Since there was no clear guidance in the literature for the specific problem we are trying to solve, we tested a few configurations and settled for the one that yielded adequate results. Here, we did not notice any hypersensitivity. Hence, although more efficient optimization approaches could be designed to select these hyperparameters, this improvement can be left for future investigations without penalizing the conclusions of our study.

Moreover, it is appropriate to discuss the influence of the number of epochs and the batch size in the optimization process. Unfortunately, no consensus exists in the literature regarding adequate values for these hyperparameters. They must be calibrated in an *ad hoc* way, usually choosing as criteria the performance derived out-of-sample, that is, the generalization power of the network. In the simulations carried out in this work, the batch size was defaulted to 32, a value in the range of 32 to 512 frequently considered in practice to balance the trade-off between convergence and generalization power; see Reference 56. It should be noted that batch sizes are usually defined as powers of 2 to offer better run time.

Finally, the computational implementation was carried out in Python. A GitHub repository was created for public access and the code is freely available to anyone interested⁵. We invite the readers to check the code to have a more thorough view on how the forecasting process works and how the hyperparameters influence the architecture.

2.3 | Benchmarks

A broad, diverse, and fundamentally justifiable collection of benchmarks is crucial for the appraisal of forecasting models. In an effort to accomplish this objective, a set of 25 benchmarks has been elected for relative performance analysis, as listed below. These benchmarks have been extensively employed in macroeconomic modeling and forecasting and/or in the machine learning literature, justifying their choice.

1. Random walk (RW);
2. Ridge regression with cross-validation (Ridge CV).
3. Bayesian Ridge regression (BRidge);
4. LASSO regression with cross-validation (LASSO CV);
5. Bayesian LASSO (BLASSO);
6. Elastic net with cross-validation (Enet CV);

⁵Available at: https://github.com/AlexandreFT31/Machine_Learning_Inflation

7. Support vector regression (SVR);
8. Random Forests (RF);
9. BART;
10. Bagging;
11. K-Nearest Neighbors regression (kNN);
12. Robust regression with Huber loss (Huber);
13. Theil-Sen robust regression (Theil-Sen);
14. Factor models (Factors);
15. GARCH model;
16. Vector error correction model (VECM);
17. SETAR model;
18. Moving average model, based on Reference 57 (MA);
19. Seasonal ARIMA (SARIMA);
20. Fractional ARIMA (ARFIMA);
21. Gradient boosting (GradBoost);
22. AdaBoost (AdaBoost);
23. Bayesian regression (Bayes Reg.);
24. Multilayer perceptron (MLP);
25. Standard LSTM network (LSTM).

With respect to the standard, plain-vanilla LSTM network that appears in the list, other versions were thought as potential benchmarks, such as GRU (Gated Recurrent Unit) or BiLSTM (Bidirectional LSTM). However, these networks lack the additional capabilities introduced by the convolutional part of the VAE-ConvLSTM architecture and have an architecture similar to the LSTM. Also, they offer mixed improvements over the original LSTM; see Reference 58, for instance. Thereby, to keep the list succinct and focus on the comparison of deep learning methods versus usual econometric and machine learning approaches, we left them out of the comparison.

Finally, the criteria adopted to compare the performance of each model is comprised by four metrics computed for the set of out-of-sample predictions. These metrics are well established in the machine learning literature and many empirical studies resort to them as a means to compare different models. The metrics are: mean squared error (MSE), mean absolute error (MAE), mean absolute percentage error (MAPE), and R^2 . Intrinsicly, the theoretical distribution of these performance metrics, although unknown, most certainly depart from the normal distribution. A method to effectively overcome the aforementioned challenge entails *kernel density estimation* (KDE). KDE was implemented here using the popular Gaussian Kernel and a bandwidth following the rule-of-thumb provided by Reference 59. Hypothesis testing was carried out using a significance level of 5%.

3 | DATA

In this section, the database used to fit the model and the benchmarks is detailed and general descriptive statistics are presented.

3.1 | Database for model fitting

The database employed was provided by Reference 9. The authors maintain in their website⁶ a public macroeconomic database comprising 134 monthly US time series freely available at FRED (Federal Reserve Economic Data, which belongs to the Federal Reserve Bank of St. Louis). These time series had to be supplemented by the PMI (Purchasing Managers' Index) compiled by the Institute for Supply Management (ISM), which, at the time of this publication, is no longer available at FRED. The variables are grouped in the following categories: (1) Output and income (17 time series); (2) Labor market (32 time series); (3) Housing (10 time series); (4) Consumption, orders, and inventories (13 time series); (5) Money and credit (14 time series); (6) Interest and exchange rates (22 time series); (7) Prices (21 time series); and (8) Stock market (5 time series).

⁶<https://research.stlouisfed.org/econ/mccracken/fred-databases/>. Last access: 28 December 2020.

The selection of this database is justified for multiple reasons. Primordially,⁹ implemented the best practices reported in the literature to design a database convenient for empirical analysis that requires big data. Hence, it seems appropriate for the intent of this work. Furthermore, this data has been extensively used in the literature in similar studies; for example, see Reference 8. Finally, the time series provided are lengthy, covering several decades and economic cycles, thus being appropriate for the performance comparison between multiple models avoiding potential biases when short periods of analysis are used.

For the purposes of this work, the period analyzed ranges from January 1978 to December 2019, amounting to 504 observations, which is a size deemed significantly superior to the number of variables contained in the dataset and is sufficient for model estimation with controlled errors. In this time period, there are no missing observations to be treated. Inflation is gauged by US Consumer Price Index (CPI) for All Urban Consumers, which is a measure of the average monthly change in the price for goods and services paid by urban consumers in United States between any two periods. Also, each time series was subjected to the corresponding transformation suggested by Reference 9. Generally, the transformations encompass computing the log values when economically justified and differentiating the series (once or twice, depending on the case) to avoid unit roots and other statistical issues.

3.2 | Descriptive statistics

As a preliminary background before presenting the results, we describe the empirical properties of the inflation time series. The first difference of the log prices is exhibited in Figure 3. The profile suggests that, after the transformation, no persistence remains and, thus, the series does not have unit roots, which can be confirmed via the Augmented Dickey-Fuller (ADF) test. Also, inflation is nonstationary due to time-varying volatility, which is shown in Figure 4 and can also be confirmed through a hypothesis test.

Moreover, as the Q-Q plot in Figure 5 shows, the hypothesis of normality may be immediately rejected (although not provided here, a Jarque-Bera test also corroborate this inference). Indeed, extreme observations have been reported far more often than implied by the normal distribution, signaling that the true distribution must have positive excess kurtosis, consistently with Reference 60. These observation have typically occurred during turbulent periods, such as the late 1970s, when the US economy suffered from high and indomitable inflation, and the 2008–2009 financial crisis.

The autocorrelation and partial autocorrelation functions of the first difference of the log inflation are plotted in Figures 6 and 7. Both plots show that lags have predictive power to explain current inflation. There are no apparent evidence of long-term memory in the series, despite the fact that the autocorrelation function decays somewhat slowly and several lags are statistically significant.

Since Reference 9 advise in favor of the use of the second difference of the log prices, the respective autocorrelation and partial autocorrelation functions are also displayed in Figures 8 and 9. This additional differencing corrects the autocorrelation function, which now decays exponentially, leaving few statistically significant lags. The partial autocorrelation function remains well-behaved.

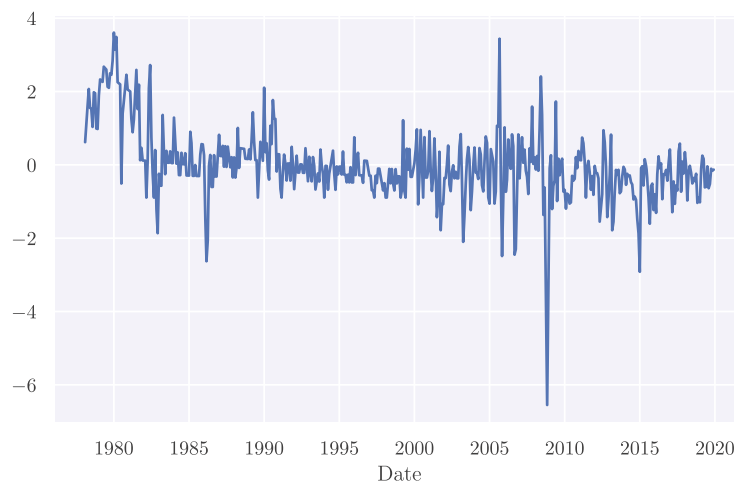


FIGURE 3 Normalized first difference of the log prices, as measured by the US CPI.

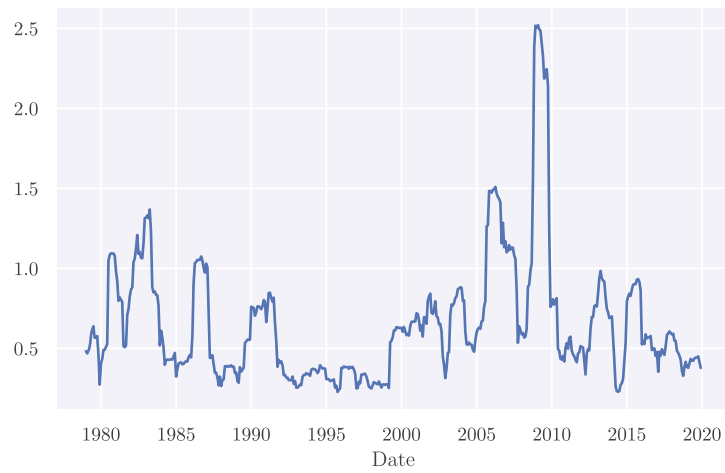


FIGURE 4 Standard deviation of the first difference of the log prices, as measured by the US CPI. Computed using a 12-month rolling window.

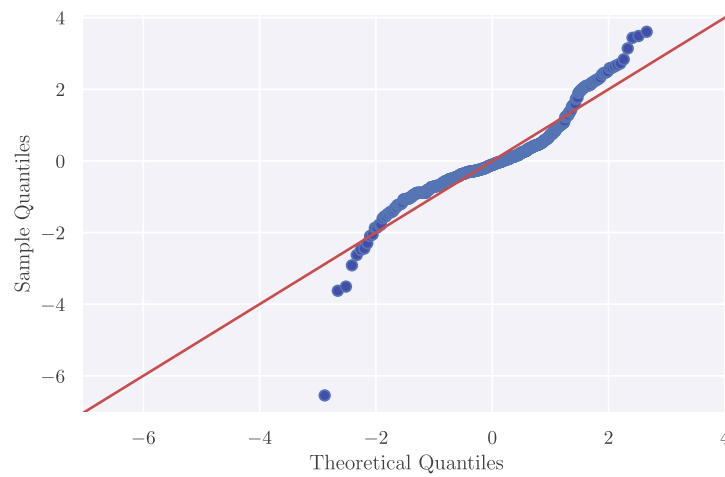


FIGURE 5 Q-Q plot of the normalized first difference of the log prices, as measured by the US CPI.

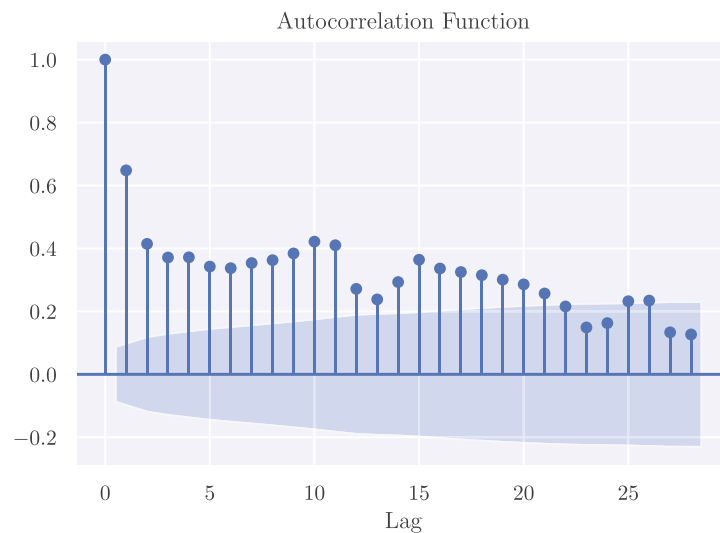


FIGURE 6 Autocorrelation function of the first difference of the log prices, as measured by the US CPI. Confidence interval (shaded area) is computed using Bartlett's formula with a significance level of 5%.

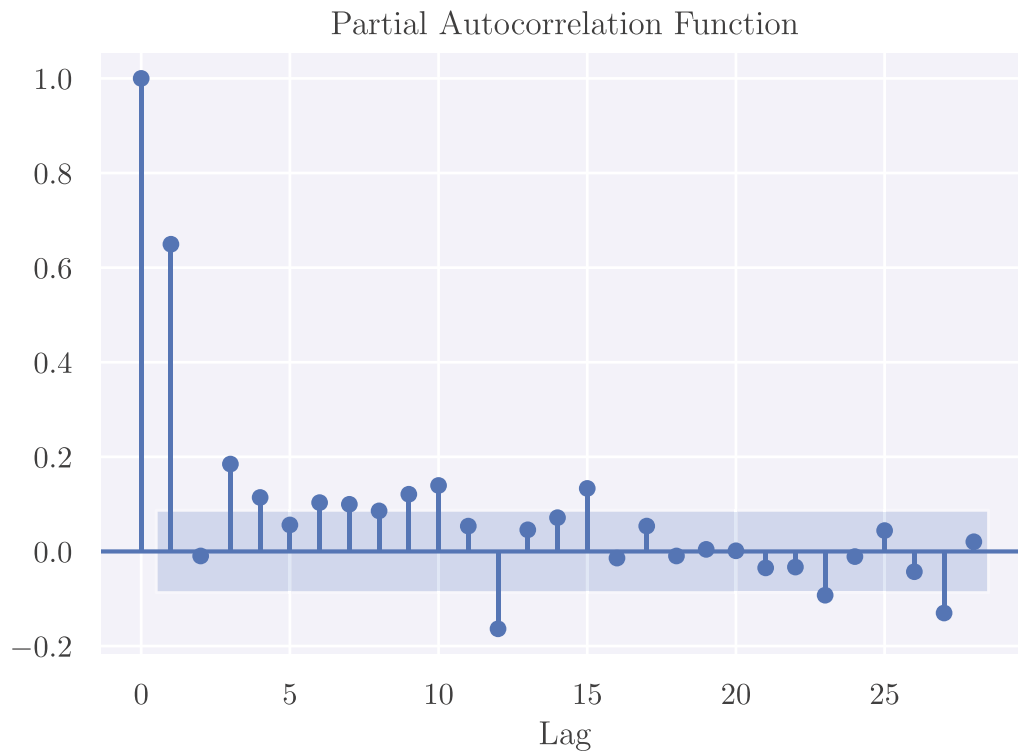


FIGURE 7 Partial autocorrelation function of the first difference of the log prices, as measured by the US CPI. Confidence interval (shaded area) is computed using Bartlett's formula with a significance level of 5%.

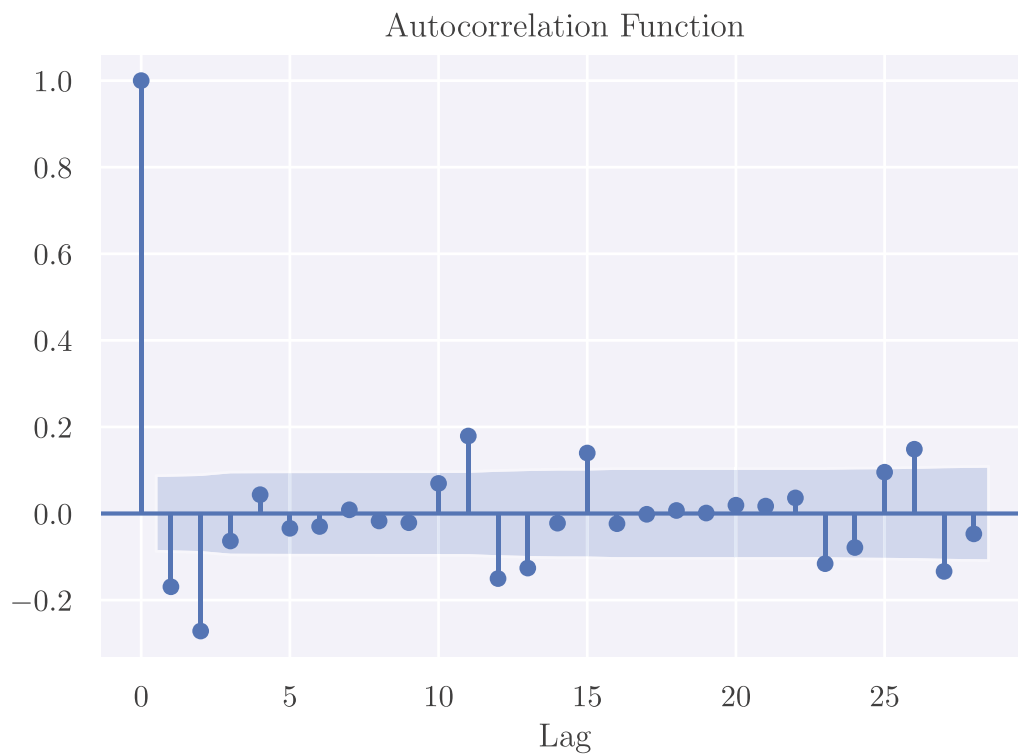


FIGURE 8 Autocorrelation function of the log inflation series after twice-differencing. Confidence interval (shaded area) is computed using Bartlett's formula with a significance level of 5%.

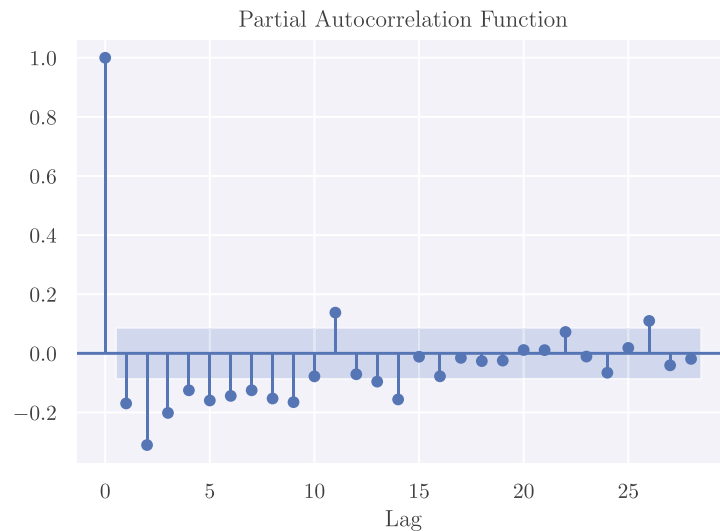


FIGURE 9 Partial autocorrelation function of the log inflation series after twice-differencing. Confidence interval (shaded area) is computed using Bartlett's formula with a significance level of 5%.

4 | RESULTS

Initially, Table 1 presents the ranks produced by the average of the out-of-sample MSE for all forecasting horizons (1, 2, 3, 6, and 12 months ahead) as well as for the cumulative forecasts over 3, 6, and 12 months. We leave the actual numbers and its correspondent reduction with respect to a random walk forecast to the appendix⁷. In addition, interested readers may find in the appendix analogous results for other performance metrics such as MAE, MAPE, and R^2 . Since the conclusions inferred from each of these metrics are virtually the same, and MSE is possibly the most common one for performance assessment in machine learning, we will concentrate the analysis only on the MSE.

Comparing the median MSE across the models, one infers that the winning model is the ConvLSTM coupled with a variational autoencoder for dimension reduction. Since this conclusion stems from a significant amount of simulations, with a rigorous division of the sample into training, validation, and test windows, the accomplishments of the ConvLSTM model are fairly sound and robust. In fact, the superiority is statistically significant at 5% in most horizons considered and in comparison with most benchmarks, as confirmed by the confidence intervals computed.⁸ In particular, the ConvLSTM model generate loss reductions as large as 91.5% for some horizons.

By contrast, the random walk is the worst performer in some forecasting windows. An immediate conclusion is that, given the weak accuracy exhibited by the naive random walk, even simple econometric models enhance out-of-sample predictions. Therefore, although inflation forecasting remains challenging, the results shown demonstrate that informative predictions can be generated, confronting other studies that claim the impossibility of beating the random walk. In addition, the simple moving average model of Reference 57 is also defeated by the ConvLSTM and other benchmarks, showing inflation can be forecast.

Analyzing the full picture, corroborating the findings of Reference 8, we find that machine learning methods and models that impose sparsity and/or regularization perform satisfactorily and provide substantial improvements with respect to the random walk. However, deep learning models do not perform equally. Indeed, the multilayer perceptron delivers lackluster results, consistently with the findings by Reference 8. In our view, such behavior is explained by the fact that, unlike LSTM and ConvLSTM, the MLP does not have an architecture designed to capture temporal dependencies in the input data. Therefore, it requires an excessive number of parameters to adjust to the inputs, leading to overfitting and, thus, poor out-of-sample performance.

⁷The empirical probability distributions of the MSE and MAE metrics reveal that, across simulations, the performance naturally oscillates, but within a tight range, and never reaching high loss levels. The additional data is available upon request.

⁸Usually, for more distant horizons, the intervals become wider due to the higher uncertainty and, thus, it is impossible to reject the null hypothesis that the models produce equivalent out-of-sample performance. Nevertheless, it must be mentioned though that, in most machine learning forecasting studies, the direct comparison of the median (or average) MSE is sufficient to claim the superiority of a model and, using this criterion, the ConvLSTM is a indisputable winner in every horizon, as one can see in the appendix.

TABLE 1 Ranks produced by comparing the models according to their average mean squared error (MSE) through the simulations.

Model	1	2	3	6	12	3M	6M	12M
LSTM	2	2	2	2	2	2	2	2
ConvLSTM	1	1	1	1	1	1	1	1
MLP	19	19	18	17	17	20	21	21
RW	26	24	24	24	24	26	24	24
Ridge CV	11	7	6	6	6	11	10	9
BRidge	10	14	12	13	12	13	14	14
LASSO CV	8	6	5	5	5	9	9	8
BLASSO	9	5	4	4	4	8	8	7
Enet CV	7	4	3	3	3	7	6	6
SVR	14	8	7	7	7	10	12	13
RF	18	16	13	14	14	16	17	15
BART	12	11	10	10	10	12	11	11
Bagging	20	20	16	16	16	19	20	20
kNN	21	9	8	8	8	14	13	12
Huber	23	25	25	25	25	24	25	25
Theil-Sen	24	26	26	26	26	25	26	26
Factors	22	12	9	11	11	17	18	16
GARCH	4	15	19	19	18	5	4	5
VECM	3	23	23	23	23	6	7	18
SETAR	17	21	21	20	19	23	23	23
MA	15	17	14	12	9	21	15	10
SARIMA	6	10	17	18	21	4	3	3
ARFIMA	5	3	20	21	20	3	5	4
GradBoost	13	18	15	15	15	18	19	19
AdaBoost	16	13	11	9	13	15	16	17
Bayes Reg.	25	22	22	22	22	22	22	22

Note: The two best models are highlight in bold font.

The performance of the factor model offers interesting insights. The mixed results are most likely explained by a missing structure of linear factors in the data, consistently with the findings reported by Reference 8. This conclusion is also supported by the fragile accuracy of other standard linear models added as benchmarks and reinforces the use of deep learning to identify these nonlinear interactions between factors⁹.

A question that immediately arises when adjusting models to time series is whether the out-performance of a certain model is verified exclusively in a particular time window, or whether it depends on business cycles or other exogenous variables. The approach established in this work, based on successive splits of the dataset, conceiving a diverse collection of training, validation, and test samples with which models are adjusted, addresses this question. Hence, each model has been trained on different samples, covering distinct periods of time, and tested on multiple settings as well.

With the purpose of inspecting how MSE behaves across time, we display the evolution of the MSE for the ConvLSTM (Figure 10). We find a strong, positive correlation between the MSE and the CPI volatility, meaning that, in periods of greater turbulence, performance seems to deteriorate, which is reasonable. Still, in the case of the model proposed, even during the 2008–2009 crisis, when volatility reached its peak, the MSE remained under control. We find that the out performance of ConvLSTM does not depend on the state of the economy, meaning that it outperforms both during expansions and recessions, or during periods with high and low uncertainty.

⁹Reference 34 also supports such conclusion using financial data. Using autoencoders, the authors demonstrate that a nonlinear factor structure is more suitable to explain the variability of the data in comparison with linear models.

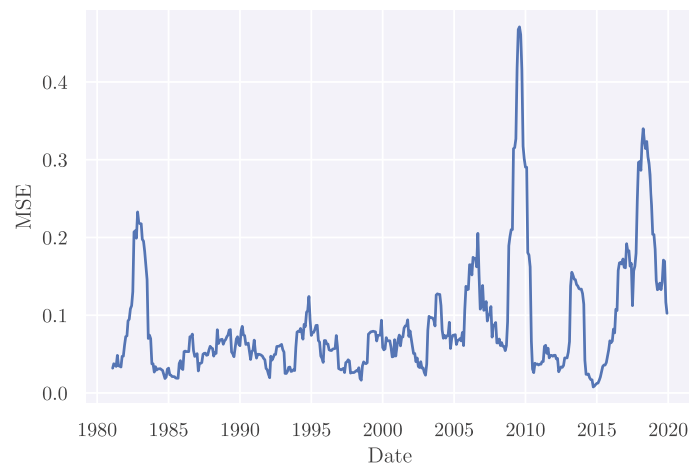


FIGURE 10 Mean squared error (MSE) of the convolutional LSTM (ConvLSTM) model computed using a 12-month rolling window.

In the framework of machine learning and big data, dimension reduction gains become explicit when assessing the performance offered by the ConvLSTM model with and without the variational autoencoder. As the outcomes show, reducing the dimension of the dataset improves the out-of-sample performance, meaning that redundant information exist in the time series. In order to confirm that using a nonlinear technique for dimension reduction provides greater gains than a linear one, the same exercise is conducted, but replacing the VAE by PCA. Although not fully reported here, but available at request, the results demonstrate that the combination of VAE and ConvLSTM is superior to the alternative specification proposed, which is another piece of evidence of the existence of a nonlinear factor structure in the input data.

The inferences obtained using the MSE criterion hold if one uses RMS, MAPE and R^2 , although the confidence intervals seem wider and losses are higher. To conclude, the results shown in this section confirm that the combination of ConvLSTM and variational autoencoders yields the best out-of-sample performance according to different metrics and in distinct windows of the time horizon considered in this study. Furthermore, the out performance of nonlinear models reveals that the linear Phillips curve, despite its theoretical appeal, fails to explain inflation due to the nonlinearities present in the latter.

5 | CONCLUSIONS

The present work is devoted to the evaluation of deep learning methods for inflation forecasting. In particular, a combination of ConvLSTM networks and variational autoencoders is compared against a wide selection of benchmarks comprised by popular econometric and machine learning models to forecast US inflation data. The decision to pursue this analysis stems from the promising findings reported in the literature, showing the robust prediction power deep learning has, especially in contexts where nonlinearities and nonstationarity are ubiquitous.

As expected, the experiments conducted demonstrate that coupling variational autoencoders to ConvLSTM networks yields compelling results. Merging these techniques significantly improves the out-of-sample accuracy in comparison with the benchmarks, generating the lowest median MSE across simulations. The same conclusion is obtained using distinct performance metrics, such as MAE, and is robust in multiple forecasting horizons. The simulations using different training and test samples confirm that, despite the variations observed across iterations, the proposed model delivers more accurate predictions in every scenario.

Since this is a initial study on the application of deep learning in macroeconomics, venues for improvement abound. On the modelling side, multiple extensions of the present work can be envisioned, such as applying wavelets for denoising and decomposition of the input data.⁶¹ By working in multiple scales and high resolution, wavelets determine not only which frequencies are existent in a signal, but also at which time they have occurred, which is helpful to detect noise and local features. An instructive example is provided by Reference 62. Furthermore, adding a Bayesian refinement through the incorporation of priors might be worth exploring since inflation is usually positive due to the low frequency of deflation periods in most economies.

Apart from that, variable selection was outside the scope of this work, but may also improve out-of-sample performance.⁶³ appraise inflation forecast accuracy over short-term horizon via CPI disaggregated data. The authors adopt

a bottom-up approach, aggregating forecasts and later comparing against predictions obtained using the aggregated CPI. In this sense, the combination of forecasts for disaggregated prices also shows potential; see References 64 and 65.

Another area of improvement comprehends prevention of overfitting. In this study, techniques such as dropout layers and batch normalization have been extensively adopted with this purpose, but other approaches exist. For instance, Reference 66 introduce the BinaryConnect, a method which consists in training a deep neural network with binary weights. The authors conclude that, analogously to other dropout schemes, BinaryConnect acts as a regularizes and yields near state-of-the-art results in many applications.

Taking together the refinements envisioned, the performance of the VAE-ConvLSTM model could be improved, but this does not invalidate the study, since the current version is successful in tackling the main issues laid down in the introduction: (1) demonstrating the existence of nonlinearities in inflation, which can be seen through the lackluster performance of the linear benchmarks and the statistical analysis in section 4; (2) showing the importance of preparation and dimension reduction of input data; and (3) presenting a promising modeling alternative based on deep learning to deal with nonlinearities and redundant input data.

About this last point, we conjecture that our model could also be successful in forecasting other macroeconomic variables, since most of them suffer from the same complexities identified in inflation time series. Based on our findings and the accomplishments of the VAE-ConvLSTM model, this exploration is definitely worthy for future studies.

ACKNOWLEDGMENTS

We would like to thank Miguel Bandeira for his invaluable comments. Of course, all remaining errors are our own. The views expressed are those of the authors and do not necessarily reflect those of the Central Bank of Brazil.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in FRED MD and FRED QD: Monthly and Quarterly Databases at <https://research.stlouisfed.org/econ/mccracken/freddatabases/>.

ORCID

Alexandre Fernandes Theoharidis  <https://orcid.org/0000-0003-4736-2327>

REFERENCES

1. Faust J, Wright JH. Forecasting inflation. *Handbook of Economic Forecasting*. 1st ed. Elsevier; 2013:2-56.
2. Rudd J, Whelan K. Modeling inflation dynamics. A critical review of recent research. *J Money Credit Bank*. 2007;39(1):155-170.
3. Kumar A, Orrenius PM. A closer look at the Phillips curve using state-level data. *J Macroecon*. 2016;47(3):84-102.
4. Zhang L. Modeling the phillips curve in China. A nonlinear perspective. *Macroecon Dyn*. 2017;21(2):439-461.
5. Daly MC, Hobijn B. Downward nominal wage rigidities bend the phillips curve. *J Money Credit Bank*. 2014;46(2):51-93.
6. Grauwe PD, Ji Y. Inflation targets and the zero lower bound in a behavioral macroeconomic model. *Econom*. 2019;86(342):262-299.
7. Bloom N. The impact of uncertainty shocks. *Econom*. 2009;77(3):623-685.
8. Medeiros MC, Vasconcelos GFR, Veiga A, Zilberman E. Forecasting inflation in a data-rich environment. The benefits of machine learning methods. *J Bus Econ Stat*. 2021;39(1):98-119. doi:10.1080/07350015.2019.1637745
9. McCracken M, Ng S. FRED-MD. A monthly database for macroeconomic research. *J Bus Econ Stat*. 2016;34(4):574-589.
10. Lerouge J, Herault R, Chatelain C, Jardin F, Modzelewski R. IODA. An input/output deep architecture for image labeling. *Pattern Recognit*. 2015;48(9):2847-2858.
11. Li X, Wu X. Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing. Brisbane: IEEE. 2015 4520-4524.
12. Kim J, Calhoun VD, Shim E, Lee J-H. Deep neural network with weight sparsity control and pre-training extracts hierarchical features and enhances classification performance. Evidence from whole-brain resting-state functional connectivity patters of schizophrenia. *NeuroImage*. 2016;124(1):127-146.
13. Goodfellow I, Bengio Y, Courville A. *Deep Learning*. 1st ed. The MIT Press; 2016.
14. Liu W, Wang Z, Liu X, Zeng N, Liu Y, Alsaadi FE. A survey of deep neural network architectures and their applications. *Neurocomput*. 2017;234(1):11-26.
15. Atsalakis GS, Valavanis KP. Surveying stock market forecasting techniques – part II. Soft computing methods. *Expert Syst Appl*. 2009;36(3):5932-5941.
16. Ahmed NK, Atiya AF, El Gayar N, El-Shishiny H. An empirical comparison of machine learning models for time series forecasting. *Econom Rev*. 2010;29(5):594-621.
17. Långkvist M, Karlsson L, Loutfi A. A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognit Lett*. 2017;42(2):11-24.
18. Tkáč M, Verner R. Artificial neural networks in business. Two decades of research. *Appl Soft Comput*. 2016;38(1):788-804.
19. McAdam P, McNelis P. Forecasting inflation with thick models and neural networks. *Econ Model*. 2005;22(5):848-867.

20. Choudhary MA, Haider A. Neural network models for inflation forecasting. An appraisal. *Appl Econ*. 2012;44(20):2631-2635.
21. Garcia MGP, Medeiros MC, Vasconcelos GFR. Real-time inflation forecasting with high-dimensional models. The case of Brazil. *Int J Forecast*. 2017;33(3):679-693.
22. Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Comput*. 1997;9(1):1735-1780.
23. Gers FA, Schmidhuber J, Cummins F. Learning to forget. Continual prediction with LSTM. *Neural Comput*. 2000;12(10):2451-2471.
24. Graves A, Schmidhuber J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw*. 2005;18(5):602-610.
25. Cho K, van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation. Proceedings Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics. 2014 1724-1734.
26. Shi X, Chen Z, Wang H, Yeung D-Y, Wong W-K, Woo W-C. Convolutional LSTM network. A machine learning approach for precipitation nowcasting. Proceedings International Conference on Neural Information Processing Systems. Cambridge: NIPS. 2015 802-810.
27. Bao W, Yue J, Rao Y. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLoS ONE*. 2017;12(7):203-228.
28. Essien A, Giannetti C. A deep learning framework for univariate time series prediction using convolutional LSTM stacked autoencoders. Proceedings IEEE International Symposium on Innovations in Intelligent Systems and Applications. Sofia: IEEE. 2019 1-6.
29. Fischer T, Krauss C. Deep learning with long short-term memory networks for financial market predictions. *Eur J Oper Res*. 2018;270(2):654-669.
30. Wang K, Qi X, Liu H. Photovoltaic power forecasting based LSTM-Convolutional network. *Energy*. 2019;189(1):1-11.
31. Álvarez-Díaz M, Gupta R. Forecasting US consumer price index. Does nonlinearity matter? *Appl Econ*. 2016;48(46):4462-4475.
32. Ülke V, Sahin A, Subasi A. A comparison of time series and machine learning models for inflation forecasting. Empirical evidence from the USA. *Neural Comput Appl*. 2018;30(5):1519-1527.
33. Wang Y, Yao H, Zhao S. Auto-encoder based dimensionality reduction. *Neurocomput*. 2016;184(1):232-242.
34. Gu S, Kelly B, Xiu D. Autoencoder asset pricing models. *J Econom*. 2021;222(1):429-450. doi:10.1016/j.jeconom.2020.07.009
35. Doersch C. Tutorial on Variational Autoencoders. 2016 arXiv: 1606.05908 [stat.ML].
36. Bernanke BS, Boivin J. Monetary policy in a data-rich environment. *J Financ Econ*. 2003;50(3):525-546.
37. Adamowski JF. Peak daily water demand forecast modeling using artificial neural networks. *J Water Resour Plan Manag*. 2008;134(2):119-128.
38. Galeshchuk S. Neural networks performance in exchange rate prediction. *Neurocomput*. 2016;172(1):446-452.
39. Kim HY, Won CH. Forecasting the volatility of stock price index. A hybrid model integrating LSTM with multiple GARCH-type models. *Expert Syst Appl*. 2018;103(1):25-37.
40. Kim C, Kim TY, Oh KJ, Do JD. Artificial neural networks for non-stationary time series. *Neurocomput*. 2004;61(8):439-447.
41. Correa AS, Minella A. Nonlinear mechanisms of the exchange rate pass-through. A Phillips curve model with threshold for Brazil. *Rev Brasileira de Econ*. 2010;64(3):231-243.
42. Ball L, Mazumder S. Inflation dynamics and the great recession. *Brook Papers Econ Activ*. 2011;42(1):337-405.
43. Messina R, Louradour J. Segmentation-free handwritten Chinese text recognition with LSTM-RNN. Proceedings International Conference on Document Analysis and Recognition. Tunis: IEEE. 2015 171-175.
44. Liu G, Guo J. Bidirectional LSTM with attention mechanism and convolutional layer for text classification. *Neurocomput*. 2019;337(1):325-338.
45. Sezer OB, Ozbayoglu AM. Algorithmic financial trading with deep convolutional networks. Time series to image conversion approach. *Appl Soft Comput*. 2018;70(1):525-538.
46. Bai S, Tang H, An S. Coordinate CNNs and LSTMs to categorize scene images with multi-views and multi-levels of abstraction. *Expert Syst Appl*. 2019;120(3):298-309.
47. Kristjanpoller W, Minutolo MC. Gold price volatility. A forecasting approach using the artificial neural network-GARCH model. *Expert Syst Appl*. 2015;42(20):7245-7251.
48. Dash NB, Panda SN, Remesan R, Sahoo N. Hybrid neural modeling for groundwater level prediction. *Neural Comput Appl*. 2010;19(8):1251-1263.
49. Khashei M, Bijari M. A novel hybridization of artificial neural networks and ARIMA models for time series forecasting. *Appl Soft Comput*. 2011;11(2):2664-2675.
50. Boivin J, Ng S. Are more data always better for factor analysis? *J Econom*. 2006;132(1):169-194.
51. Pereira J, Silveira M. Unsupervised anomaly detection in energy time series data using variational recurrent autoencoders with attention. Proceedings International Conference on Machine Learning and Applications. Orlando, FL: IEEE. 2018 1-13.
52. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout. A simple way to prevent neural networks from overfitting. *J Mach Learn Res*. 2014;15(6):1929-1958.
53. Dozat T. Incorporating Nesterov momentum into Adam. Proceedings International Conference on Learning Representations. San Juan, Puerto Rico: ICLR. 2016 1-4.
54. James G, Witten D, Hastie T, Tibshirani R. *An Introduction to Statistical Learning with Application in R*. 1st ed. Springer New York; 2013.
55. Kuhn M, Johnson K. *Applied Predictive Modeling*. 1st ed. Springer; 2013.
56. Keskar NS, Mudigere D, Nocedal J, Smelyanskiy M, Tang PTP. On large-batch training for deep learning. Generalization gap and sharp minima. Proceedings International Conference on Learning Representations. Toulon, France: ICLR. 2017 1-13.

57. Atkeson A, Ohanian LE. Are phillips curves useful for forecasting inflation? *Federal Reserve Bank of Minneapolis Quart Rev.* 2001;25(1):2-11.
58. Su Y, Kay J. On extended long short-term memory and dependent bidirectional recurrent neural network. *Neurocomput.* 2019;356(1):151-161.
59. Silverman B. *Density Estimation for Statistics and Data Analysis.* 1st ed. Chapman and Hall; 1986.
60. Monache DD, Petrella I. Adaptive models and heavy tails with an application to inflation forecasting. *Int J Forecast.* 2017;33(2):482-501.
61. Percival DB, Walden AT. *Wavelet Methods for Time Series Analysis.* 2nd ed. Cambridge University Press; 2006.
62. Hsieh T-J, Hsiao H-F, Yeh W-C. Forecasting stock markets using wavelet transforms and recurrent neural networks. An integrated system based on artificial bee colony algorithm. *Appl Soft Comput.* 2011;11(2):2510-2525.
63. Duarte C, Rua A. Forecasting inflation through a bottom-up approach. How bottom is bottom? *Econ Model.* 2007;24(6):941-953.
64. Monacelli T, Sala L. The international dimension of inflation. Evidence from disaggregated consumer price data. *J Money Credit Bank.* 2009;41(1):101-120.
65. Coleman S. Inflation persistence in the Franc zone. Evidence from disaggregated prices. *J Macroecon.* 2010;32(1):426-442.
66. Courbariaux M, Bengio Y, David J-P. BinaryConnect. Training deep neural networks with binary weights during propagations. Proceedings International Conference on Neural Information Processing Systems. MIT Press. 2015 3123-3131.

How to cite this article: Theoharidis AF, Guillén DA, Lopes H. Deep learning models for inflation forecasting. *Appl Stochastic Models Bus Ind.* 2023;39(3):447-470. doi: 10.1002/asmb.2757

APPENDIX A. OUT-OF-SAMPLE MSE

Tables A1 and A2 show the out-of-sample MSE produced by each model, together with the average reduction in MSE with respect to the random walk.

TABLE A1 Descriptive statistics of the out-of-sample mean squared error (MSE) of each model.

Model	Statistic	1	2	3	6	12	3M	6M	12M
LSTM	Q1	0.24	0.46	0.50	0.50	0.51	0.37	0.47	0.41
	Median	0.32	0.51	0.55	0.55	0.57	0.42	0.55	0.49
	Q3	0.40	0.60	0.60	0.64	0.62	0.49	0.68	0.57
ConvLSTM	Q1	0.15	0.40	0.44	0.45	0.46	0.30	0.39	0.29
	Median	0.17	0.42	0.45	0.46	0.46	0.33	0.43	0.34
	Q3	0.19	0.44	0.46	0.46	0.47	0.37	0.50	0.40
MLP	Q1	0.92	1.10	1.10	1.10	1.09	1.05	1.09	1.11
	Median	0.99	1.16	1.16	1.16	1.18	1.11	1.15	1.17
	Q3	1.07	1.22	1.21	1.25	1.25	1.17	1.24	1.24
RW	Q1	1.89	1.84	1.87	1.88	1.81	1.90	1.94	1.91
	Median	2.04	1.99	1.99	2.05	1.97	2.01	2.06	2.07
	Q3	2.15	2.11	2.16	2.19	2.18	2.14	2.19	2.24
Ridge CV	Q1	0.80	0.99	0.99	0.99	0.97	0.97	0.99	0.99
	Median	0.85	1.01	1.02	1.03	1.03	0.99	1.00	1.00
	Q3	0.89	1.02	1.03	1.04	1.07	1.01	1.02	1.01
BRidge	Q1	0.78	1.03	1.03	1.03	1.02	0.97	1.00	1.02
	Median	0.85	1.07	1.08	1.08	1.10	1.01	1.04	1.05
	Q3	0.91	1.11	1.11	1.14	1.15	1.07	1.10	1.11
LASSO CV	Q1	0.78	1.00	1.00	0.99	0.98	0.96	1.00	1.00
	Median	0.81	1.01	1.01	1.02	1.03	1.00	1.00	1.00
	Q3	0.87	1.01	1.02	1.04	1.07	1.00	1.00	1.00

(Continues)

TABLE A1 (Continued)

Model	MSE	1	2	3	6	12	3M	6M	12M
LASSO	Q1	0.78	1.00	0.99	0.99	0.97	0.96	0.99	0.99
	Median	0.83	1.01	1.01	1.02	1.02	0.98	1.00	1.00
	Q3	0.90	1.01	1.02	1.04	1.07	1.00	1.00	1.00
Enet CV	Q1	0.71	0.98	0.98	0.98	0.95	0.93	0.96	0.97
	Median	0.76	1.00	1.00	1.01	1.01	0.96	0.98	0.99
	Q3	0.83	1.01	1.01	1.03	1.06	0.98	1.00	1.00
SVR	Q1	0.90	1.01	1.01	1.00	1.00	0.97	0.99	1.00
	Median	0.92	1.03	1.03	1.05	1.06	0.99	1.02	1.03
	Q3	0.94	1.06	1.05	1.08	1.10	1.02	1.04	1.06
RF	Q1	0.87	1.05	1.04	1.05	1.03	1.01	1.02	1.02
	Median	0.93	1.09	1.09	1.10	1.10	1.05	1.05	1.07
	Q3	1.00	1.13	1.13	1.15	1.17	1.08	1.10	1.10
BART	Q1	0.86	1.01	1.02	1.02	1.02	0.96	0.98	1.00
	Median	0.91	1.05	1.06	1.07	1.08	1.00	1.01	1.02
	Q3	0.96	1.08	1.10	1.11	1.13	1.04	1.04	1.04
Bagging	Q1	0.94	1.12	1.08	1.08	1.10	1.05	1.05	1.05
	Median	0.99	1.16	1.14	1.15	1.17	1.10	1.09	1.12
	Q3	1.06	1.21	1.21	1.21	1.23	1.15	1.14	1.17
kNN	Q1	0.98	1.00	1.00	1.01	1.00	1.00	1.01	1.00
	Median	1.00	1.03	1.04	1.05	1.05	1.02	1.03	1.03
	Q3	1.02	1.07	1.07	1.07	1.11	1.05	1.05	1.06
Huber	Q1	0.93	1.99	2.03	1.98	2.01	1.67	1.88	1.92
	Median	1.04	2.18	2.25	2.32	2.20	1.85	2.04	2.14
	Q3	1.19	2.42	2.43	2.55	2.52	2.04	2.25	2.44
Theil-Sen	Q1	0.94	2.08	2.10	2.14	2.11	1.71	1.93	1.97
	Median	1.04	2.31	2.33	2.44	2.33	1.90	2.09	2.19
	Q3	1.20	2.55	2.58	2.68	2.60	2.11	2.30	2.52
Factors	Q1	0.93	1.01	1.01	1.02	1.02	1.01	1.01	1.01
	Median	1.01	1.04	1.05	1.07	1.08	1.06	1.05	1.03
	Q3	1.08	1.08	1.08	1.11	1.13	1.10	1.11	1.08
GARCH	Q1	0.49	0.72	0.83	0.79	0.82	0.67	0.60	0.64
	Median	0.64	0.93	1.05	1.11	1.09	0.72	0.70	0.70
	Q3	0.89	1.34	1.52	1.52	1.46	0.82	0.80	0.83
VECM	Q1	0.63	1.28	1.45	1.27	1.42	0.78	0.79	0.77
	Median	0.70	1.44	1.63	1.48	1.57	0.91	0.96	1.13
	Q3	0.77	1.65	1.94	1.70	1.82	1.05	1.15	1.38
SETAR	Q1	0.80	0.99	1.11	1.04	1.04	1.27	1.33	1.53
	Median	0.93	1.16	1.27	1.20	1.17	1.42	1.55	1.81
	Q3	1.06	1.33	1.48	1.36	1.41	1.59	1.74	2.01
MA	Q1	0.84	1.01	0.99	0.92	0.92	1.01	0.93	0.88
	Median	0.93	1.13	1.12	1.06	1.07	1.12	1.06	1.02
	Q3	1.03	1.27	1.27	1.22	1.24	1.24	1.20	1.19

(Continues)

TABLE A1 (Continued)

Model	MSE	1	2	3	6	12	3M	6M	12M
SARIMA	Q1	0.70	0.93	1.02	1.09	1.10	0.67	0.60	0.61
	Median	0.77	1.02	1.13	1.24	1.22	0.75	0.69	0.68
	Q3	0.82	1.14	1.28	1.35	1.34	0.81	0.78	0.77
ARFIMA	Q1	0.69	0.83	1.02	1.08	1.08	0.62	0.63	0.62
	Median	0.75	0.95	1.14	1.19	1.21	0.70	0.71	0.71
	Q3	0.82	1.11	1.38	1.34	1.35	0.79	0.81	0.81
GradBoost	Q1	0.84	1.07	1.09	1.09	1.07	1.03	1.03	1.05
	Median	0.92	1.13	1.16	1.14	1.15	1.07	1.08	1.10
	Q3	0.98	1.19	1.20	1.20	1.23	1.12	1.13	1.15
AdaBoost	Q1	0.88	1.02	1.03	1.03	1.02	0.99	1.02	1.02
	Median	0.93	1.06	1.06	1.07	1.10	1.04	1.06	1.06
	Q3	0.97	1.09	1.10	1.10	1.15	1.08	1.09	1.11
Bayes Reg.	Q1	0.99	1.22	1.22	1.23	1.24	1.19	1.20	1.23
	Median	1.07	1.33	1.32	1.33	1.34	1.30	1.31	1.33
	Q3	1.20	1.43	1.42	1.42	1.50	1.40	1.42	1.46

TABLE A2 Average mean squared error (MSE) reduction delivered by each model with respect to the random walk.

Model	1	2	3	6	12	3M	6M	12M
LSTM	83.6%	72.8%	72.4%	71.6%	70.9%	78.4%	71.4%	75.2%
ConvLSTM	91.5%	79.0%	77.9%	77.9%	77.0%	83.4%	77.8%	82.8%
MLP	50.5%	41.5%	42.5%	42.3%	40.8%	44.8%	43.7%	43.4%
RW	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Ridge CV	57.5%	49.3%	49.9%	49.9%	49.0%	50.6%	51.3%	51.7%
BRidge	57.7%	46.0%	46.9%	46.6%	45.7%	49.8%	49.1%	48.9%
LASSO CV	58.8%	49.3%	50.3%	50.4%	49.1%	51.2%	51.4%	51.8%
BLASSO	58.5%	49.5%	50.3%	50.5%	49.3%	51.6%	51.7%	51.9%
Enet CV	62.0%	50.1%	50.8%	50.9%	50.0%	53.0%	52.7%	52.7%
SVR	54.5%	48.1%	48.9%	49.1%	47.5%	50.7%	50.8%	50.4%
RF	53.3%	44.8%	46.1%	45.9%	45.0%	48.4%	48.6%	48.8%
BART	55.0%	47.1%	47.4%	47.2%	46.2%	50.4%	50.9%	51.1%
Bagging	50.4%	41.4%	43.2%	42.5%	41.5%	45.6%	46.9%	46.5%
kNN	50.3%	47.9%	48.6%	48.4%	47.4%	49.3%	50.2%	50.4%
Huber	47.4%	-12.5%	-11.7%	-11.9%	-13.4%	7.5%	-0.3%	-4.9%
Theil-Sen	47.0%	-18.4%	-17.5%	-17.7%	-19.0%	4.8%	-3.0%	-7.6%
Factors	49.6%	46.9%	48.2%	47.0%	46.2%	47.3%	48.5%	48.7%
GARCH	65.0%	46.0%	41.0%	40.1%	39.3%	63.0%	65.5%	64.8%
VECM	65.0%	25.7%	15.1%	17.2%	17.4%	54.4%	52.7%	47.3%
SETAR	53.6%	41.1%	35.3%	40.0%	38.6%	28.6%	24.2%	15.1%
MA	54.0%	43.1%	44.2%	46.8%	46.4%	44.4%	48.8%	51.2%
SARIMA	62.0%	47.8%	43.0%	40.7%	38.2%	63.0%	66.0%	66.6%
ARFIMA	62.7%	51.1%	40.5%	39.3%	38.6%	64.7%	65.3%	65.2%
GradBoost	54.6%	42.8%	43.3%	42.9%	42.2%	46.8%	47.5%	47.0%
AdaBoost	54.0%	46.6%	47.2%	47.2%	45.6%	48.4%	48.7%	48.6%
Bayes Reg.	45.5%	32.9%	34.3%	33.9%	31.4%	35.2%	35.9%	35.2%

Note: Negative values mean that the model increased the MSE. The winning model is highlighted in bold font.

APPENDIX B. OUT-OF-SAMPLE MAE

Tables B1, B2, and B3 show the out-of-sample MAE produced by each model, together with the average reduction in MAE with respect to the random walk and the corresponding performance ranks.

TABLE B1 Descriptive statistics of the out-of-sample mean absolute error (MAE) of each model.

Model	Statistic	1	2	3	6	12	3M	6M	12M
LSTM	Q1	0.40	0.55	0.57	0.57	0.56	0.49	0.55	0.51
	Median	0.45	0.58	0.60	0.59	0.61	0.53	0.59	0.57
	Q3	0.51	0.63	0.63	0.64	0.64	0.58	0.66	0.63
ConvLSTM	Q1	0.30	0.52	0.53	0.54	0.54	0.44	0.50	0.43
	Median	0.32	0.53	0.54	0.54	0.54	0.46	0.53	0.46
	Q3	0.35	0.54	0.55	0.55	0.55	0.49	0.58	0.50
MLP	Q1	0.68	0.76	0.76	0.76	0.77	0.79	0.82	0.86
	Median	0.71	0.79	0.79	0.80	0.80	0.82	0.85	0.88
	Q3	0.74	0.82	0.82	0.84	0.83	0.85	0.88	0.91
RW	Q1	1.02	1.01	1.01	1.02	1.00	1.06	1.08	1.09
	Median	1.07	1.05	1.06	1.07	1.05	1.10	1.11	1.13
	Q3	1.10	1.09	1.11	1.10	1.11	1.13	1.15	1.16
Ridge CV	Q1	0.62	0.69	0.69	0.68	0.69	0.75	0.77	0.80
	Median	0.64	0.72	0.71	0.71	0.71	0.77	0.79	0.82
	Q3	0.67	0.73	0.73	0.74	0.74	0.79	0.82	0.84
BRidge	Q1	0.63	0.72	0.72	0.72	0.72	0.75	0.79	0.81
	Median	0.65	0.75	0.74	0.76	0.75	0.78	0.82	0.84
	Q3	0.68	0.78	0.77	0.78	0.78	0.80	0.84	0.86
LASSO CV	Q1	0.62	0.69	0.69	0.69	0.70	0.74	0.77	0.80
	Median	0.64	0.71	0.71	0.71	0.71	0.76	0.79	0.82
	Q3	0.66	0.73	0.73	0.74	0.74	0.78	0.82	0.84
BLASSO	Q1	0.63	0.69	0.69	0.68	0.69	0.74	0.77	0.80
	Median	0.66	0.71	0.71	0.71	0.71	0.76	0.79	0.82
	Q3	0.69	0.73	0.73	0.73	0.74	0.78	0.81	0.84
Enet CV	Q1	0.60	0.68	0.68	0.68	0.68	0.73	0.76	0.79
	Median	0.63	0.71	0.71	0.71	0.71	0.75	0.78	0.81
	Q3	0.65	0.73	0.72	0.73	0.73	0.77	0.81	0.83
SVR	Q1	0.65	0.70	0.70	0.70	0.70	0.74	0.78	0.80
	Median	0.67	0.73	0.73	0.73	0.73	0.77	0.80	0.83
	Q3	0.69	0.75	0.74	0.75	0.75	0.79	0.82	0.85
RF	Q1	0.67	0.74	0.72	0.73	0.73	0.76	0.79	0.81
	Median	0.69	0.76	0.75	0.76	0.76	0.79	0.82	0.84
	Q3	0.71	0.78	0.77	0.78	0.79	0.81	0.84	0.87
BART	Q1	0.66	0.71	0.71	0.71	0.72	0.75	0.78	0.80
	Median	0.68	0.74	0.74	0.74	0.75	0.77	0.80	0.83
	Q3	0.71	0.76	0.76	0.77	0.78	0.79	0.82	0.85

(Continues)

TABLE B1 (Continued)

Model	Statistic	1	2	3	6	12	3M	6M	12M
Bagging	Q1	0.70	0.77	0.74	0.75	0.76	0.79	0.81	0.83
	Median	0.72	0.79	0.77	0.78	0.79	0.82	0.83	0.86
	Q3	0.74	0.82	0.80	0.81	0.82	0.83	0.86	0.89
kNN	Q1	0.69	0.71	0.70	0.71	0.71	0.76	0.78	0.81
	Median	0.71	0.73	0.72	0.73	0.73	0.78	0.81	0.83
	Q3	0.73	0.75	0.75	0.76	0.77	0.80	0.83	0.85
Huber	Q1	0.72	1.10	1.09	1.11	1.10	1.01	1.09	1.11
	Median	0.76	1.15	1.15	1.17	1.16	1.07	1.14	1.16
	Q3	0.79	1.22	1.22	1.23	1.25	1.10	1.20	1.24
Theil-Sen	Q1	0.72	1.11	1.12	1.13	1.13	1.02	1.09	1.12
	Median	0.76	1.19	1.18	1.20	1.20	1.08	1.16	1.18
	Q3	0.79	1.25	1.26	1.27	1.28	1.12	1.22	1.26
Factors	Q1	0.69	0.71	0.71	0.71	0.71	0.76	0.79	0.81
	Median	0.72	0.74	0.74	0.74	0.74	0.79	0.82	0.83
	Q3	0.76	0.76	0.76	0.77	0.77	0.82	0.85	0.86
GARCH	Q1	0.56	0.67	0.68	0.70	0.71	0.62	0.61	0.62
	Median	0.63	0.74	0.79	0.81	0.81	0.67	0.65	0.67
	Q3	0.72	0.89	0.93	0.93	0.95	0.71	0.70	0.73
VECM	Q1	0.61	0.89	0.95	0.90	0.94	0.70	0.69	0.72
	Median	0.64	0.93	1.02	0.96	0.99	0.75	0.78	0.85
	Q3	0.68	1.02	1.12	1.03	1.06	0.81	0.86	0.94
SETAR	Q1	0.64	0.73	0.78	0.76	0.75	0.87	0.92	1.00
	Median	0.68	0.78	0.85	0.81	0.81	0.93	0.98	1.07
	Q3	0.72	0.84	0.91	0.87	0.89	0.99	1.06	1.14
MA	Q1	0.61	0.68	0.66	0.64	0.64	0.68	0.64	0.62
	Median	0.68	0.76	0.75	0.73	0.74	0.75	0.73	0.72
	Q3	0.76	0.85	0.85	0.84	0.86	0.83	0.83	0.84
SARIMA	Q1	0.67	0.77	0.81	0.83	0.84	0.65	0.62	0.62
	Median	0.71	0.81	0.85	0.88	0.88	0.69	0.67	0.67
	Q3	0.74	0.85	0.91	0.93	0.95	0.73	0.71	0.71
ARFIMA	Q1	0.65	0.73	0.80	0.82	0.83	0.63	0.64	0.63
	Median	0.69	0.77	0.85	0.87	0.87	0.66	0.68	0.68
	Q3	0.72	0.84	0.94	0.92	0.93	0.70	0.71	0.72
GradBoost	Q1	0.66	0.75	0.76	0.75	0.75	0.78	0.80	0.83
	Median	0.68	0.77	0.78	0.78	0.78	0.80	0.83	0.86
	Q3	0.71	0.80	0.80	0.80	0.81	0.82	0.85	0.88
AdaBoost	Q1	0.66	0.72	0.72	0.72	0.72	0.76	0.79	0.82
	Median	0.69	0.74	0.74	0.74	0.75	0.78	0.82	0.84
	Q3	0.71	0.76	0.77	0.77	0.78	0.81	0.84	0.86
Bayes Reg.	Q1	0.73	0.82	0.82	0.82	0.83	0.85	0.86	0.90
	Median	0.77	0.86	0.86	0.87	0.87	0.88	0.91	0.93
	Q3	0.81	0.90	0.90	0.90	0.92	0.92	0.95	0.98

TABLE B2 Average mean absolute error (MAE) reduction delivered by each model with respect to random walk.

Model	1	2	3	6	12	3M	6M	12M
LSTM	57.9%	44.5%	43.5%	43.5%	42.1%	51.6%	46.9%	49.5%
ConvLSTM	69.7%	49.7%	48.9%	49.2%	48.0%	58.2%	52.1%	58.8%
MLP	33.1%	24.5%	25.7%	25.6%	23.8%	25.7%	23.3%	22.0%
RW	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Ridge CV	39.8%	31.6%	32.9%	33.1%	31.8%	30.2%	28.8%	27.1%
BRidge	39.1%	28.7%	30.1%	29.9%	28.4%	29.2%	26.5%	25.3%
LASSO CV	39.7%	32.6%	33.7%	33.4%	32.3%	30.4%	28.9%	27.3%
BLASSO	38.3%	32.4%	33.7%	33.8%	32.4%	30.6%	28.9%	27.3%
Enet CV	41.2%	32.6%	33.7%	33.7%	32.3%	31.4%	29.6%	27.7%
SVR	36.9%	30.4%	31.6%	32.5%	30.4%	29.7%	28.1%	26.1%
RF	35.7%	27.4%	29.3%	29.7%	27.1%	28.5%	26.4%	25.6%
BART	36.3%	29.2%	30.5%	30.5%	28.8%	30.0%	28.5%	26.6%
Bagging	32.5%	24.6%	27.4%	26.8%	24.7%	25.6%	25.7%	23.6%
kNN	33.2%	30.3%	32.0%	31.6%	30.4%	29.4%	27.6%	26.0%
Huber	29.2%	-10.0%	-7.8%	-9.0%	-11.2%	2.5%	-2.0%	-2.9%
Theil-Sen	28.6%	-14.0%	-10.6%	-11.9%	-14.5%	1.4%	-4.2%	-4.6%
Factors	32.2%	29.8%	30.9%	30.5%	29.2%	27.6%	26.7%	26.0%
GARCH	40.7%	29.1%	25.4%	23.8%	23.0%	39.3%	41.4%	40.6%
VECM	39.9%	11.2%	3.8%	5.9%	5.4%	31.3%	29.8%	24.6%
SETAR	36.3%	25.5%	20.4%	24.6%	23.0%	15.7%	12.3%	5.3%
MA	35.9%	27.9%	29.9%	31.9%	29.6%	32.0%	34.3%	36.5%
SARIMA	33.7%	22.4%	19.9%	18.1%	15.9%	37.2%	40.3%	40.9%
ARFIMA	35.4%	27.0%	20.1%	18.6%	17.1%	39.5%	38.8%	39.8%
GradBoost	36.3%	26.1%	26.7%	27.2%	26.0%	27.4%	25.6%	24.0%
AdaBoost	35.3%	29.4%	30.1%	30.0%	28.8%	28.7%	26.6%	25.6%
Bayes Reg.	27.7%	18.0%	18.9%	19.6%	17.3%	19.7%	18.4%	17.3%

Note: Negative values mean that the model increased the MAE. The winning model is highlighted in bold font.

TABLE B3 Ranks produced by comparing the models according to their average mean absolute error (MAE) through the simulations.

Model	1	2	3	6	12	3M	6M	12M
LSTM	2	2	2	2	2	2	2	2
ConvLSTM	1	1	1	1	1	1	1	1
MLP	20	20	17	17	19	20	21	21
RW	26	24	24	24	24	26	24	24
Ridge CV	6	6	6	6	6	11	11	10
BRidge	8	13	11	13	13	15	17	17
LASSO CV	7	3	5	5	5	10	9	9
BLASSO	9	5	3	3	4	9	10	8
Enet CV	3	4	4	4	3	7	8	7
SVR	10	7	8	7	8	13	13	12
RF	15	15	14	14	15	17	18	15

(Continues)

TABLE B3 (Continued)

Model	1	2	3	6	12	3M	6M	12M
BART	13	11	10	11	10	12	12	11
Bagging	21	19	15	16	17	21	19	20
kNN	19	8	7	9	7	14	14	13
Huber	23	25	25	25	25	24	25	25
Theil-Sen	24	26	26	26	26	25	26	26
Factors	22	9	9	10	11	18	15	14
GARCH	4	12	18	19	12	4	3	4
VECM	5	23	23	23	23	8	7	18
SETAR	11	18	19	18	18	23	23	23
MA	14	14	13	8	9	6	6	6
SARIMA	18	21	21	22	21	5	4	3
ARFIMA	16	16	20	21	20	3	5	5
GradBoost	12	17	16	15	16	19	20	19
AdaBoost	17	10	12	12	14	16	16	16
Bayes Reg.	25	22	22	20	22	22	22	22

Note: The two best models are highlight in bold font.

APPENDIX C. OUT-OF-SAMPLE MAPE

Since MAPE is closely related to MAE and MSE, for the sake of brevity, we are here providing only the ranks of the models according to their respective MAPE (Table C1).

TABLE C1 Ranks produced by comparing the models according to their average mean absolute error (MAE) through the simulations.

Model	1	2	3	6	12	3M	6M	12M
LSTM	2	2	2	2	2	2	2	2
ConvLSTM	1	1	1	1	1	1	1	1
MLP	21	22	21	22	20	20	21	21
RW	26	25	24	24	24	26	24	24
Ridge CV	6	7	6	6	6	9	9	10
BRidge	7	8	7	9	7	15	17	17
LASSO CV	8	3	5	5	5	10	11	9
BLASSO	3	5	3	3	4	4	3	8
Enet CV	5	4	4	4	3	7	8	7
SVR	10	6	8	7	8	13	13	12
RF	15	15	14	14	15	17	18	18
BART	4	11	10	11	10	11	10	11
Bagging	20	19	15	16	17	21	19	20
kNN	19	13	11	13	13	14	14	13
Huber	23	24	25	25	25	24	25	25
Theil-Sen	25	26	26	26	26	25	26	26
Factors	22	9	9	10	11	18	15	14
GARCH	13	12	18	19	12	12	12	4

(Continues)

TABLE C1 (Continued)

Model	1	2	3	6	12	3M	6M	12M
VECM	9	23	23	23	23	8	7	15
SETAR	11	18	19	18	18	23	23	23
MA	14	14	13	8	9	6	6	6
SARIMA	18	21	17	17	21	5	4	3
ARFIMA	16	16	20	21	19	3	5	5
GradBoost	17	17	16	15	16	19	20	19
AdaBoost	12	10	12	12	14	16	16	16
Bayes Reg.	24	20	22	20	22	22	22	22

Note: The two best models are highlight in bold font.

APPENDIX D. OUT-OF-SAMPLE R²

Analogously to the case of MAPE, we are simply providing the ranks of each model according to their respective out-of-sample R^2 (Table D1).

TABLE D1 Ranks produced by comparing the models according to their average R^2 through the simulations.

Model	1	2	3	6	12	3M	6M	12M
LSTM	2	2	2	2	2	2	2	2
ConvLSTM	1	1	1	1	1	1	1	1
MLP	17	17	16	15	15	18	19	19
RW	25	22	22	22	22	22	22	22
Ridge CV	26	26	26	26	26	26	26	26
BRidge	9	6	5	5	5	10	9	8
LASSO CV	24	25	25	25	25	25	25	25
BLASSO	8	5	4	4	4	8	8	7
Enet CV	7	4	3	3	3	7	6	6
SVR	12	7	6	6	6	9	11	12
RF	16	14	11	12	12	14	15	13
BART	10	10	9	9	9	11	10	10
Bagging	18	18	14	14	14	17	18	18
kNN	19	8	7	7	7	12	12	11
Huber	21	23	23	23	23	23	23	23
Theil-Sen	22	24	24	24	24	24	24	24
Factors	20	11	8	10	10	15	16	14
GARCH	4	13	17	17	16	5	4	5
VECM	3	21	21	21	21	6	7	16
SETAR	15	19	19	18	17	21	21	21
MA	13	15	12	11	8	19	13	9
SARIMA	6	9	15	16	19	4	3	3
ARFIMA	5	3	18	19	18	3	5	4
GradBoost	11	16	13	13	13	16	17	17
AdaBoost	14	12	10	8	11	13	14	15
Bayes Reg.	23	20	20	20	20	20	20	20

Note: The two best models are highlight in bold font.