Autoencoder Neural Network Approaches for Anomaly Detection in IBOVESPA Stock Market Index

Lucas de Azevedo Takara*, Viviana Cocco Mariani*†, Leandro dos Santos Coelho*‡
*Department of Electrical Engineering (PPGEE), Federal University of Parana (UFPR),
Curitiba, Parana, Brazil. 81530–000.
†Mechanical Engineering Graduate Program (PPGEM), Pontifical Catholic University of Parana (PUCPR),
Curitiba, Parana, Brazil. 80215–901
‡Industrial and Systems Engineering Graduate Program (PPGEPS), Pontifical Catholic University of Parana (PUCPR),
Curitiba, Parana, Brazil. 80215–901
Email: *lucastakara2@gmail.com, *†viviana.mariani@pucpr.br, *‡leandro.coelho@pucpr.br

***Abstract*—Anomalies are patterns in data that do not conform to a well-defined notion of normal behavior. Anomaly detection has been applied to many problems such as bank fraud, fault detection, noise reduction, among many others. Some approaches to detect anomalies include classical statistical econometric methods such as AutoRegressive Moving Average (ARMA) and AutoRegressive Integrated Moving Average (ARIMA) approaches. More recently, with the progress of artificial intelligence and more specifically, machine learning, new algorithms such as one-class support vector machines, isolation forest, gradient boosting, and deep neural networks were applied to such tasks. This paper focuses on propose an anomaly detection framework for the *Índice da Bolsa de Valores de São Paulo* (IBOVESPA). It is a major stock market index that tracks the performance of around 50 most liquid stocks traded on the São Paulo Stock Exchange in Brazil. Exploring unsupervised autoencoder neural network algorithms, we compare the long short-term autoencoder, bidirectional long short-term autoencoder, and convolutional autoencoder models, aiming to explore the performance of these architectures for anomaly detection. Due to the ability of autoencoders to learn a compressed representation of their respective input, we train these models with standard data by minimizing the mean absolute error (MAE) loss function and evaluate them with anomalous inputs. We set a reconstruction error threshold, and in case that the reconstruction error of the test data sample is beyond it, anomalies are detected. Our results show that these models perform quite well and can be applied to real stock market data.**

***Index Terms*—Anomaly Detection, Stock Markets, Deep Learning, Autoencoders, Financial Signal Processing**

I. Introduction

Anomaly detection refers to the problem of finding patterns in data that do not conform to expected behavior [1]. It has been researched within diverse application domains such as in industrial machine fault detection, financial fraud, and noise reduction (e.g. [2]–[5]), because anomalies in data represent significant information. Over the years, many anomaly detection techniques have been developed in several research communities. Some approaches have been specifically developed for certain application domains, while others are general. The analysis of anomalies in time series data examines anomalous behaviors across time. Many approaches for this particular kind of data define a region representing normal behavior and declare any observation in the data that does not belong to this normal region as an anomaly.

The *Índice da Bolsa de Valores de São Paulo* (IBOVESPA) tracks the performance of around 50 most liquid stocks traded on the São Paulo Stock Exchange in Brazil. Its data contain a highly non-linear and non-parametric behavior as a result of a large amount of noise caused by exogeneous variables such as financial or political events, psychology of the investors, and other market influences [6]–[8], leading to a highly non-stationary time series. In financial data, anomalies are often not rare objects, but unexpected bursts in activity.

Aiming to detect these unexpected bursts in stock markets, many statistical and machine learning approaches were proposed. Traditional statistical methods focus on developing autoregressive models, such as AutoRegressive Integrated Moving Average (ARIMA) [9], AutoRegressive Conditional Heteroskedastic (ARCH), and generalized AutoRegressive Conditional Heteroskedasticity (GARCH) to detect anomalies [10]. ARCH is a method that explicitly models the change in variance over time in a time series. Its extension, GARCH, incorporates a moving average component together with the autoregressive component. Finally, an ARIMA model is similar to ARCH however, its "CH" component models the previously squared residuals at each previous point in time.

Machine Learning is an artificial intelligence subfield in which studies computer algorithms that improve automatically through experience and by the use of data. Classical machine learning methods, such as K-Means Clustering – Subsequence Time-Series Clustering, Isolation Forest, and Extreme Gradient Boosting are widely applied to univariate time series anomaly detection [11]–[13]. These algorithms have good performance mainly in structured linear data. To overcome this limitation, deep learning techniques have become popular over the decades and have been extensively used to successfully

address a wide range of traditional applications due to the ability to learn from nonlinear data. At its core, activation functions decide as to whether or not to fire a neuron concerning a particular input by creating the corresponding output and nonlinearity. Recurrent, convolutional, and autoencoder neural network architectures have been explored in many publications, proposing univariate time series anomaly detection [14]–[17].

This paper proposes to use unsupervised autoencoder neural networks to create a framework to detect anomalies in univariate time series data. We build, evaluate and compare 3 distinct autoencoder models. All of them are trained and tested in the IBOVESPA stock market index. The main contributions of this paper are:

- Propose and evaluate long short-term memory (LSTM), bidirectional long short-term memory (BiLSTM), and convolutional autoencoder neural networks to detect univariate time series anomalies.
- Provide insights about anomalies in the IBOVESPA stock market index and its correlations with exogeneous factors.

The remainder of this paper is organized as follows: In Section II, a brief overview related to autoencoder neural networks and the background of the proposed machine learning approaches are introduced. In Section III, the adopted methodology and proposed approaches are detailed. Then, the analysis of the results to anomaly detection are presented in Section IV. Finally, the paper is concluded in Section V.

II. Background

In this section, we briefly review some machine learning algorithms that are necessary to build the proposed algorithm for anomaly detection, including the LSTM and BiLSTM architectures, the autoencoder network and the convolutional neural network.

A. Long Short-Term Memory

Recurrent neural networks (RNNs) are a class of artificial neural networks that allow previous outputs to be used as inputs while having hidden states. This class of neural networks does not perform well on learning long-term dependencies due to the vanishing gradient problem, where the contribution of information decays geometrically over time. LSTM was introduced to solve this problem [18]. The long short-term memory is a special kind of RNN that allows a recurrent system to learn over as many time steps as possible without the loss of information. As shown in Figure 1, the LSTM architecture is composed of gates that regulate the information flow and turn long-term learning possible.

The forget gate f_t takes the last output of the LSTM unit h_{t-1}, the current input X_t and then combines them into a single tensor $[h_{t-1}, X_t]$. Then, it applies a linear transformation with W_f being the weights of the gate and b_f its respective bias, followed by a sigmoid σ activation layer. Once f_t is calculated, the forget gate decides what parts to keep

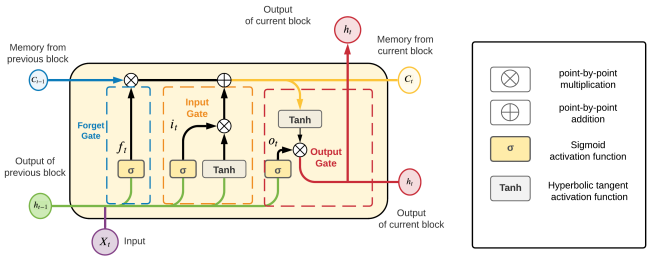


Fig. 1: Long Short-Term Memory

and to forget. The input gate i_t takes the $[h_{t-1}, X_t]$ tensor into a sigmoid function. A hyperbolic tangent $tanh$ activation function is also applied to $[h_{t-1}, X_t]$. Finally, the $tanh$ output is multiplied with the sigmoid output. The sigmoid output will decide which information is important to keep from the $tanh$ output. W_i represents the weights of the input gate and b_i its bias. The output gate o_t also takes $[h_{t-1}, X_t]$ tensor and applies it to a sigmoid activation layer as well as to a $tanh$. Then, it multiplies the $tanh$ output with the sigmoid output to decide what information the hidden state should carry. W_0 represents the weights of the input gate and b_0 its bias. The forget, input, and output gates are given respectively by:

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{1}$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{2}$$

$$o_t = \sigma(W_0[h_{t-1}, x_t] + b_0) \tag{3}$$

B. Autoencoder Network

An autoencoder is an unsupervised deep learning model that seeks to learn a compressed representation of its respective input [19]. As shown in Figure 2, its architecture consists of an encoder, code, and a decoder. The encoder component takes the input data and compresses it as a compressed representation in a reduced dimension. The input size of an encoder network is larger than its output size. The code contains the reduced representation of the input that is fed into the decoder. It has a smaller dimensional value than the input information. Finally, the decoder is responsible for reconstructing the input back to the original dimensions from the code. The autoencoder output is compared with the initial data and the error that measures the quality of the reconstruction, is backpropagated through the architecture to update the weights of the network.

There are several types of autoencoders such as convolutional autoencoder, denoising autoencoder and LSTM autoencoder. LSTM autoencoders refer to the autoencoder that both the encoder and the decoder are LSTM networks. These types of autoencoders are particularly suited for anomaly detection in time series due to LTSM's ability to learn temporal patterns in data over long sequences. Convolutional autoencoders can

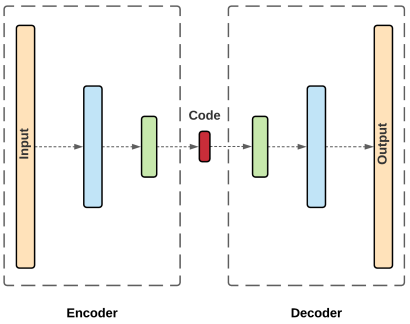


Fig. 2: Architecture of the autoencoder neural network

learn spatial patterns through their convolutional kernels ability. Transforming time series data in "textures", convolutional neural networks aim to limit the number of inputs while maintaining the strong "spatially local" correlation of the images.

C. Bidirectional Long Short-Term Memory

BiLSTM is an extension of the LSTM architecture [20]. This model overcomes the conventional LSTM by being able to make use of the previous context. It achieves this through processing the data in both directions with two separate hidden layers, which are then fed to the same output layer. This means that for every point in a given sequence, the BiLSTM has complete sequential information about all points before and after it. Figure 3 shows the BiLSTM architecture.

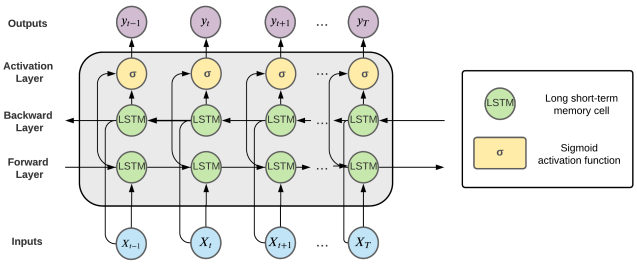


Fig. 3: Bidirectional LSTM architecture

D. Convolutional Neural Network

A convolutional neural network (CNN) is a type of neural network originally designed to process image data. Although there are plenty of applications to work with two-dimensional data, CNNs can be used to model univariate time series forecasting and anomaly detection problems (e.g. [15], [21]). At its core lies the convolutional layer, which applies the operation called "convolution". This linear operation represents the multiplication of a set of weights with the input. Given that the technique was designed for two-dimensional input, the multiplication is performed between an array of input data and a two-dimensional array of weights, called a filter or a kernel. As shown in Figure 4, the architecture of convolutional neural networks usually consists of convolutional layers, pooling layers, and fully connected layers. The purpose of the convolution layer is pattern recognition, in other words, it is to extract different features of the input, and more layers can iteratively extract more complex features from the last feature. Pooling layer aims to reduce dimensionality, Dense layers combine all local features into global ones and it is utilized to calculate the final result.

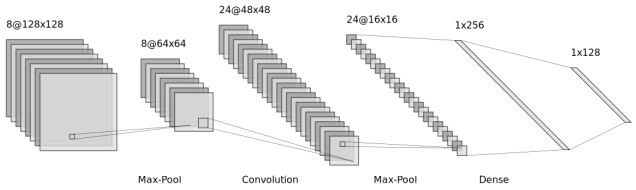


Fig. 4: Convolutional neural network with LeNet framework

III. Experimental Setup and Proposed Approaches

In this section, we discuss the applied methodologies to preprocess the IBOVESPA stock market index data and explain our approach to detect anomalies as well as our model architectures. The proposed models were built in Keras, which is an open-source software library that provides a Python interface for artificial neural networks.

A. Dataset

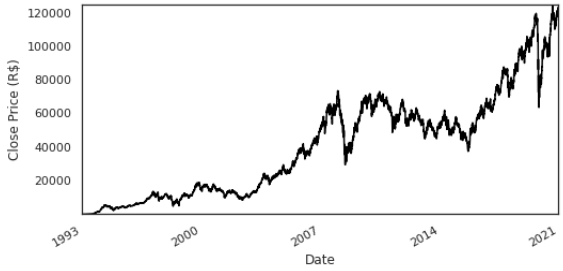
The IBOVESPA index dataset was obtained through the Yahoo Finance website. As shown in table 1, the original dataset contains the following features: date, open, high, low, close, adjusted close, and volume. The trade date is the day that an order is executed in the market. Open feature stands for the cash value of the 1st transacted stock when the market opens. The closing price is the raw price, which is just the cash value of the last transacted price before the market closes. The adjusted closing price amends a stock's closing price to reflect that stock's value after accounting for any corporate actions. The high and low stands for respectively the highest and lowest transacted price that the stock was daily traded. Lastly, volume is the number of shares of a stock traded during a given period.

TABLE I: IBOVESPA Dataset.

Date	Open	High	Low	Close	Adjusted Close	Volume
07/05/21	119922	122038	119922	122038	122038	8865100
10/05/21	122038	122772	121795	121909	121909	8219100
11/05/21	121904	122964	120145	122964	122964	7988600
12/05/21	122964	122964	119458	119710	119710	9405200
13/05/21	119711	121426	119711	120706	120706	9356200

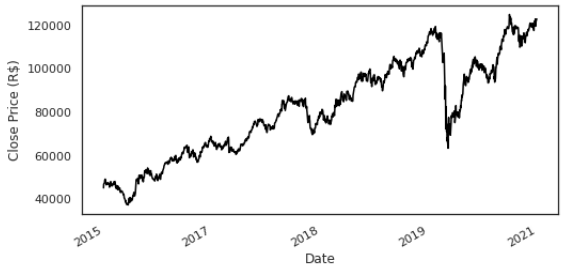
In this study, we only consider date and close price features to predict anomalies. The IBOVESPA closing prices feature contains the total amount of 6937 samples that were daily collected, during business days, between the periods of 1993 to 2021. Aiming to feed the data to deep learning models, the time series samples were sliced respectively into the percentage of 80% for training and 20% for testing. The

training dataset carries the interval between "1993-04-28 - 2015-09-30", whereas the test dataset holds data from "2015-09-30 - 2021-05-18". Figure 5 shows the original, train and test IBOVESPA close prices dataset.



(a) IBOVESPA index 1993 - 2021

(b) IBOVESPA index from 1993 - 2015



(c) IBOVESPA index 2015 - 2021

Fig. 5: IBOVESPA index preprocessed datasets

Before feeding the time series data to our models, the data is standardized so that the mean of observed values is 0 and the standard deviation is 1. In this paper, we adopt a sequential prediction approach, in other words, we define the LSTM autoencoder, BiLSTM autoencoder, and convolutional autoencoder architectures expecting an input sequence with 30-time steps, containing the information of close prices during 30 days.

B. Anomaly Detection Procedure

In this paper, we use unsupervised autoencoder deep learning models to detect anomalies, which are considered as close prices data that the autoencoder models are struggling to reconstruct. During the training phase, the period of "1993-04-28 - 2015-09-30" is adopted as input data to the encoder. The code layer will learn the behavior of this period, which represents 80% of the entire close price transactions of the IBOVESPA index. Then, daily close prices from the "2015-09-30 - 2021-05-18" period will be fed into the model. We hypothesize that if the close price data from the previous mentioned period behaves differently from learned representation, the autoencoder will have difficulties in reconstructing it, hence the model will result in a high reconstruction error value. Having this procedure in mind, we set a threshold for the reconstruction error to predict anomalies. Adopting the mean absolute error (MAE) as the loss function, the train and test loss distributions are generated and the reconstruction error threshold is set as the maximum train loss value. Finally, if a test loss error value is greater than the threshold adopted, then it is labeled as an anomaly. Figure 6 shows the flowchart of the aforementioned procedure.

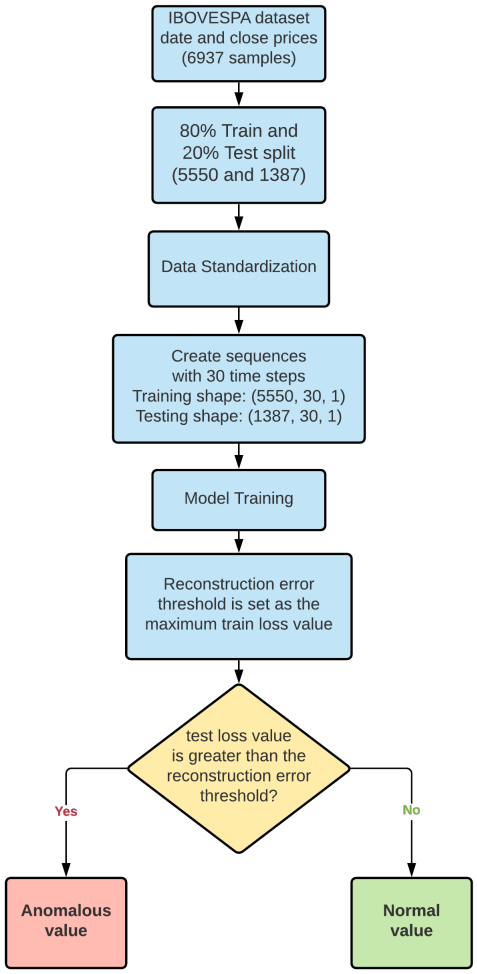


Fig. 6: Anomaly Detection methodology

C. Proposed Approaches

An LSTM autoencoder is an implementation of an autoencoder for sequential data using an encoder-decoder LSTM architecture. In this structure, an encoder LSTM model reads

the input sequentially. Once read the entire input sequence, the hidden state of this model stands for an internal learned representation of the entire input sequence as a fixed-length vector. The vector is then provided as an input to the decoder, which interprets it and generates an output that maps back from the internal learned representation to the input space. We define an LSTM autoencoder architecture containing a total of 4,385 trainable parameters. It expects input sequences with 30-time steps, in other words, 30 days and one feature, which is closing prices, outputting a sequence with the same dimensions. The LSTM autoencoder contains in its structure a Repeat Vector layer, which acts like a bridge between the encoder and decoder modules, duplicating the LSTM cell output 30 times. Moreover, it also contains Time Distributed and Dense layers to generate the output. The Time Distributed layer creates a vector of length equal to the number of features outputted from the previous layer, in this case, the previous layer outputs 128 features, therefore, the Time Distributed layer creates a 128 long vector and multiplicates by 1, which is the number of features $(128; 1)$. The Dense layer applies a dot product between the output of the Repeat Vector layer $(30; 128)$ and the output of the Time Distributed layer $(128; 1)$, resulting in the same dimensional shape as the input of the network $(30; 1)$. The objective of fitting the network is to make this output close to the input.

A BiLSTM autoencoder is an implementation of an autoencoder for sequence data using an encoder-decoder bidirectional LSTM architecture. Similarly, as in LSTM autoencoder, the input expects sequences with 30-time steps and one feature and outputs a sequence same dimensions The model has a total of 133,377 trainable parameters, containing the same structure of the LSTM autoencoder, however, is replaced by BiLSTM cells.

Lastly, the convolutional reconstruction autoencoder model takes the input of shape with 30-time steps and one feature, returning an output of the same shape. Containing a total of 137 trainable parameters, its structure accommodates a convolutional layer holding 8 filters and a kernel size of 1, a max-pooling layer of pool size of 2, and a flatten layer. The Repeat Vector layer duplicates the inputs 30 times. Finally, Time Distributed and Dense layers are added at the end to shape the final output. Due to its reduced number of parameters compared to LSTM based autoencoders, convolutional autoencoders consume less computational power than the aforementioned models and takes less time to train.

D. Hyperparameters Tuning

Before training the adopted neural network architectures, the number of hidden units, epochs, batch size, and learning rate hyperparameters must be tuned. Hyperparameters are variables that determine how the network structure is built and trained. To tune the number of hidden units and epochs of each model, the grid search technique was adopted. It performs an exhaustive search between different values, attempting to compute the optimum number that minimizes the train and test loss values.

The grid search for the optimal hidden units value was performed between 5 different numbers in both LSTM and BiLSTM based models: 16, 32, 64, 128 and 256. Evaluated with the lowest mean test loss value of 0.1875, the amount of 32 hidden units was elected in the LSTM. Unlike the previous model, the selected number of 128 hidden units had the lowest test loss value of 0.1744. The grid search for the optimal number of hidden units of the convolutional autoencoder neural network was searched between 4, 8, 16, 32 and 64 values. The selected number of 8 hidden unit value had the lowest mean test loss value of 0.1896.

Once selected the number of hidden units, the grid search algorithm is performed to select the optimal number of epochs to prevent overfitting. The LSTM based autoencoder search had 5 different values: 15, 30, 45, 60, and 100. 60 epochs were chosen as the optimal value, due to its lowest test loss value. For greater values, the model starts to overfit. Containing the greatest number of parameters compared to the remaining models, the optimal value of 100 epochs was selected to the BiLSTM. Lastly, the number of 45 epochs was selected in the convolutional autoencoder neural network. The batch size number was chosen as the 32 default number for all models. The Adam optimization algorithm was the optimizer selected, containing a learning rate of 0.001.

IV. Results Analysis

To assess the performance of the autoencoder models for anomaly detection, the mean absolute error is adopted as the loss function, which is given by:

$$MAE = \frac{1}{N}\sum_{x=1}^{N}|y_i - x_i| \tag{4}$$

where N is the number of samples, y_i is the predicted value and x_i is the true value. Table II and III contain the mean, maximum and minimum train and test loss values in respect to each model's performance, as well as the adopted threshold for anomaly detection.

TABLE II: Autoencoder Models Performance in Train Dataset

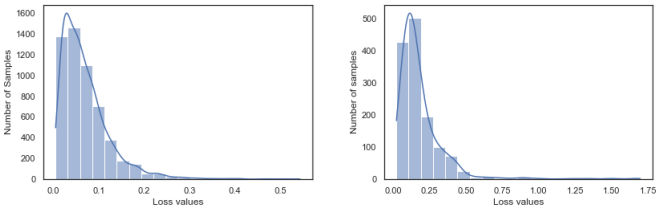
Train loss distribution values	LSTM Autoencoder	BiLSTM Autoencoder	Convolutional Autoencoder
Minimum	0.0052	0.0119	0.0203
Maximum (Threshold)	0.5438	0.5424	0.3972
Mean	0.0737	0.0779	0.0984

A. LSTM Autoencoder

Figure 7 shows the LSTM autoencoder train and test loss histograms along with the respective kernel density estimate (KDE) plots. Achieving a mean error value of 0.1875 in the test loss, this model contains respectively a 1.12% lower and 7.51% greater mean loss value than the convolutional and BiLSTM autoencoders, scoring the 2nd best performance

TABLE III: Autoencoder Models Performance in Test Dataset

Test loss distribution values	LSTM Autoencoder	BiLSTM Autoencoder	Convolutional Autoencoder
Minimum	0.0238	0.0225	0.0271
Maximum	1.6959	1.6558	1.6653
Mean	0.1875	0.1744	0.1896



(a) Train Loss (b) Test Loss

Fig. 7: LSTM autoencoder train and test loss histograms along with KDE plots in IBOVESPA index.

between the chosen neural networks. This distribution contains the maximum error of 1.6959, resulting in a 211.86% greater number compared to the achieved 0.5438 maximum train loss value. This fact indicates that there are anomalous points and the LSTM autoencoder is having difficulties in reconstructing it. The train loss distribution is 2.2 right-skewed. Skewness refers to an asymmetry that deviates from the normal distribution. It also contains a 8.4 kurtosis, which determines the heaviness of the distribution tails. As previously mentioned, the maximum train loss error value generated in the LSTM autoencoder is 0.5438, therefore, this is the selected threshold number. The test loss distribution is also right-skewed. Containing a 4.2 positive value, it is 90.91% more asymmetrical than the train loss. Moreover, it contains a 25.8 kurtosis, indicating heaviness of the distribution tail, as expected. Figure 8 shows the test loss distribution from 2015 to 2021, indicating the 0.5438 selected threshold in IBOVESPA index.

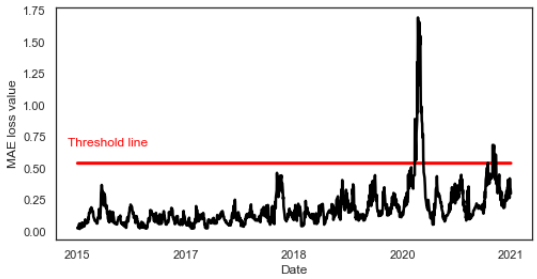


Fig. 8: LSTM autoencoder test loss distribution in IBOVESPA index during 2015 - 2021, containing the selected threshold.

Ranging from 0.0 to approximately 1.7, there's a clearly indication of a loss peak in the year of 2020 and some anomalous error values in 2021 as well. Figure 9 shows the total of 36 anomalous daily close prices detected by the LSTM autoencoder in the IBOVESPA index dataset. Through its ability to learn temporal patterns, this model indicates anomalous close prices in a 2020 dump from R$ 100,000 to approximately R$ 60,000. It also indicates an anomalous behavior in a 2021 dump from approximately R$ 115,000 to R$ 112,500.

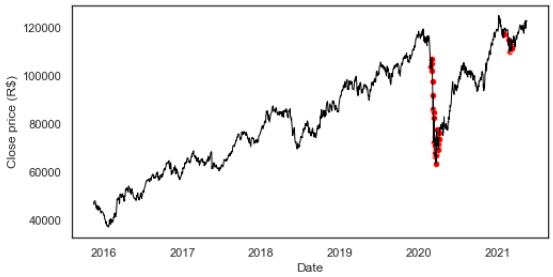
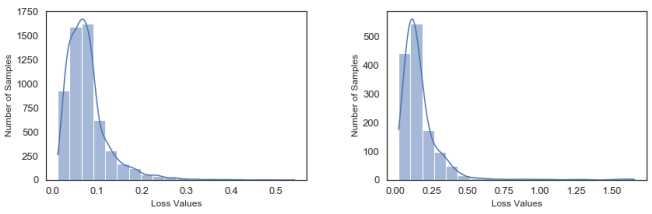


Fig. 9: Detected anomalous daily close prices in LSTM autoencoder from 2015 to 2021 in IBOVESPA index.

B. *Bidirectional LSTM Autoencoder*

Figure 10 shows the BiLSTM autoencoder train and test loss histograms along with the respective KDE plots. In regard to the test loss, this model achieved the mean value of 0.1744. In consequence, it is elected as the optimal autoencoder for detecting anomalies between the proposed approaches, due to its 7.51% and 8.72% lower test mean loss compared to the LSTM autoencoder and convolutional autoencoder. In addition to this fact, the same distribution achieved a maximum and minimum error of 1.6558 and 0.0225 respectively.

Evaluated with maximum, minimum and mean values of respectively 0.5424, 0.0119, and 0.0779, the train loss has a 2.5 positive skewness and a 11.4 kurtosis. Therefore, it is 13.64% more asymmetrical and 35.71% heavier tailed than LSTM's autoencoder train loss distribution. In respect to the test loss, this model has a 4.8 positive skewness value and a 32.3 kurtosis, hence it is 14.29% more asymmetrical and 25.19% heavier tailed than LSTM's autoencoder test loss. Lastly, the selected threshold in this model is 0.5424. This is 0.26% lower compared to the previous autoencoder approach.



(a) Train Loss (b) Test Loss

Fig. 10: BiLSTM autoencoder train and test loss histograms along with KDE plots in IBOVESPA index.

Figure 11 shows the test loss distribution during 2015 - 2021, indicating the 0.5424 selected threshold. Ranging from 0.0 to approximately 1.656, there is a clearly indication of a loss peak in the year of 2020 and some anomalous behavior in 2021. Figure 12 shows the total of 32 anomalous daily close prices detected by the BiLSTM autoencoder in the IBOVESPA index dataset.

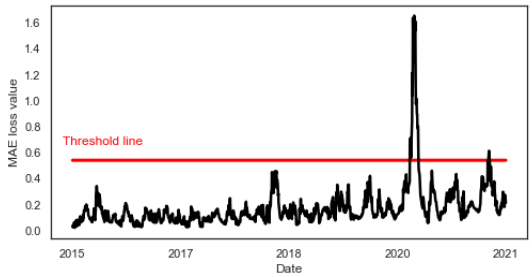


Fig. 11: BiLSTM autoencoder test loss distribution in IBOVESPA index during 2015 - 2021, containing the selected threshold

Through BiLSTM's ability to learn temporal patterns in both directions, this model possesses a more robust approach, indicating 4 fewer anomalous close prices compared to the previous model. BiLSTM based autoencoder indicates anomalies in a 2020 dump from R$ 100,000 to approximately R$ 60,000. However, unlike the LSTM autoencoder, this model presented one anomalous close price in a dump in the year of 2021, however since it is a single mark, it may be considered as a noisy event.

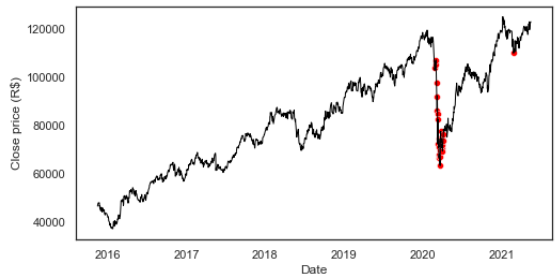


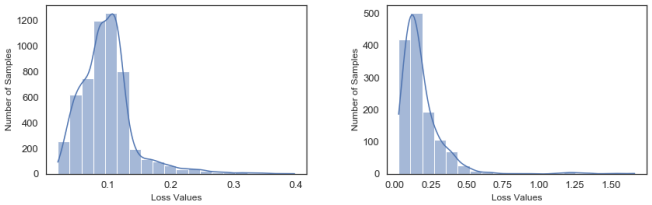
Fig. 12: Detected anomalous daily close prices in BiLSTM autoencoder from 2015 to 2021 in IBOVESPA index.

C. Convolutional Autoencoder

Figure 13 shows the convolutional autoencoder train and test loss histograms along with the respective KDE plots. Achieving the mean test loss value of 0.1896, this model had the worst performance in comparison to the proposed approaches, due to its 1.12% and 8.72% greater test loss value compared to the LSTM autoencoder and BiLSTM autoencoder. Moreover, the same distribution achieved a maximum and a minimum error of 1.6653 and 0.0271.

Evaluated with maximum, minimum and mean values of respectively 0.3972, 0.0203, and 0.0984, the train loss distribution is 1.8 right-skewed and contains a 6.4 kurtosis, hence it is respectively 19.75% and 39.35% more symmetrical than the LSTM and BiLSTM autoencoders' distribution. Moreover, it is respectively 29.96% and 77.72% more light-tailed than the aforementioned models train loss. The selected threshold value is 0.3972, hence it is 36.91% and 36.54% lower than the LSTM and BiLSTM selected values.

The test loss is 4.2 positively skewed and contains a 25.3 kurtosis. This means that this distribution has approximately the same asymmetry, from a normal distribution, compared to the LSTM autoencoder and it is 14.29% more symmetrical than BiLSTM's autoencoder test loss. In addition to this fact, the same distribution is respectively 1.98% and 27.84% more light-tailed than the previous mentioned models' loss distributions.



(a) Train Loss (b) Test Loss

Fig. 13: Convolutional autoencoder train and test loss histograms along with KDE plots in IBOVESPA index.

Figure 14 shows the test loss of the convolutional autoencoder from 2015 to 2021. Ranging from 0.0 to 1.75 as well, this model presents 6 loss peak values intersecting the threshold line. Anomalies are detected during the years of 2018, 2020, and 2021. Through the use of convolutional layers, hence being able to learn spatial patterns, this model possesses a greater volatility sensibility, which is the reason for the lowest threshold value selected compared to the other proposed approaches.

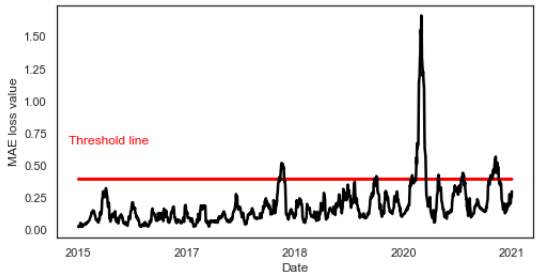


Fig. 14: Convolutional autoencoder test loss distribution in IBOVESPA index during 2015 - 2021, containing the selected threshold.

Figure 15 shows the total of 91 anomalous daily close prices detected by the convolutional autoencoder in the IBOVESPA

index dataset during 2015 - 2021. This model indicates anomalous close prices in a 2018 dump from R$ 70,000 to R$ 65,000, a 2020 dump from R$ 100,000 to approximately R$ 60,000, and finally, a 2021 dump from approximately R$ 115,000 to R$ 112,500. Some noisy points in the middle of 2019 and in 2020 are also detected. The convolutional autoencoder contains 55 more anomalous close prices compared to the LSTM autoencoder and 59 compared to the BiLSTM autoencoder.

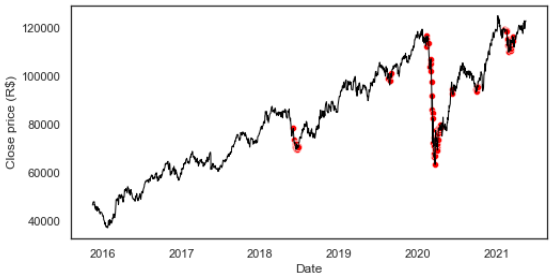


Fig. 15: Detected anomalous daily close prices in convolutional autoencoder from 2015 to 2021 in IBOVESPA index.

V. Conclusion and Future Research

In this work, we presented an anomaly detection framework based on unsupervised autoencoder neural networks. The obtained results have shown that these methods achieve a good performance on the IBOVESPA stock market index dataset. The collection of anomalous close price points detected are highly correlated with the start of the COVID-19 pandemic in Brazil, during March 2020. In March 2021, another cluster of anomalous points is correlated with political events. These anomalies emphasize that stock markets are highly influenced by exogenous factors.

The LSTM, BiLSTM, and convolutional autoencoder performances were evaluated showing respectively a test loss mean value of 0.1875, 0.1744, and 0.1896. LSTM and BiLSTM autoencoders tend to learn sequences by exploring temporal features, whereas the convolutional autoencoder tends to learn spatial patterns. The threshold value of these models was selected by choosing the maximum train loss value of the respective model. The LSTM and BiLSTM thresholds were rated respectively at 0.5438 and 0.5424, representing a more robust approach, whereas the convolutional threshold value was 0.3972, indicating a higher volatility sensibility. These models can be adjusted more precisely by fine-tuning the threshold value. BiLSTM autoencoder was elected the optimal neural network autoencoder to detect anomalies. This model had respectively 7.51% and 8.72% lower mean test loss value than the LSTM and convolutional autoencoders.

A future investigation should include hybrid approaches such as LSTM-CNN autoencoder models in which explore both temporal and spatial data patterns. In addition to this, optimization algorithms could be explored such as Bayesian optimization, aiming to check its performance in hyperparameters tuning.

References

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, July 2009.

[2] V. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artificial Intelligence Review*, vol. 22, pp. 85–126, Oct 2004.

[3] E. Ngai, Y. Hu, Y. Wong, Y. Chen, and X. Sun, "The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature," *Decision Support Systems*, vol. 50, no. 3, pp. 559–569, 2011.

[4] P. Park, P. D. Marco, H. Shin, and J. Bang, "Fault detection and diagnosis using combined autoencoder and long short-term memory network," *Sensors*, vol. 19, no. 21, 2019.

[5] E. Dasan and I. Panneerselvam, "A novel dimensionality reduction approach for ecg signal via convolutional denoising autoencoder with lstm," *Biomedical Signal Processing and Control*, vol. 63, no. 102225, 2021.

[6] K. J. Forbes and R. Rigobon, "No contagion, only interdependence: Measuring stock market comovements," *The Journal of Finance*, vol. 57, no. 5, pp. 2223–2261, 2002.

[7] W. F. M. De Bondt and R. Thaler, "Does the stock market overreact?," *The Journal of Finance*, vol. 40, no. 3, pp. 793–805, 1985.

[8] X. Li, H. Xie, L. Chen, J. Wang, and X. Deng, "News impact on stock price return via sentiment analysis," *Knowledge-Based Systems*, vol. 69, pp. 14–23, 2014.

[9] D. Agnieszka and L. Magdalena, "Detection of outliers in the financial time series using arima models," in *Applications of Electromagnetics in Modern Techniques and Medicine (PTZE)*, pp. 49–52, 2018.

[10] P. H. Franses and D. van Dijk, *GARCH, Outliers, and Forecasting Volatility*, pp. 136–159. London, UK: Palgrave Macmillan UK, 2011.

[11] T. Idé, "Why does subsequence time-series clustering produce sine waves?," in *Knowledge Discovery in Databases: PKDD 2006* (J. Fürnkranz, T. Scheffer, and M. Spiliopoulou, eds.), (Berlin, Germany), pp. 211–222, Springer Berlin, 2006.

[12] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM TTransactions on Knowledge Discovery from Data*, vol. 6, Mar. 2012.

[13] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, (New York, NY, USA), p. 785–794, Association for Computing Machinery, 2016.

[14] W. Yang, R. Wang, and B. Wang, "Detection of anomaly stock price based on time series deep learning models," in *Management Science Informatization and Economic Innovation Development Conference (MSIEID)*, (Guangzhou, China), pp. 110–114, 2020.

[15] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "Deepant: A deep learning approach for unsupervised anomaly detection in time series," *IEEE Access*, vol. 7, pp. 1991–2005, 2019.

[16] H. Homayouni, S. Ghosh, I. Ray, S. Gondalia, J. Duggan, and M. G. Kahn, "An autocorrelation-based lstm-autoencoder for anomaly detection on time-series data," in *2020 IEEE International Conference on Big Data (Big Data)*, pp. 5068–5077, 2020.

[17] F. Caliva, F. S. De Ribeiro, A. Mylonakis, C. Demaziere, P. Vinai, G. Leontidis, and S. Kollias, "A deep learning approach to anomaly detection in nuclear reactors," in *International Joint Conference on Neural Networks (IJCNN)*, (Rio de Janeiro, Brazil), pp. 1–8, IEEE, 2018.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017.

[20] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[21] A. Essien and C. Giannetti, "A deep learning framework for univariate time series prediction using convolutional lstm stacked autoencoders," in *IEEE International Symposium on Innovations in Intelligent SysTems and Applications (INISTA)*, (Sofi, Bulgaria), pp. 1–6, 2019.