COMMODITY PRICE PREDICTION USING DEEP LEARNING APPROACH

*Report submitted to*

*Haldia Institute of Technology,*

*Haldia for the award of the degree*

*Of*

Bachelor Of Technology in Computer Science

*by*

Abhay Khawas

Shubham Kumar

Shubham Vijoy

Rohan Kumar



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

HALDIA INSTITUTE OF TECHNOLOGY, HALDIA JUNE

2022

**DECLARATION**

a. The work contained in this report is original and has been done by us under the guidance of our supervisor(s).

b. The work has not been submitted to any other Institute for any degree or diploma.

c. We have followed the guidelines provided by the Institute in preparing the report.

d. We have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

e. Whenever we have used materials (data, theoretical analysis, figures, and text) from other sources, we have given due credit to them by citing them in the text of the report and giving their details in the references.

Signature of the Students

**CERTIFICATE**

This is to certify that the Dissertation Report entitled, “**COMMODITY PRICE PREDICTION USING DEEP LEARNING APPROACH**” submitted by Abhay Khawas (10300118142), Shubham Kumar (10300118140), Shubham Vijoy (10300118141), Rohan Kumar (10300118139) to Haldia Institute of Technology, Haldia, India, is a record of bonafide Project work carried out by him/her under my/our supervision and guidance and is worthy of consideration for the award of the degree of Bachelor of Technology in Computer Science and Engineering of the Institute.

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

Supervisor Supervisor

DATE :

**ACKNOWLEDGEMENT**

We would like to acknowledge the contributions of the following people without whose help and guidance this report would not have been completed. We acknowledge the counsel and support of our training coordinator, Mrs. Rajrupa Metia, Assistant Professor, CSE Department, with respect and gratitude, whose expertise, guidance, support, encouragement, and enthusiasm has made this report possible. Her feedback vastly improved the quality of this report and provided an enthralling experience. We are indeed proud and fortunate to be supported by her. We are also thankful to Prof. (Dr.) Subhankar Joardar, H.O.D of Computer Science Engineering Department, Haldia Institute of Technology, Haldia, WB for his constant encouragement, valuable suggestions and moral support and blessings. Although it is not possible to name individually, we shall ever remain indebted to the faculty members of Haldia Institute of Technology Haldia, WB for their persistent support and cooperation extended during this work. This acknowledgement will remain incomplete if we fail to express our deep sense of obligation to our parents and God for their consistent blessings and encouragement

**Table of Contents**

Chapter Page

1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .1

1.1 Commodities, Markets and Profits . . . . . . . . . . . . 1

1.2 Time series and financial analysis . . . . . . . . . . . . .4

1.3 Methodology . . . . . . . . . . . . . . . . . . . . . . . . . . . . .4

2 Pure Neural Network Approach . . . . . . . . . . . . . . . . 7

2.1 Background on Neural Networks . . . . . . . . . . . . . 7

2.2 Neural Networks and Financial Time Series Analysis 9

2.3 Software used / Methodology . . . . . . . . . . . . . . . .9

2.4 Results / Conclusions . . . . . . . . . . . . . . . . . . . . . .11

3 Use a Genetic Algorithm to evolve architecture for a

neural network . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

3.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .16

3.2 Background on Genetic Algorithms . . . . . . . . . . . 17

3.3 Using a Genetic Algorithm to Evolve Parameters

for Neural Networks . . . . . . . . . . . . . . . . . . . . . . . . . . 19

3.4 GA evolved NN Parameters in Finance . . . . . . . . 21

3.5 Software used / Methodology . . . . . . . . . . . . . . . . 22

3.6 Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .24

4 Use a Genetic Algorithm to evolve the weights for a

neural network. . . . . . . . . . . . . . . . . . . . . . . . . . . . . .26

4.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26

4.2 Background on using a GA to evolve weights for

NN . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .27

4.3 Software used / Methodology . . . . . . . . . . . . . .28

4.4 Results / Conclusions . . . . . . . . . . . . . . . . . . . . 30

5 Summary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32

Bibliography . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .. 34

ABSTRACT

This paper applies a recurrent neural network (RNN) method to forecast cotton and oil prices. We show how these new tools from machine learning, particularly Long-Short Term Memory (LSTM) models, complement traditional methods. Our results show that machine learning methods fit reasonably well with the data but do not outperform systematically classical methods such as Autoregressive Integrated Moving Average (ARIMA) or the naïve models in terms of out of sample forecasts. However, averaging the forecasts from the two types of models provide better results compared to either method. Compared to the ARIMA and the LSTM, the Root Mean Squared Error (RMSE) of the average forecast was 0.21 and 21.49 percent lower, respectively, for cotton. For oil, the forecast averaging does not provide improvements in terms of RMSE. We suggest using a forecast averaging method and extending our analysis to a wide range of commodity prices.

Acknowledgment

This work was undertaken as part of the CGIAR Research Program on Policies, Institutions, and Markets (PIM) led by the International Food Policy Research Institute (IFPRI). This paper has not gone through IFPRI’s standard peer-review procedure. The opinions expressed here belong to the authors and do not necessarily reflect those of PIM, IFPRI, or CGIAR.

Chapter 1

Forecasting commodity prices is paramount for many economic actors. When building budgets, experts rely on growth projections at the government level, which are almost always based on the country’s underlying forecasts of primary commodities exported. Oil-dependent countries represent a typical example where this kind of scenario is encountered. Many depend on a few raw materials (agricultural and minerals) for developing countries, the price of which determines the growth rate. Being able to forecast commodity prices is vital for other public entities or parastatals too. In many developing countries, parastatals are managing stabilization funds aimed at smoothing commodity price movements. These entities need world price forecasts in order to fix producer prices for the current campaign. For researchers, knowing the best data generating process and the forecast errors is essential for various modeling purposes. For instance, in agricultural models involving expectations1, the expected price is generally given by an ARMA model (Antonovitz and Green, 1990; De Janvry and Sadoulet, 1995). As we will show later, LSTM networks provide a good alternative to ARMA models.

Since the seventies, Box-Jenkins (1970)2 approaches have been popular in forecasting time series. These approaches have introduced ARMA models and their extensions as the cornerstone of forecasting tools. However, machine learning methods that can handle time-series data and perform forecasting have grown over the last three decades. Among all methods, those based on Recurrent Neural Networks are particularly interesting as they can carry over information (memory) from previous periods into the future. Early papers using RNNs to forecast time series include Kamijo and Tanigawa (1990), Chakraborty et al. (1992), and a comparison with ARIMA models by Kohzadi et al. (1996). However, one issue encountered with the first generation of RNN is the so-called vanishing gradient problem for highly dependent (long memory) data. Thus, in a seminal paper, Hochreiter and Schmidhuber (1997) proposed a new approach called Long Short-term Memory with the capacity to filter out which information from the past should be processed and retained3. This triggered new literature on forecasting time series (Gheyas and Smith 2011; Khandelwal et al., 2015; Kumar et al., 2018).

This paper aims to focus on LSTM models as they present a series of interesting characteristics. First, they are non-parametric so that they can handle suitably non-linear patterns. Second, they do not require the error term to follow a distribution. Third, LSTM models do not require the underlying data to follow a stationary process, so they are not affected by unit-roots. These three features present an interesting advantage over regression-based methods, be they ARIMA or not. We focus on two commodities widely analyzed in the literature: cotton and oil.

As previously mentioned, both commodities are important for a broad group of countries, particularly in the developing world, as net importers or net exporters for their energy bill or as a source of foreign exchange. Cotton has been at the center of hard talks at WTO involving many African countries

and Brazil versus the United States and the EU. Understanding the evolution of prices was key in that debate, and previous studies showed that the movements of prices are very complex and characterized by nonlinearities in their dynamics (Deaton and Laroque, 1992). The same results hold for oil prices also subject to strong nonlinearities (Moshiri and Foroutan, 2006; Kisswani and Nussair, 2013), which require non-parametric methods such as LSTM models. Using monthly cotton and oil prices, our results suggest that the LSTM model fits the data reasonably well but does not outperform the ARIMA or naïve models for out of sample forecasts. However, a combination of forecasts from the two models yields better results for the cotton dataset than either approach.

The remainder of this paper is organized as follows. We first present in section 2 the traditional Recurrent Neural Networks (RNN), and then, the LSTM approach proposed to solve the vanishing gradient problem of the first generations of RNN. In section 3, we present the data set used. Section 4 presents the computational aspects, while the LSTM approach applied to oil and cotton prices is highlighted in section 5. In the sixth section, we compare the LSTM approach with ARIMA and naïve models. In the seventh section, the forecast averaging approach and our conclusions are discussed in section 8.

2. Methodology

2.1. Artificial Neural Networks

An artificial Neural Network is a supervised learning technique within the machine learning set of models. It is used for both regression and classification tasks depending on the type of data used and the modeling purpose. An artificial neural network comprises an input layer that encounters the different input features, at least one hidden layer that will process the data set's hidden characteristics, and an output layer that will yield the network’s forecasts. Each layer has several units called “neurons,” which have the role of receiving information from preceding neurons and sending processed information to the following ones. The input layer has several neurons equal to the number of the dataset’s features, and each layer is augmented with a so-called “bias” neuron whose role is to facilitate algorithmic writing.

The number of hidden layers – which is one of the so-called hyperparameters - is set by the modeler through a fine-tuning process. The number of output layer neurons is equal to the number of expected outputs from the network. Within an artificial neural network, each neuron communicates with all the neurons from the preceding layer and with those from the following layer (aside from the last layer) through weights affected at each connection. The latter is computed by applying an activation function - which is usually a sigmoid or hyperbolic tangent function, among others that are used in the literature - to the product of a weights vector controlling the mapping function from a layer 𝑙𝑙 to the following layer 𝑙𝑙 + 1, and the preceding activations of the previous layer.

The learning process of an artificial neural network is performed using the backpropagation algorithm (Rumelhart, Hinton, & William, 1986), which is the optimization process through which the weights that represent the connections between neurons are updated using the chain rule method to compute the gradient of the loss function with respect to the weights. A gradient descent rule or related version is used to update their values. The weights are randomly initialized using a uniform distribution, the bias units are set to zero, and the initial activations for the first layer are equal to the dataset’s features. The backpropagation procedure is as follows: A forward pass is performed by computing the “cash” values used as an argument to assess the activation of layers. For an L −layers neural network, the 𝑘𝑘𝑡𝑡ℎ layer uses the following constitutive equations to yield the cash vector 𝑧𝑧[𝑘𝑘] and the activation vector 𝑎𝑎[𝑘𝑘] for each neuron and layer.

𝑧𝑧[𝑘𝑘] = 𝜂𝜂[𝑘𝑘] 𝑎𝑎[𝑘𝑘−1] + 𝑏𝑏[𝑘𝑘]

𝑎𝑎[𝑘𝑘] = 𝑔𝑔[𝑘𝑘] (𝑧𝑧[𝑘𝑘])

In equations (1) and (2), 𝜂𝜂[𝑘𝑘] is the vector of weights of all connections going from neurons at layer 𝑘𝑘 − 1 to neurons at layer 𝑘𝑘, and 𝑔𝑔[𝑘𝑘] is the activation function applied to neurons at layer 𝑘𝑘. The network’s last activation function corresponds to the network’s predicted value, which is compared with the actual .

Artificial neural networks are suitable for numerical and qualitative datasets for regression and classification tasks. However, for sequential-type datasets where inputs and outputs are sequences, ANNs are not adequate for two main reasons: (i) Inputs and outputs need to have the same lengths, and such could not be the case in sequential datasets, (ii) ANNs cannot deal with the need of sharing features learned across different positions in a sequential dataset. Recurrent Neural Networks (RNN) have been introduced to solve the limitations mentioned above

value from the training dataset’s response variable. A backward pass is then applied to the artificial neural network to compute the gradients’ cost function for weights assigned to each connection. The residual 𝛿𝛿[𝐿𝐿] = 𝑎𝑎[𝐿𝐿] – 𝑦𝑦, at the last layer between the network predictions 𝑎𝑎[𝐿𝐿] and actual values 𝑦𝑦 is backpropagated into the network, and the residual associated with the preceding layers is computed as follows:

𝛿𝛿[𝑘𝑘] = 𝜂𝜂[𝑘𝑘]

𝑇𝑇 𝛿𝛿[𝑘𝑘+1] ∘ 𝑔𝑔[𝑘𝑘]′ 𝑧𝑧[𝑘𝑘]