Borradores de ECONOMÍA

Colombian inflation forecast using Long Short-Term Memory approach

By:
Julián Alonso Cárdenas-Cárdenas
Deicy J. Cristiano-Botia,
Nicolás Martínez-Cortés

No. 1241
2023

BANCO DE LA REPÚBLICA
COLOMBIA

Colombian inflation forecast using Long Short-Term Memory approach

Julián Alonso Cárdenas-Cárdenas
Banco de la República
jcardeca@banrep.gov.co

Deicy J. Cristiano-Botia
Banco de la República
dcristbo@banrep.gov.co

Nicolás Martínez-Cortés
Banco de la República
nmartico@banrep.gov.co

Abstract

We use Long Short Term Memory (LSTM) neural networks, a deep learning technique, to forecast Colombian headline inflation one year ahead through two approaches. The first one uses only information from the target variable, while the second one incorporates additional information from some relevant variables. We employ sample rolling to the traditional neuronal network construction process, selecting the hyperparameters with criteria for minimizing the forecast error. Our results show a better forecasting capacity of the network with information from additional variables, surpassing both the other LSTM application and ARIMA models optimized for forecasting (with and without explanatory variables). This improvement in forecasting accuracy is most pronounced over longer time horizons, specifically from the seventh month onwards.

Keywords: Deep learning, Long Short Term Memory neural networks, forecast, inflation.

JEL Classification: C45, C51, C52, C53, C61, E37.

Pronóstico de la inflación en Colombia utilizando la metodología *Long Short-Term Memory*

Julián Alonso Cárdenas-Cárdenas
Banco de la República
jcardeca@banrep.gov.co

Deicy J. Cristiano-Botia
Banco de la República
dcristbo@banrep.gov.co

Nicolás Martínez-Cortés
Banco de la República
nmartico@banrep.gov.co

Resumen

A través de dos enfoques utilizamos redes neuronales *Long Short-Term Memory* (LSTM), una técnica de aprendizaje profundo, para pronosticar la inflación en Colombia con un horizonte de doce meses. El primer enfoque emplea solo información de la variable objetivo, la inflación, mientras que el segundo incorpora información adicional proveniente de algunas variables relevantes. Utilizamos *rolling sample* dentro del proceso tradicional de construcción de las redes neuronales, seleccionando los hiperparámetros con criterios de minimización del error de pronóstico. Nuestros resultados muestran una mejor capacidad de pronóstico de la red bajo el segundo enfoque, superando al primer enfoque y a modelos ARIMA optimizados para pronóstico (con y sin variables explicativas). Esta mejora en la capacidad de pronóstico es más pronunciada en horizontes más largos, específicamente entre el séptimo y doceavo mes.

Palabras clave: Aprendizaje profundo, redes neuronales *Long Short-Term Memory*, pronóstico, inflación.

Clasificación JEL: C45, C51, C52, C53, C61, E37.

1 Introduction

The behavior of prices, which is measured through inflation, is a critical variable in any economy. It determines the efficient allocation of resources, the purchasing power of the population and influences the investment decisions of different agents, thus affecting a broad set of macro and microeconomic variables, such as economic growth and the welfare of the population. Due to its relevance, institutions such as central banks have the inflation among their objectives. For example, in Colombia, Banco de la República (BR) is responsible for the monetary policy, including actions to maintain a low and stable inflation and to achieve sustainable levels of output and employment under an inflation targeting scheme. For these reasons, it is crucial for this institution to have accurate projections of inflation because they facilitate its correct reading and adequate monetary policy decisions.

In order to enhance its forecasting methodologies, the BR is always exploring new techniques and tools. This work, in particular, focuses on the application of artificial intelligence (AI) methods, which according to the literature, have demonstrated a high performance in predicting variables with similar characteristics: Long Short-Term Memory neural networks (LSTM). By incorporating AI methods, we aim to diversify the methods and further improve the accuracy of inflation projections.

Recent advances in pattern recognition in sequential data have provided a great opportunity to implement machine and deep learning methods for forecasting time series. Some of these methods have great potential as they deal with patterns based on a chain structure, incorporating various characteristics of the historical behavior of the series. One of them is the Recurrent Neural Network (RNN), which processes a time series step by step, summarizing its information and obtaining predictions using the lags of the same series[1]. A particular case is the LSTM, designed to also incorporate long-term dependency. In this case, the network uses a greater amount of information, remembering relevant events throughout its sequence, not just the most recently observed data. This has enabled better forecasting performance compared to traditional methodologies in many applications.

However, the strength of its forecast contrasts with its low capacity to identify the forces that define the projections, since these models extract the relevant information and use it through complex and non-linear interactions, which provides more accurate predictions but

[1] In most applications, it only uses information from the series to be forecast, however, it can include additional information from other sources.

complicates its interpretability[2]. In this sense, the objective of our application is only based on forecasting capabilities.

In the field of economic series forecasting, Siami-Namini et al. (2018) and Ülke et al. (2018) experimentally compared machine learning techniques with traditional methods. The first work shows a better performance of the LSTM algorithm than ARIMA models for a set of economic variables. The second work compares the behavior of machine learning models for the inflation forecast in the United States, taking different models for univariate and multivariate data. They found that machine learning models work better in the presence of volatility and irregular series when it comes to short-term forecasts.

For their part, in recent years international organizations and central banks have taken initiatives to incorporate machine learning techniques to evaluate and structure data (or text), find significant patterns and predict values. Some examples are Doerr et al. (2021), Chakraborty and Joseph (2017), and Rodríguez-Vargas (2020). The latest work is closely related to ours, it provides an evaluation of five machine learning methods applied to forecast inflation in Costa Rica, finding that LSTM shows the best results in all forecast horizons. Under our literature review, this is the first formal document to include deep learning techniques to forecast Colombian inflation, especially using LSTM.

We conduct two applications of this methodology, one using only information on the variable to be predicted, and another that incorporates information on relevant auxiliary variables due to their economic relationship. Moreover, we adjust the hyperparameters of the LSTM optimally, evaluating the root of the weighted mean square error (RWMSE) for rolling windows samples. We also compare the relative performance of the selected neural networks, with that obtained by ARIMA and ARIMAX models, also estimated under forecasting optimization criteria. Finally, we present an analysis for the period 2020-2022, relevant and challenging due to the strong shocks and uncertainty that accompanied this period and the need to evaluate how some forecasting techniques work in periods with more instabilities.

Overall, LSTM models performed better than comparable ARIMA models in almost all horizons. The LSTM with additional information outperformed the ARIMAX results in all forecast horizons (12 months). For its part, the LSTM without additional information outperformed to the ARIMA except for the first horizon, with significantly better performance after the fourth horizon. Finally, LSTM with additional information showed a clear improvement over LSTM (and ARIMA) with only information on the objective variable.

[2]Masini et al. (2021) shows that nonlinear ML models can be extremely useful for economic forecasting.

This document is organized into five sections, this introduction being the first. The second presents the data set used and some of its characteristics. The third section shows a description of the implemented methods, including the particular procedure used for the hyperparameterization of the LSTM models. The fourth section presents and compares the results of the models in terms of their forecasting capacity. In the last section we discuss the main conclusions.

2 Data set

The data revolve around headline inflation in Colombia, measured through the annual variation of the Consumer Price Index (CPI), which is calculated monthly by the National Administrative Department of Statistics (DANE). We have a monthly sample from January 2002 to December 2019. The starting point lies in a reasonable time after BR adopted the inflation targeting strategy and a flexible exchange rate system, characteristics that remain to this day. To avoid the atypical behavior of inflation caused by the COVID-19 pandemic, the sample for the construction of the LSTM ends before the pandemic began. We consider that once the shocks caused by the pandemic and other highly relevant recent ones such as those associated with the conflict between Russia and Ukraine are overcome, inflation will return to more traditional behavior, which we seek to capture in our LSTM forecasts. However, we conducted a separate exercise to assess the models just for the period from January 2020 to December 2022.

In one of the two LSTM applications, we include additional variables that we consider relevant to guide the short-term forecast (up to one year ahead): the Colombian Peso Market Exchange Rate, COP-USD (TRM), monthly average, calculated by BR with information from the Financial Superintendency of Colombia; the Real Gross Domestic Product of Colombia, seasonally adjusted and corrected for calendar effects (GDP), calculated by DANE[3]; and the precipitation gap in the country, calculated with data from the United States National Oceanic and Atmospheric Administration (NOAA)[4]. Figure 1 shows the behavior of all the variables.

[3]To incorporate GDP into the model, we transformed the quarterly value into one with monthly frequency. We did this based on the Economic Monitoring Indicator (ISE) behavior, also prepared by DANE. We do not directly use the ISE to take advantage of the GDP forecasts that are frequently produced in the BR.

[4]We define the precipitation gap as the difference between the monthly average level of precipitation in the country and the average precipitation for the same month during the previous ten years.

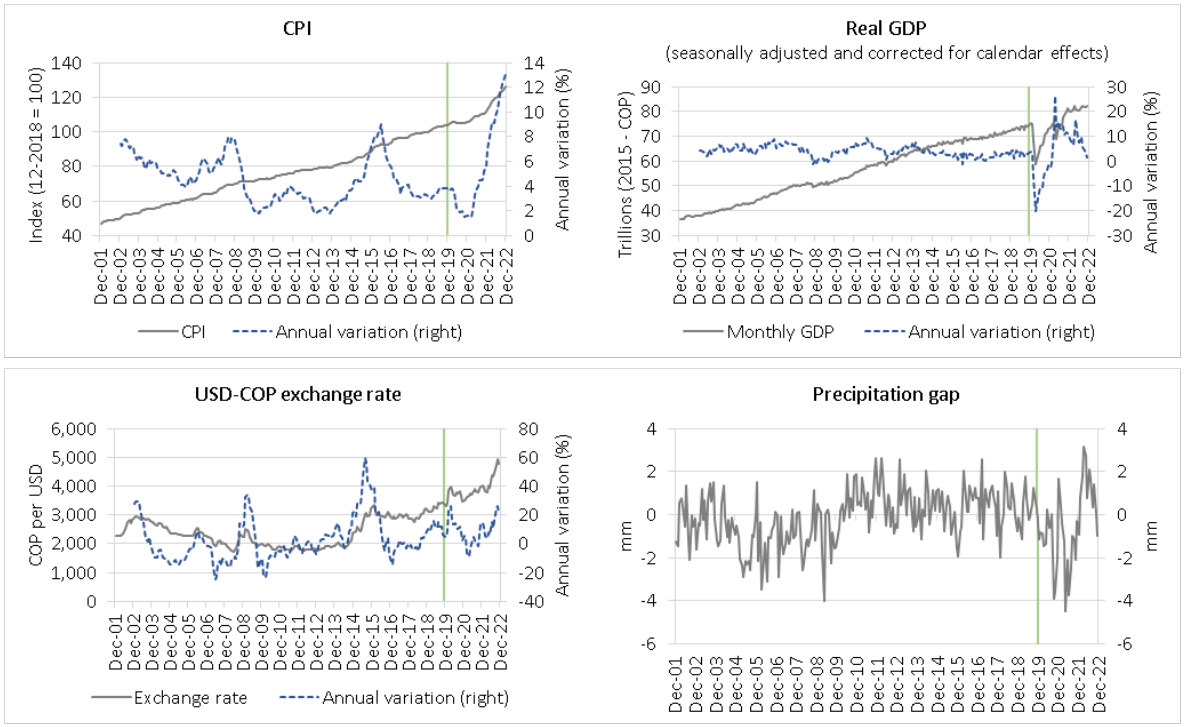

Figure 1: Variables of the models.

Variables reported in levels and in annual variations to facilitate the analysis. The green line is located at Dec-19.

The selection of these three variables is based on their relevant impact on the behavior of inflation in Colombia. The TRM can affect the CPI especially due to the transmission of the prices of imported goods and inputs (costs), due to the effect of competition between national and foreign goods (utility maximization) and because it is a condenser of various external shocks which can also affect investment, consumer confidence, expectations, etc. GDP, on the other hand, provides a picture of the state of demand in the economy. Finally, the precipitation gap captures the most frequent and relevant supply shocks that affect the headline inflation in Colombia[5]. While there are other variables that could be important in predicting inflation, we have chosen to focus on these three variables for simplicity and considering that these three variables may somehow reflect the effects of variables not taken into account. Future studies may consider evaluating other variables.

Regarding the treatment of the data, we previously standardized the series for the construction of the LSTM. On the other hand, we transform the variables by taking the logarithm and

[5]About it, Julio-Román et al. (2020), Bejarano-Salcedo et al. (2020), Abril-Salcedo et al. (2020) and Melo-Velandia et al. (2022) study the incidence of El Niño and/or La Niña weather phenomena on inflation in Colombia from different perspectives.

first differences according to each case for the ARIMA-type contrast models.

3 Method

In this section, we describe the LSTM methodology, detailing its key features, procedure and construction. We focus on the hyperparameter tuning process, which is complemented by the Bayesian optimization method. Our optimization criterion is the root of the out-of-sample weighted mean square error (RWMSE), since our objective is to improve forecast accuracy. Furthermore, we discuss our two LSTM applications: one that only uses inflation information data, and another that incorporates additional information from auxiliary variables. Finally, we introduce the ARIMA and ARIMAX methodologies, which we employ to evaluate the forecast quality of the LSTM models, using the root mean square error out of sample per forecast horizon.

3.1 Long Short Term Memory

Artificial Neural Networks (ANN) models have gained a lot of attention due to the rapid improvement of machine learning and deep learning algorithms, supported by the development of computational capacity, leading to their application in various research areas. This type of model tries to replicate the functioning of neuronal cells, simulating connections between them to take decisions. In statistical terms, these models extract the relevant information through correlations and interactions between the variables in a complex and non-linear way, making these models a particular class of non-linear parametric models (see Kuan and White (1994)). The general structure of ANNs has at least three layers of nodes (or layers of neurons, units or cells) through which information is processed: input nodes (grouped in the input layer), hidden nodes (hidden layer), and output nodes (output layer). The input layer supplies the information to be modeled; the hidden one (which can be more than one layer, each with multiple nodes) establishes the relevant relationships between the information in order to optimize the response in the output layer (see Figure 2). The number of hidden layers and nodes in each layer depends on the characteristics of the data and their relationships. Hyperparameters such as the number of epochs, the learning rate, and the dropout also contribute to this (we will address them in the section 3.2).

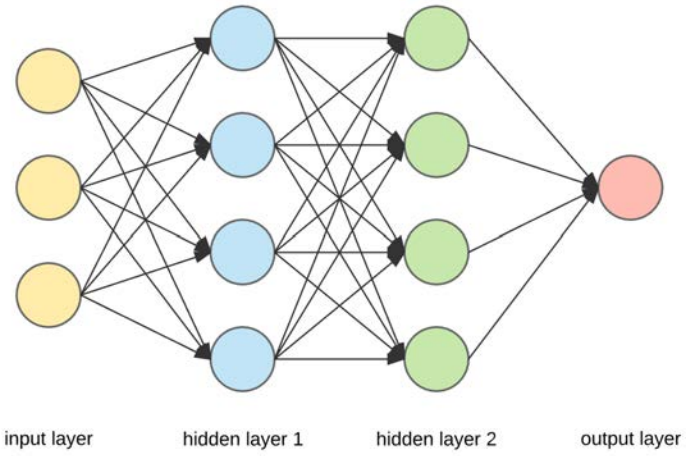

Figure 2: ANN basic components example.
Taken from upGrad (2020).

In the application to time series, the use of Recurrent Neural Networks (RNN) is stood out, a class of ANN that processes a sequence of inputs and retains part of its state while processing the next sequence of inputs, creating connections forward and backward. Its architecture has the form of a chain of repetitive neural network modules, where hidden nodes activate certain functions for each point in the data flow and store past information to predict the future. Again, in statistical terms, this model can be interpreted from the perspective of non-linear models; in this case, RNN looks for an optimal approximation of a dynamic system based on recursive patterns. In its architecture, lags are used as input to forecast future values, with an internal state that summarizes the information they have seen so far. The feedback loops of the recurrent cells inherently address the temporal order and the temporal dependencies of the sequences.

However, one drawback of the RNNs in time series is that they do not store a large amount of history, and they become highly dependent on the recent past. Given this, Hochreiter and Schmidhuber (1997) proposed the Long Short-Term Memory neuronal networks (LSTM), which explicitly address long-term dependency. Unlike normal RNNs, LSTMs have three different gates that control the flow of information (input[6], output and forget gates). They interact in a way that incorporates some of the information from memory over time and removes non-informative information to describe the behavior of the series. In this way, LSTMs maintain the chain-like structure, but the repeating modulus (cell) has a different form (Figure 3). The memorization of the data behavior is possible through the gates and

[6]Whose result interacts directly with additional information to update the state of the cell.

a memory line. It allows us to persist long-term states in addition to short-term states (for more details see Chollet and Allaire (2018)).

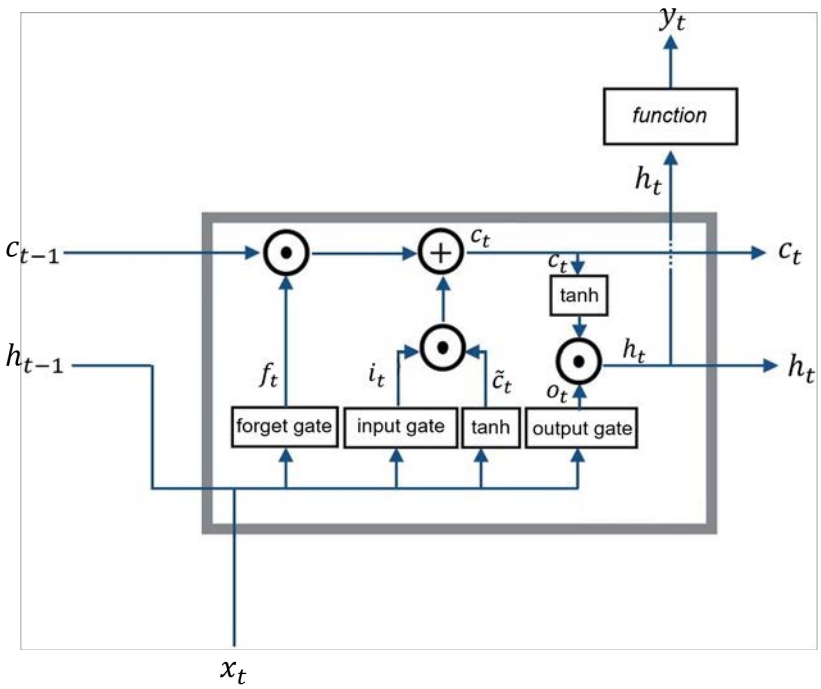

Figure 3: LSTM cell.
Adapted from Berhane (2019).

The equations 1 show the operation of the LSTM, following the formulation by Graves (2013).

$$\begin{aligned} f_t &= \sigma_g\left(W_f x_t + U_f h_{t-1} + b_f\right) \\ i_t &= \sigma_g\left(W_i x_t + U_i h_{t-1} + b_i\right) \\ o_t &= \sigma_g\left(W_o x_t + U_o h_{t-1} + b_o\right) \\ \tilde{C}_t &= \tanh\left(W_C x_t + U_C h_{t-1} + b_C\right) \end{aligned} \tag{1}$$

where x_t represents the input variables (lags of the target variable or from additional variables), W_f, W_i, W_o and W_C are the weight matrices mapping the hidden layer input to the gates and the input cell state, while the U_f, U_i, U_o and U_C are the weight matrices connecting the previous cell output state to the three gates and the input cell state. The b_f, b_i, b_o, and b_C are four bias vectors. The σ_g is the activation function, which normally is the sigmoid function, and the tanh is the hyperbolic tangent function. The role of these functions is to filter some of the information (useful/not useful) towards the next step. Based on the results of the four above equations, at each time iteration t, the cell output state, C_t, and the layer

output, h_t, can be calculated as follows:

$$\begin{aligned} C_t &= f_t C_{t-1} + i_t \tilde{C}_t \\ h_t &= o_t \tanh(C_t) \end{aligned} \tag{2}$$

The final output of a LSTM layer should be a vector of all the outputs, represented by $\boldsymbol{Y}_T = [h_{T-n}, \ldots, h_{T-1}]$. Here, when we are faced with the prediction problem, the last element of the output vector, h_{T-1}, is what we want to predict. Thus, the predicted value ($\hat{y}$) for the next time iteration, T, is h_{T-1}, namely $\hat{y}_T = h_{T-1}$.

3.2 Tuning of hyperparameters

Under the previous structure, the next step is to outline the process by which the key values for the operation of the model are determined, these values are the hyperparameter. They include the number of hidden layers, nodes in each layer, among others. To achieve this, we followed an iterative process (that include the Bayesian optimization) aimed to optimize a function (in our case the RWMSE), and in which other values affect, such as the learning rate, the number of epochs or the dropout.

In machine-learning and deep-learning algorithms, the hyperparameters need to be tuned to achieve optimal performances of the models. Hyperparameters are determined by finding its optimal values across multiple unknown values through an optimization procedure. We focused on tuning the number of epochs, which refers to number of times that all training data passes through the network to learn about it; the number of nodes and hidden layers (our application considers 1 and 2 hidden layers due to the characteristics of the time series to forecast); the learning rate which controls the speed at which the model adapts to the problem by determining the amount that the weights are updated during training; and the dropout which involves the probability of removing certain nodes or neurons from the network in order to verify the performance of a simpler network, it is a tool for fighting against overfitting, see Srivastava et al. (2014))[7].

This section presents the description of our procedure for tuning the hyperparameters mentioned. We use the RWMSE as the value of reference to analyze the performance of the models. We perform out-of-sample validation to find the set of hyperparameters that produce the lowest RWMSE through a rolling analysis.

[7] These are some of the most relevant hyperparameters for LSTM. For the others, we took the default values.

We follow the next procedure:

i. Take the (training) sample January 2002 to December 2015 and a set of possible values for the hyperparameters.

ii. Train the model and compute the predictions twelve steps ahead. Calculate and store the RMSE per horizon.

iii. Add one month to the (training) sample, repeat the procedure until taking December 2018 in the sample (with forecasts until December 2019).

iv. Average the RMSE between samples for each forecast horizon. Calculate and store the associated RWMSE.

v. Repeat the procedure changing the values of the hyperparameters until complete the grid defined in the following paragraphs.

vi. Choose the model with the hyperparameter values that generated the lowest RWMSE.

Looking for a trade-off between low RMSE and computational efficiency, a set of hyperparameters was established. We initially selected a range of values for these hyperparameters based on their characteristics and empirical evidence. A neural network with 1 or 2 hidden layers is typically adequate for less complex tasks, for example. In our case, since the length of the data is not excessively large, we explored the behavior of the neural network using a maximum of two hidden layers. Furthermore, as we expect that the consumer price index stays under control over time, the dynamic could be explained by some characteristics of the series with paths that do not require many combinations inside the optimization. Another factor is related to the computational efficiency of the process; the literature suggests using numbers of nodes in each layer ranging from 2 to 128 to improve this efficiency. To select the possible values of the dropout, we follow the suggestions from the literature in this kind of model. See Cheng et al. (2017) for details.

Taking into account the previous considerations, we evaluate the following values: one and two hidden layers, with 4, 16, 32, 64, and 128 nodes in each layer; for the number of epochs[8] we test with 20, 50, 100, 150 and 200, and for dropout percentage with 0.4, 0.3 and 0.2. We take the learning rate[9] as fixed (0.001) because the difference was slight when we made some

[8]The number of epochs vary depending on the complexity of the model and the appropriate number should be determined by experimantation.

[9]Typically it is sufficient to search a value in the range 10^{-6} to 10^{-2}, see Greff et al. (2015)

preliminary comparisons and because we already have a sufficient amount of hyperparameter combinations.

Once we have these first results, we analyze the patterns of hyperparameter combinations which produced the lowest RWMSE and we select a new range around the optimal values for each hyperparameter. We then run again the algorithm and evaluated the RWMSE. We found that while the sensitivity to hyperparameter tuning differed between wide ranges, there were no substantial disparities when we took a finer grid around specific values. It allowed us to have a trade-off between precision and computational efficiency, and gave us confidence in selecting the optimal hyperparameter values.

To complete the analysis, we took the Bayesian optimization approach, in which samples are generated inside a range of values for each hyperparameter and, through an iterative procedure, it provides the optimal combination to get the lowest RWMSE.

3.2.1 Bayesian optimization

We use the Bayesian optimization proposed by Snoek et al. (2012). This method has acquired great importance, especially now when the tuning of machine learning models is sometimes a difficult task to do. Then, we include this method because it is exhaustive evaluating all the combinations in some intervals. This procedure focuses on finding the minimum of a certain function $f(x)$ under a bounded set, let in this case, the set A. The Bayesian methods boost the finding of the optimal set of hyperparameters by creating a probabilistic model where the value of the objective function $f(x)$ is the metric of the model validation, in our case, the RMSE. For that purpose, we must make two main choices: the prior and the acquisition function.

First, to use the virtues of the Bayesian method, we must select a prior distribution. Snoek et al. (2012) suggests the use of "Gaussian process prior" because of its "flexibility and tractability", which for any finite set of N points in A induces a multivariate Gaussian distribution on R^N. On the other hand, the acquisition function guides the algorithm through the hyperparameters space, determining what points to be evaluated in the next steps. There are three different acquisition functions, Probability of Improvement (PI), Expected Improvement (EI), and GP Upper Confidence Bound. We follow Snoek et al. (2012), who use the EI criterion due to the fact it has shown better performance than the PI and does not require tuning its parameters. To explain this function, suppose that f' is the minimal value of $f(x)$

observed. The function used in EI maximizes the expected improvement (EI) over the f'. This corresponds to the following utility function:

$$EI_{f(x)} = max(0, f' - f(x)) \tag{3}$$

It means that the "improvement" is equal to $f' - f(x)$. If $f(x)$ turns out to be less than f'; then, there is no gain. Under a Gaussian prior, this function has a closed-form, and the EI acquisition function is the most common.[10] In this work, we use the rBayesianOptimization library (see Yachen (2021)).

3.2.2 Optimization criteria

As our main purpose is prediction, we take the Root Mean Square Error (RMSE) out of sample as the criterion of model performance. This criterion is defined as the Equation 4, where N represents the sample size, y_t the observed value at time t and $\hat{y}_t$ the forecast of y_t.

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(y_t - \hat{y}_t)^2} \tag{4}$$

We also compute a weighted RMSE giving more importance to the shortest horizons. With this measure, we assess the behavior and predictability, taking into account the forecast horizon ($h = 1, ..., 12$). Then, the Root Weighted Mean Square Error (RWMSE) for a model j is defined as the Equation 5.

$$RWMSE_j = \frac{\sum_h \frac{1}{h} RMSE_{jh}}{\sum_h \frac{1}{h}} \tag{5}$$

We use the RWMSE into the tuning process to select the best model for each methodology, and the RMSE per forecast horizon to compare among the selected models.

3.3 Two LSTM applications models

Following the LSTM approach, in this document we propose a neural network trained to predict twelve values ahead based on the diagnose of the patterns of the own lag of the time series (LSTM_FM) and another including some explanatory variables (LSTM_EM).

[10] For details see Snoek et al. (2012), where authors explain widely how to use this theory.

- **LSTM_FM** The first model we propose is based only on the history of the variable to be forecasted. In this case, the neural network inputs are the own lags of the time series without considering the effect of other variables.
- **LSTM_EM** In this second model, we also use information from additional variables. The rationalization for this methodology is that the effect of other variables is relevant to the future behavior of time series of interest. The algorithm works as described previously, including also the lags of the explanatory variables as inputs.

3.4 ARIMA Model

The main model in the traditional approaches for forecasting univariate time series is the autoregressive integrated moving average (ARIMA) model, so we took it as our baseline. The model is written as

$$\Delta^d y_t = c + \phi_1 \Delta^d y_{t-1} + \cdots + \phi_p \Delta^d y_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t \tag{6}$$

where $\Delta^d y_t$ is the d order differenced series. The predictors on the right-hand side include both lagged values of $\Delta^d y_t$ and lagged errors. We call this an ARIMA (p, d, q). A seasonal ARIMA model is formed by including additional seasonal terms in the ARIMA models we have seen so far. It is written as ARIMA$(p, d, q)(P, D, Q)_m$, where (p, d, q) represent the non-seasonal part of the model, (P, D, Q) is the seasonal part and m is the number of observations per year. We use the uppercase notation for the seasonal features of the model and lowercase notation for the non-seasonal parts of the model (for more details see Hyndman and Athanasopoulos (2018)).

An ARIMAX model is the ARIMA model with a new term for the exogenous variables (equation 7), in this case x_t is a vector that includes the three explanatory variables (exposed previously) with l number of lags (these variables can also be differentiated, especially to guarantee their stationarity).

$$\Delta^d y_t = \beta_1 x_t + ... + \beta_l x_l + \phi_1 \Delta^d y_{t-1} + \cdots + \phi_p \Delta^d y_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t \tag{7}$$

To determine the best model in each case, we employ an algorithm that estimates the models resulting from the possible combinations of parameters, explanatory variables, and number of lags in the ARIMA (ARIMAX) model. Similar to selecting the best LSTM model, the weighted root mean square error is used to identify the best combination of these aspects to produce the more appropriate model in terms of forecasting ability in the ARIMA and ARIMAX context.

4 Empirical Results

We followed the aforementioned approach to train the models and evaluate their performance based on the RWMSE. This section provides the key findings of our hyperparameters tuning results and show the selected models. We compare through the RMSE per forecast horizon, the LSTM_FM, which includes only its own lags as inputs, with the ARIMA model. Similarly, the LSTM_EM, which incorporates the explanatory variables, with the ARIMAX model. Additionally, we dedicate a subsection to evaluate the performance of the selected models during the complex period 2020-2022.

4.1 LSTM models

Figure 4 shows the training and testing datasets, the black line represents the training set, and the red line the testing set, both for the rolling approach. The graphs have a split based on 7-month jumps to show the differences in the training and test data over different periods. We take the last 12 observations as the testing set, with the same length as the number of horizons to forecast (h=12). We split the data in this way due to the relevance of the ordering into sequential data.

Figure 4: Illustration of the rolling sampling.

Notes: Each diagram (*Slice#*) represents, as an example, a step of the monthly rolling sampling. Each step is made up of a training sample (black line) and its 12-month test sample (red line).

We focus on selecting the number of layers, the number of nodes inside each layer, the number of epochs and the dropout rate, but for practicality we do not report the iterations for this last hyperparameter. Figure 5 reports LSTM_FM performance based on a model with two hidden layers and different numbers of epochs. Both hidden layers include different specifications for the number of nodes. The colors in the visualization indicates the RWMSE metric, where the lightest boxes correspond to the lowest RWMSE, and the darkest ones indicates the highest RWMSE. Our results reveal that the most accurate forecasts are situated in the left panel (20 epochs), regardless the number of nodes within each layer. Concerning the number of nodes, the best performance is obtained by models with few nodes in each layer (bottom boxes).

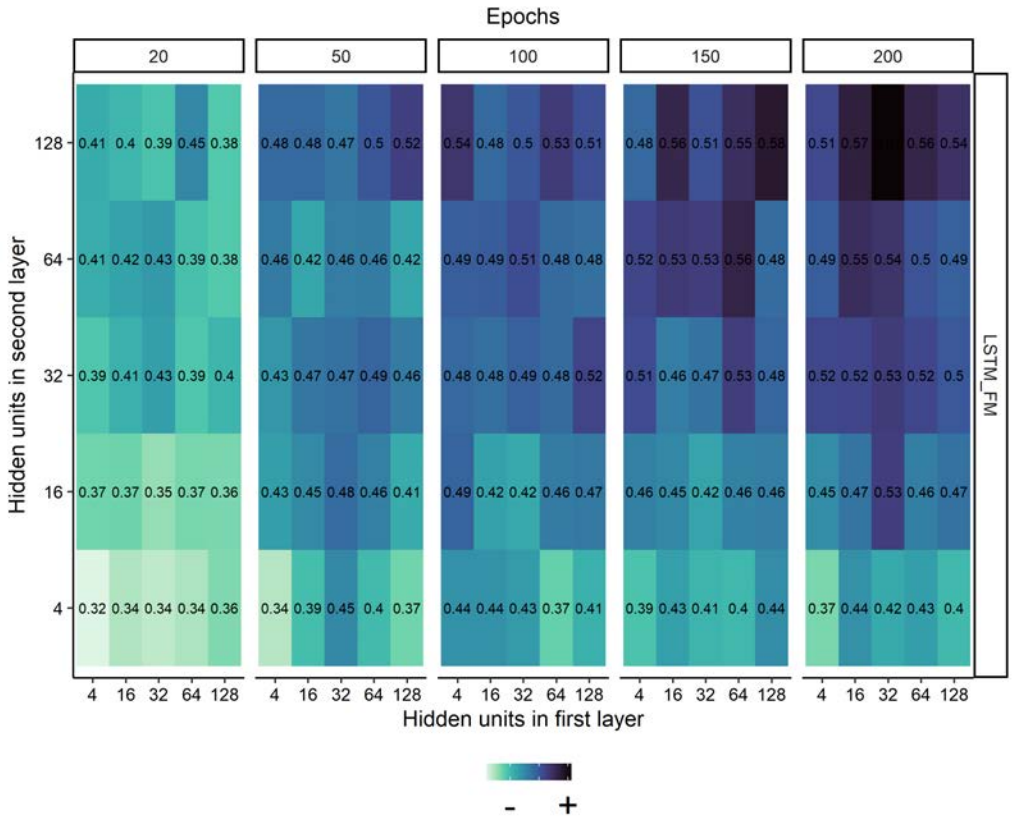

Figure 5: RWMSE for the LSTM_FM with two hidden layers and different nodes and epochs.
Notes: The numbers and the colors represent the RWMSE values. Lighter green colors represent lower values, darker blue colors higher values.

However, by emphasizing the number of hidden layers, we generally get similar or worse results as this number increases (Figure 6). The LSTM_EM that uses two hidden layers requires a lot of computational power and in general its results are not good enough, while

for the LSTM_FM there seem to be not relevant differences. This is the first difference between LSTM_EM and LSTM_FM; the performance of the former one deteriorates when a more complex architecture is used. In detail, Figure 6 shows the RMSE for LSTM_FM and LSTM_EM at different forecast horizons. The left panel shows the RMSE for all horizons using one hidden layer, while the right panel shows performance with two hidden layers. In general, the model with one layer and explanatory variables is better in several horizons. The most notable differences appear at longer horizons, where the LSTM with explanatory variables and a hidden layer seems to be the best. Specifically for long horizons, the results suggest that the explanatory variables included in the network provide valuable information to predict inflation in Colombia. Consequently, a not so complex model, with a low number of epochs and nodes would be adequate to forecast our Colombian inflation time series. For this reason, we select a single layer and continue with the selection of the number of nodes and epochs (Figure 7).

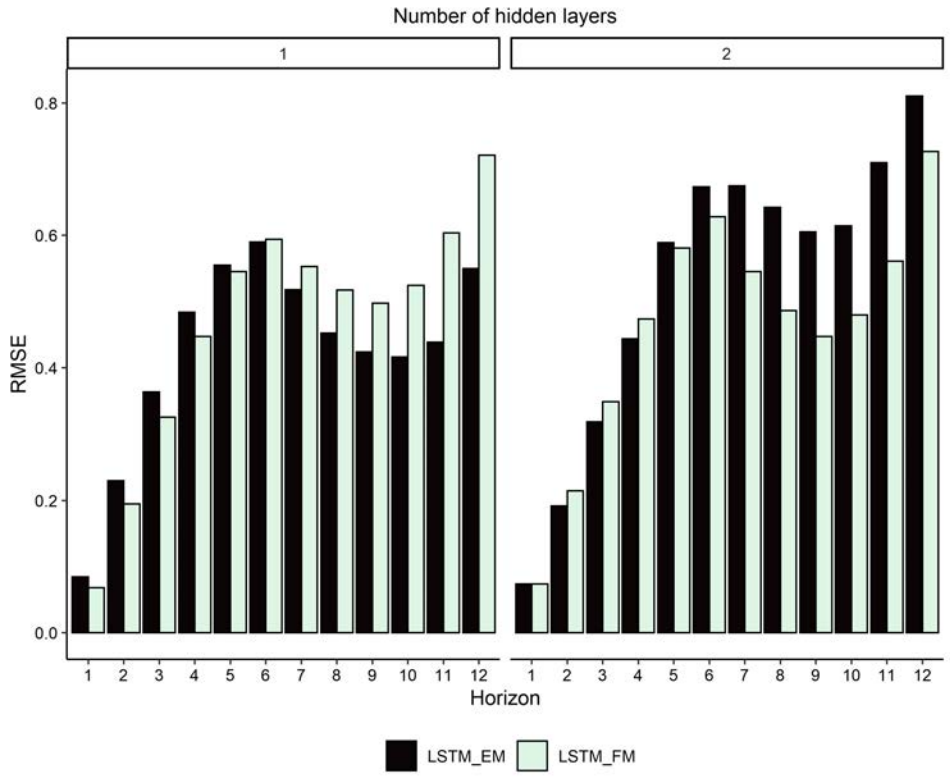

Figure 6: RMSE for LSTM_FM and LSTM_EM by horizon of prediction.

The Figure 7 is constructed using different numbers of nodes for both LSTM models with one hidden layer. It proves again that a low range of epochs is convenient for this data, as well

as that only a small number of nodes is required for both models. Additionally, the number of epochs plays a major role only in the LSTM_FM; we see that the models with many nodes get worse when they have more epochs. Unlike the LSTM with no explanatory variables, the LSTM_EM gets similar forecasting performance across the number of nodes and epochs (see bottom panel of the plot).

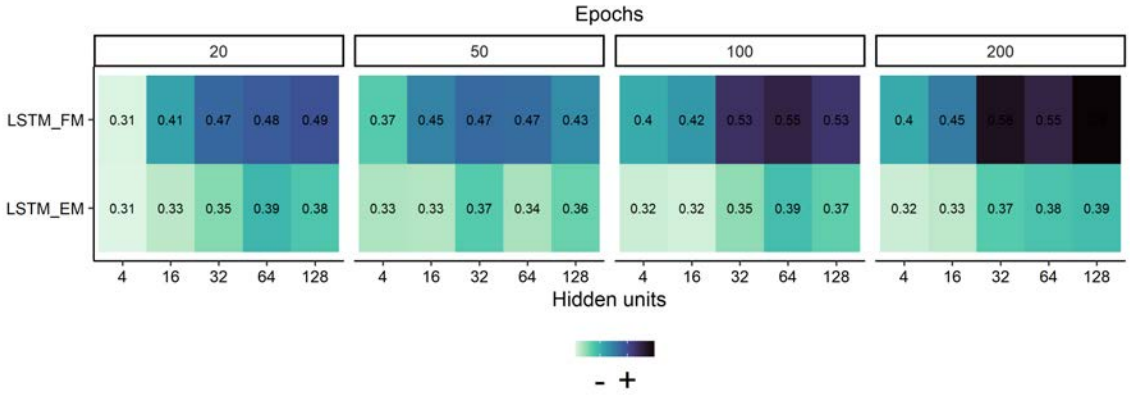

Figure 7: RWMSE for the LSTM_FM and the LSTM_EM with one hidden layer and different numbers of nodes and epochs.

Notes: The numbers and the colors within the graph represent the RWMSE values. Lighter green colors represent lower values, darker blue colors higher values.

We evaluate our models on the continuous-valued hyperparameters around the values founds more plausible according to the previous results. It was run following the Bayesian optimization; Table 1 shows the best combination of values of those hyperparameter with the best performance according to the lowest RWMSE. Thus, the algorithm converges to values close to those found previously. With these values, we run the LSTM models and compare the RWMSE for each model.

Hyperparameter	LSTM_FM	LSTM_EM
Hidden layers	1	1
Unit	13	6
Epoch	31	50
Dropout	0.2	0.1
Learning rate	0.001	0.001

Table 1: Results from Bayesian Optimization.

4.2 Comparison between LSTM and ARIMA models

Our results indicate that LSTM performs better than ARIMA models, particularly for longer horizons. Table 2 shows the RMSE for the two models, LSTM considering only lags of the CPI (LSTM_FM), and the LSTM taking information of explanatory variables (LSTM_EM). Between these two models, in general the LSTM_EM presents better performance. We also compare with the ARIMA models by calculating the percentage of reduction in the RSME for each LSTM model (ARIMA vs. LSTM_FM, ARIMAX vs. LSTM_EM, last two columns of the Table, respectively). We observe that when using LSTM-FM, there is a reduction in the RSME of around 7% to 65% as compared to ARIMA model, except for the first horizon. Moreover, the reduction in the RSME becomes progressively larger as the horizon becomes longer. When exploratory variables are included, the RMSE percentage reduction ranges from 7% to 68%, once again showing a greater reduction in longer horizons. Since the baseline model is the ARIMA model in both cases, we can argue that for this time series, we get a better performance in terms of low RMSE by using LSTM method, especially at longer horizons.

	(I)	(II)	(III)	(IV)	% Reduction	
Horizon	LSTM_FM	ARIMA	LSTM_EM	ARIMAX	(I) vs. (II)	(III) vs. (IV)
1	0.24	0.21	0.15	0.17	14.29	-11.76
2	0.41	0.44	0.25	0.31	-6.82	-19.35
3	0.53	0.70	0.36	0.41	-24.29	-12.20
4	0.63	0.95	0.44	0.50	-33.68	-12.00
5	0.72	1.24	0.57	0.61	-41.94	-6.56
6	0.77	1.49	0.60	0.68	-48.32	-11.76
7	0.77	1.66	0.58	0.75	-53.61	-22.67
8	0.76	1.86	0.53	0.83	-59.14	-36.14
9	0.76	2.04	0.46	0.92	-62.75	-50.00
10	0.76	2.20	0.39	0.98	-65.45	-60.20
11	0.80	2.29	0.35	1.02	-65.07	-65.69
12	0.83	2.36	0.34	1.05	-64.83	-67.62
RWMSE	0.50	0.87	0.32	0.45	-	-

Table 2: RMSE for each horizon of prediction by model.

These results are also displayed in the Figure 8, where we can see the trend of the RMSE for each model graphically. First, we observe that in all models, the lowest RMSE is obtained at the beginning of the prediction horizon; but it gradually increases as the horizon of prediction

gets longer. This is explained by the rise of uncertainty in longer-term predictions. However, the LSTM model with explanatory variables shows and exception to this trend. According to the graph, after the sixth horizon, we observe a soft decline in RMSE until the tenth horizon. This could indicate strong relations between the explanatory variables and the inflation rate in different lags captured by the LSTM. Secondly, this LSTM works better in long-term horizons, considering the RMSE. In horizon twelve it obtains a 68% improvement compared to the ARIMAX and 59% compared to the LSTM without explanatory variables. It is also important to note that despite differences at the beginning, the short-term dynamics are nearly identical for all models. Where the ARIMA model outperforms the LSTM model, both have RMSE very similar, which suggests that the performance of the LSTM is better or at least similar to the ARIMA model. Therefore, the LSTM models seem to be a proper tool to forecast the future behavior of the inflation rate, especially when including information from explanatory variables in the neural network.

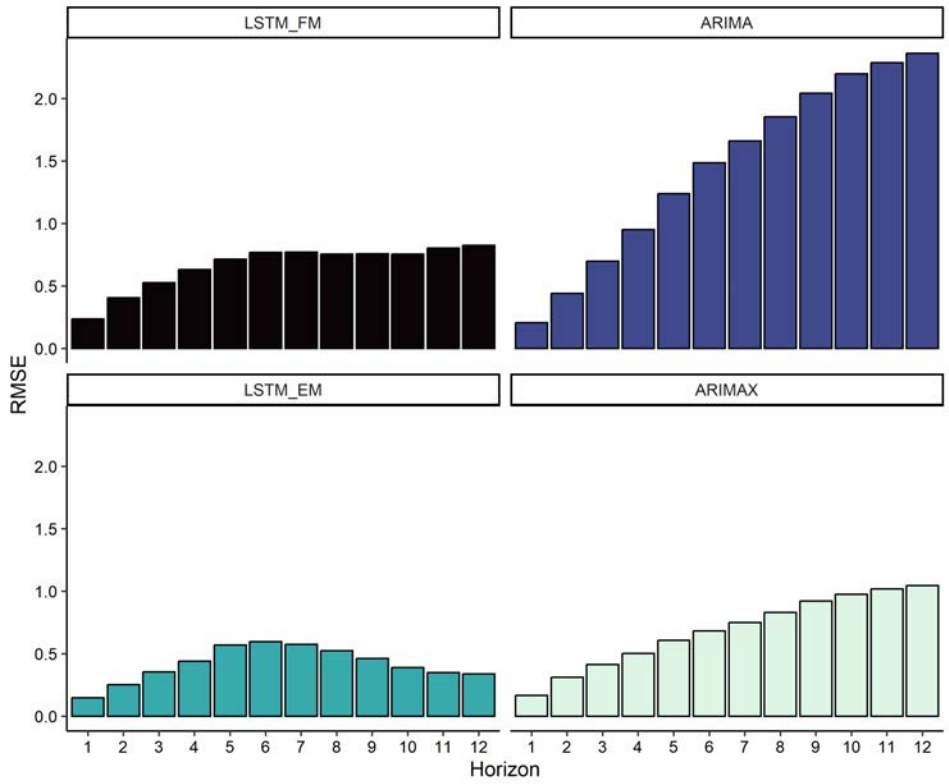

Figure 8: RMSE for horizon of prediction by model.

To further support these findings, we compared the forecasting results obtained by LSTM algorithm and traditional models using the Diebold-Mariano test. Our objective was to de-

termine the horizons in which the LSTM produces statistically more accurate forecasts than the ARIMA models. The table presents the *p-value* for each horizon of prediction, we again compare LSTM_FM with ARIMA model, and LSTM_EM with ARIMAX model. The results of the test indicate that at 10% significance level, there is a significant difference in forecast accuracy between LSTM_FM and ARIMA model in horizons greater than 4, while there are significant differences between LSTM_EM and ARIMAX model after the horizon 7. This validates our earlier discussion of larger notable differences at longer-term horizons.

	p-value	
Horizon	LSTM_FM	LSTM_EM
1	0.84	0.40
2	0.40	0.20
3	0.11	0.20
4	0.07	0.12
5	0.04	0.10
6	0.02	0.10
7	0.01	0.09
8	0.01	0.08
9	0.01	0.07
10	0.01	0.06
11	0.01	0.06
12	0.01	0.06

Table 3: Diebold-Mariano test for each horizon of prediction, LSTM_FM compared to ARIMA, and LSTM_EM compared to ARIMAX model.

Finally, a rolling analysis is presented to complete the comparison of the forecasting performance of the models exposed in this document. In this exercise, we evaluate the forecasting from 2016 to 2020, always taking twelve steps ahead. Figure 9 shows the behavior of the predictions by the model after adding one month of information in each step. Because LSTM is learning in every step, we highlight the behavior of the first period, where the predictions are far away from the observed values. However, including more and more information, the forecast follows the closest path of the observed data. It illustrates the learning process and shows that the forecast is aligned with the trend of the real data, specifically when more information is included in the training step. The results of the whole sample discussed above are validated. We highlight that the forecastings are around the observed inflation rate, especially using the LSTM_EM (bottom panel). This is important when we analyze the predictions of the LSTM, since it helps to observe the potential bias introduced by its non-linearities in the

LSTM. From the visualization, we observe that the target variable remained in the middle of the forecasting range during the period of analysis, which indicates that the model did not consistently over or under predict the inflation rate. The ARIMA models show a less satisfactory performance, apparently with a high relevance of the short-term autoregressive component, especially in its version without explanatory variables. This would cause them to replicate the most recently observed behavior, making it difficult to identify changes in the behavior of the series.

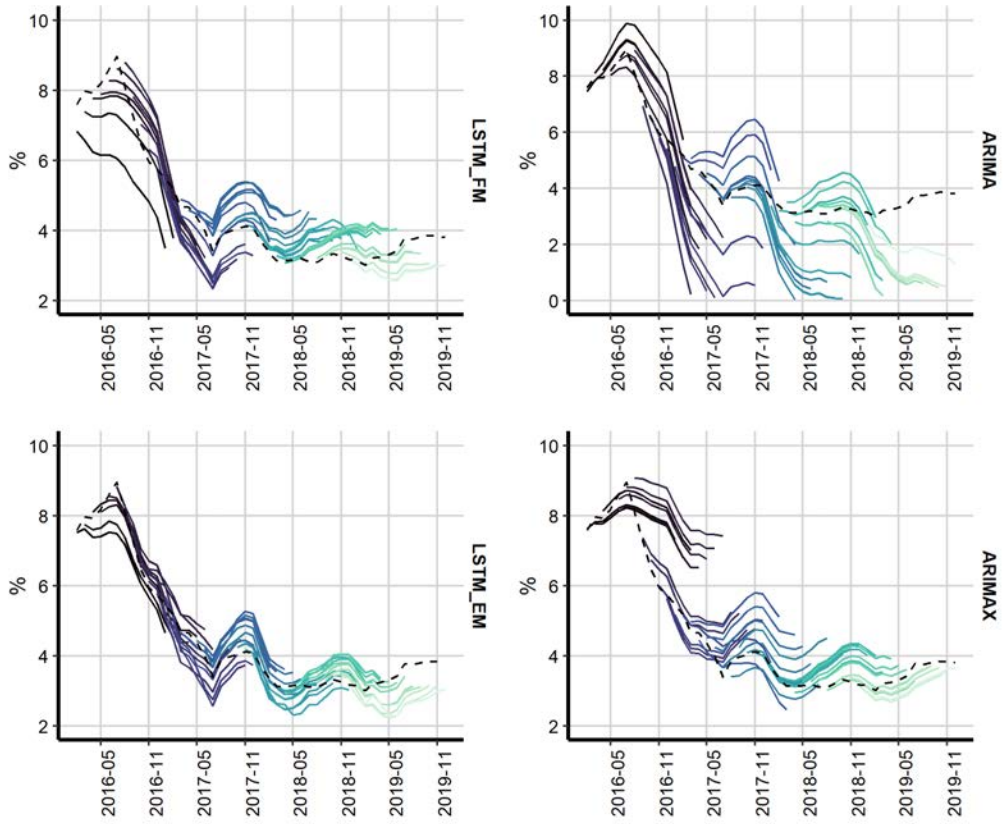

Figure 9: Rolling performance by model.

4.3 Performance in the 2020-2022 period

Finally, this section analyzes the forecasts for the period January 2020 to December 2022 obtained by the selected models. This is a period in which the world economy experienced never-before-seen shocks, mainly caused by the Covid-19 pandemic and external conflicts. These shocks generated atypical behaviors in a large part of the economic variables, including

inflation. These conditions motivated the decision of this paper to exclude this period from the previous rolling analysis, also hoping that once the shocks are overcome, inflation will return to a behavior similar to that before the pandemic.

However, it is useful to know the performance of the models in this period. Since LSTM models let the data decide what model works best for prediction, we want to see how fast the model incorporates the shocks and approaches the data in the short-term forecastings. The results of all the implemented models in terms of the RMSE metric are as shown in Table 4. We observe that, in general, the range of the RMSE for each model increases, with the RMSE for the long horizons quite high with respect to the previous results. Moreover, LSTM_FM once again gives better results after the second horizon with a small difference in the first horizon. The improved reduction of the RMSE of the LSTM_EM is given in the last prediction horizons, as we mentioned in the analysis until 2020. Consequently, the forecasting performances of the LSTM models are better or similar for the this period as well.

	(I)	(II)	(III)	(IV)	% Reduction	
Horizon	LSTM_FM	ARIMA	LSTM_EM	ARIMAX	(I) vs. (II)	(III) vs. (IV)
1	0.34	0.32	0.36	0.29	6.25	24.14
2	0.61	0.70	0.68	0.61	-12.86	11.48
3	0.94	1.06	1.00	0.91	-11.32	9.89
4	1.28	1.51	1.31	1.25	-15.23	4.80
5	1.59	1.98	1.62	1.60	-19.70	1.25
6	1.91	2.47	1.88	1.90	-22.67	-1.05
7	2.21	3.00	2.14	2.19	-26.33	-2.28
8	2.56	3.48	2.48	2.52	-26.44	-1.59
9	2.94	4.10	2.83	2.90	-28.29	-2.41
10	3.30	4.72	3.18	3.31	-30.08	-3.93
11	3.69	5.37	3.52	3.71	-31.28	-5.12
12	4.12	6.05	3.84	4.11	-31.90	-6.57

Table 4: Results for the 2020-2022 period.

5 Conclusions

This work uses deep learning algorithms, specifically the Long Short Term Memory neural network, to forecast consumer inflation in Colombia. Their results are compared with those

obtained by ARIMA models through RMSE, noting that LSTM models improve the forecasting capacity of ARIMA specifications. Given this, it is reasonable to consider the application of these novel techniques in other economic series for forecasting problems.

One of our findings is related to the modeling of inflation under these techniques. We find that increasing the complexity of the model by adding hidden layers or increasing the number of nodes or epochs does not necessarily lead to better results in terms of forecasting ability. In fact, the model with the lowest RMSE had only one hidden layer and a small number of nodes. This may be related to some characteristics of the series, such as its relatively short length (216 observations) and its apparently not very complex behavior in the context of deep learning. As a result, the neural network quickly learns the dynamics of the series.

The empirical results show that the LSTM models outperform the ARIMA models, being particularly clear in the longer forecast horizons. In addition, the results show the importance of including auxiliary or explanatory variables potentially related to the series to be forecast, in this case, inflation. The model with explanatory variables as neural network inputs markedly improves the prediction performances, and again is more noticeable at longer horizons.

In terms of the RMSE, we find a reduction of 7% for the second horizon (forecast month) and 65% for horizon 12 using the LSTM with its own lags compared to the ARIMA model. For its part, the LSTM with explanatory variables shows reductions of up to 68% in the longest horizon compared to the ARIMAX model (and 59% compared to the first LSTM).

Finally, a separate analysis was carried out for the period 2020-2022, characterized by strong shocks to the world economy. As expected, all the models got worse in their forecasting capacity metrics, however, better results remain for the LSTM than for the ARIMA techniques, and for the LSTM with explanatory variables than for the LSTM without them, particularly on the longer horizons.

References

Abril-Salcedo, D. S., Melo-Velandia, L. F., and Parra-Amado, D. (2020). Nonlinear relationship between the weather phenomenon el niño and colombian food prices. *Australian Journal of Agricultural and Resource Economics*, 64(4):1059–1086.

Bejarano-Salcedo, V., Caicedo-García, E., Lizarazo-Bonilla, N. F., Julio-Román, J. M., and

Cárdenas-Cárdenas, J. A. (2020). Hechos estilizados de la relación entre el niño, la niña y la inflación en colombia. *Borradores de Economía; No. 1105*.

Berhane, F. (2019). Building your recurrent neural network - step by step. `https://datascience-enthusiast.com/DL/Building_a_Recurrent_Neural_Network-Step_by_Step_v1.html` [Accessed: August 2021].

Chakraborty, C. and Joseph, A. (2017). Machine learning at central banks. Bank of England working papers 674, Bank of England.

Cheng, G., Peddinti, V., Povey, D., Manohar, V., Khudanpur, S., and Yan, Y. (2017). An exploration of dropout with lstms. In *Interspeech*, pages 1586–1590.

Chollet, F. and Allaire, J. (2018). *Deep Learning with R*. Manning Publications.

Doerr, S., Gambacorta, L., and Garralda, J. M. S. (2021). Big data and machine learning in central banking. BIS Working Papers 930, Bank for International Settlements.

Graves, A. (2013). Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. (2015). LSTM: A search space odyssey. *CoRR*, abs/1503.04069.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9:1735–80.

Hyndman, R. and Athanasopoulos, G. (2018). *Forecasting: principles and practice,*. OTexts: Melbourne, Australia. https://otexts.com/fpp2/.

Julio-Román, J. M., Bejarano-Salcedo, V., Caicedo-García, E., and Cárdenas-Cárdenas, J. A. (2020). Entendiendo, modelando y pronosticando el efecto de "el niño" sobre los precios de los alimentos: El caso colombiano. *Borradores de Economía; No. 1102*.

Kuan, C.-M. and White, H. (1994). Artificial neural networks: an econometric perspective. *Econometric Reviews*, 13(1):1–91.

Masini, R. P., Medeiros, M. C., and Mendes, E. F. (2021). Machine learning advances for time series forecasting.

Melo-Velandia, L. F., Orozco-Vanegas, C. A., and Parra-Amado, D. (2022). Extreme weather events and high colombian food prices: A non-stationary extreme value approach1. *Agricultural Economics*, 53(S1):21–40.

Rodríguez-Vargas, A. (2020). Forecasting Costa Rican inflation with machine learning methods. *Latin American Journal of Central Banking (previously Monetaria)*, 1(1).

Siami-Namini, S., Tavakoli, N., and Namin, A. S. (2018). A comparison of arima and lstm in forecasting time series. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 394–1401. IEEE.

Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. *arXiv preprint arXiv:1206.2944*.

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting.

Ülke, V., Sahin, A., and Subasi, A. (2018). A comparison of time series and machine learning models for inflation forecasting: empirical evidence from the USA. *Neural Comput. Appl.*, 30(5):1519–1527.

upGrad (2020). Neural network: Architecture, components and top algorithms. `https://www.upgrad.com/blog/neural-network-architecture-components-algorithms/` [Accessed: August 2021].

Yachen, Y. (2021). *Package 'rBayesianOptimization': Bayesian Optimization of Hyperparameter*. `https://cran.r-project.org/package=rBayesianOptimization`.


BANCO DE LA REPÚBLICA
COLOMBIA