Stock Trading Classifier with Multichannel Convolutional Neural Network

Davi G. Nascimento[1], **Anna H. R. Costa**[2], **Reinaldo A. C. Bianchi**[1]

[1]Electrical Engineering – FEI University Center
Humberto de Alencar Castelo Branco, 3972 – São Bernardo do Campo – SP – Brasil

davign,rbianchi@fei.edu.br

[2]Escola Politécnica – Universidade de São Paulo
Prof. Luciano Gualberto, 158 – Butantã – SP – Brasil

anna.reali@usp.br

Abstract. *Stock market forecasting has been a quite popular challenge in machine learning research. Recently, studies have been using deep learning techniques, such as Convolutional Neural Networks (CNN), to perform regressions in prices or classification in the trading signal as an operation indication. However, these works have not yet achieved a satisfactory financial result. In this paper, we propose a financially profitable stock market method that uses a new approach called Multichannel CNN Trading Classifier (MCNN-TC). The model was evaluated using data from the Brazilian stock market. The results indicate a promising financial trading performance compared to the Buy and Hold strategy and good classification metrics.*

1. Introduction

Stock market prediction has been a popular challenge for the Machine Learning (ML) community for a long time. Recently, researchers in the Deep learning (DL) community started applying DL techniques to tackle this problem. The number of traders in stock markets has been growing in recent years, and as a result, intelligent decision systems are getting attention in financial markets globally.

Prices in the stock markets are naturally highly volatile, and choosing when to buy or sell is a complex task that depends on many variables, which is why the Buy and Hold (BH) strategy is the preferred choice of many investors. BH consists of a strategy of buying and holding stock over time, considering that its price will increase in the long run.

With the motivation of helping traders to forecast future trend in the stock markets, studies in technical analysis have improved over the past few years. Candlestick chart patterns and several technical indicators have been created using past market data in order to find the best operation entry point. Among the most popular we can cite the trend trackers, moving average and financial volume [Colby and Meyers 1988].

In the literature, there are some approaches using different techniques in order to forecast future market trend or regression on the prices, such as Support Vector Machines (SVM) [Cavalcante et al. 2016], Recurrent Neural Network (RNN) [Krauss et al. 2017], Long Short Term Memory (LSTM) [Fischer and Krauss 2018] and Convolutional Neural

Networks (CNN) [Sezer and Ozbayoglu 2018]. Among these models CNN has been the most commonly adapted DL model, the most of the use of CNN has been addressed for computer vision and image analysis challenges.

Motivated by these success, in this paper, we propose a novel approach to classify stock trading signal in the financial market. Our approach is titled Multichannel CNN Trading Classifier (MCNN-TC). The key idea is to scan the time series of each input feature using the power of Deep Convolutional Neural Network in order to find significant trading patterns that indicate the most profitable trading signal for each day. We evaluate the financial trading performance of our proposal, comparing it with the BH strategy and with the CNN-TA model [Sezer and Ozbayoglu 2018].

The remainder of this paper is structured as follows: after this introduction, we present the related work in Section 2. Section 3 presents background theory and our approach and experiments are described in Sections 4 and 5. Statistical and financial evaluation are analyzed in Section 6. Finally, discussions and conclusions are given in Section 7.

2. Related work

Various approaches have been developed in financial forecasting using traditional and deep machine learning models. Some approaches were implemented based on forecasting on time series financial data, whereas others based on using technical indicators and financial news to improve forecasting performance.

Some preferred using stock market sentiment analysis based on financial news, such as in [Vargas et al. 2017] that used CNN and LSTM for a Natural Language Processing approach. Their results showed that CNN can be better than LSTM catching semantic from texts, meanwhile LSTM is better at catching the context information. In [Mittal and Goel 2012] the authors proposed a Fuzzy Neural Network model using Twitter data to predict DJIA market movement, which obtained accuracy approximately of 75%.

Other research was based on time series analysis, such as [Bao et al. 2017] that proposed a DL model to forecast the next day closing price using Wavelet transforms and auto-encoders combined with a LSTM architecture. The results provide evidences that their model is able to outperform the Buy and Hold method and approaches using LSTM trading system. In [Selvin et al. 2017] the authors based their work in LSTM and CNN models in order to forecast future NSE stock prices. The experiments showed that CNN architecture is capable of identifying the changes in trends and it was the best model in the comparison.

In a similar approach, Jain et al. compared three different models (CNN, LSTM and CNN-LSTM) in future prices forecasting task [Jain et al. 2018]. All models were based on 1-D convolution and the input data used were composed of open, high, low and close daily prices of 5 previous trading days sequentially. The results showed that DL models can be employed in forecasting future prices of stock data and a combination of CNN-LSTM is able to give more accurate results when compared to each single architecture. Hoseinzade and Haratizadeh [Hoseinzade and Haratizadeh 2019] suggested two CNN approaches for predicting one day future trend for stock market based on technical indicators and macroeconomic variables. Their results indicate that CNN model is able to indicate future trend as a classification model.

Tsai et al. [Tsai et al. 2019] proposed the use of CNN combined with Gramian Angular Field, a time series to image representation technique, for chart patterns classification. These patterns are commonly used to find turning points in stock price and may indicate a buy/sell operation. The results showed an accuracy of approximately of 80%. Sezer et al. [Sezer and Ozbayoglu 2018] developed a financial time series trading model using a CNN image approach to classify the best buy and sell points. The model was called CNN-TA and it was based on using 15 technical indicators in 15 subsequent periods to generate 2-D images with shape of $[15 \times 15]$ that represents each daily stock sample. Each image was labeled as Buy, Sell or Hold according to a labeling algorithm based on sliding window. The results were compared to the Buy and Hold strategy and other trading systems over a long period, and indicate that CNN-TA was able to outperform all commonly compared trading strategies.

When analysing financial data, the most used models in literature are LSTM and CNN. CNN is most applied for image classification problems, meanwhile LSTM has been the preferred choice for time series forecasting. An accurate trading algorithm depends on many variables and a 2-D CNN model is able to represent a set of technical indicators in time series in order to classify trading patterns. Based on the aforementioned approaches, in this paper we combine a set of technical indicators with time series analysis using a Multichannel CNN model for trading signal classification. The arrangement of each input feature will be divided into different image channels overlapped. We compared our model to CNN-TA [Sezer and Ozbayoglu 2018], which is very similar to ours.

3. Background

Two types of methods are traditionally used to make decisions in stock markets: fundamental analysis and technical analysis. The first one is interested in the business related to the stock itself, requiring domain expertise and interpretation of financial data, such as balance sheets and economic factors. BH strategy is based on analyzing these data. It depends on data interpretation, Natural Language Processing techniques are more applicable.

On the other hand, technical analysis is interested in the variation of the stock price. According to the efficient market hypothesis, the price of stocks reflects all the information about them [Malkiel and Fama 1970] and technical analysis believes that prediction of future market behavior is possible by analyzing the previous price data.

Over the years, several technical indicators have been developed aiming to indicate trading signal using previous data, such as financial volume and past prices. Among the most frequently used are the trend trackers, moving averages, oscillators and momentum indicators. Mostly, they calculate a number using the required data which will be interpreted as a buy, sell or hold indication.

Some approaches that motivated this work make use of a set of technical indicators and a DL model to recognize time series patterns and classify the best trading operation at that moment.

LeCun et. all introduced Convolutional Neural Networks in 1990 in a work applied to handwritten recognition dataset [LeCun et al. 1990]. A typical CNN is represented in Figure 1. It is composed of convolutional layers, activation functions, pooling layers and fully-connected layers.

The convolutional layer is applied to find features in the input data, such as border, shape or color in an image. This layer replaces the typical matrix multiplication of Feedforward Neural Networks by a convolution operation. It provides sparse connectivity, parameter sharing, equivariant representation, reduces the memory usage and requires fewer operations [Goodfellow et al. 2016]. Convolution is performed using a matrix, known as kernel, that is multiplied, stride by stride, by the input matrix resulting in a feature map. In addition, an activation function is used at the end of the convolution process which gives additional functionality normalization. The most commonly are the Rectified Linear Unit (ReLU), Sigmoid, TanH, and Softmax.

Pooling layer is used to achieve invariance to image transformations and more compact representations by reducing the matrix dimension keeping the image features [Boureau et al. 2010, Goodfellow et al. 2016]. The method mostly commonly used in pooling operation is the max pooling, which picks the maximum value in the area of kernel.

Dropout is a technique for addressing the problem of overfitting [Srivastava et al. 2014]. The main idea is to randomly drop units (along with their connections) from the neural network during training in order to help the learning algorithm to converge to a generalized model [Hinton et al. 2012].

The final layers of a CNN are composed of feedforward fully connected network layers, that are used to perform the classification.

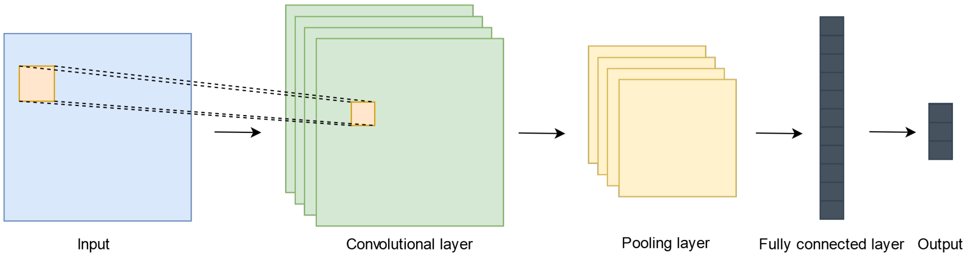

Figura 1. CNN process representation

4. Proposed model

The process of the proposed automated trading system is presented in this section. In our model, we propose a novel approach applying a CNN architecture to determine the best buy and sell points in order to achieve a better financial result than the BH strategy.

4.1. Input and data pre-processing

The input data is composed of the historical open, high, low and closing prices. We use Technical Analysis Library in Python[1] to compute the technical indicators values.

[1] `https://technical-analysis-library-in-python.readthedocs.io`, accessed on March, 2020.

The feature sets were empirically evaluated. We chose that set that provided the best accuracy in the learning process. The selected input features are as follows:

- **Stationary Closing Price (SCP)**: to be able to capture time series patterns invariant to the effect of time, we use the stationary price. A stationary series is easier to predict: you simply predict that its statistical properties in the future will be the same as they were in the past. Thus, each daily closing price is adjusted using the first difference method. We use the last twelve days for computation. The value of following day is subtracted and we apply Hyperbolic Tangent Function (Tanh) for normalization. As a result, a closing price mean around zero-point is obtained, similar to oscillator indicators;
- **Relative Strength Index (RSI)**: a momentum indicator that measures the magnitude of recent price changes to evaluate overbought or oversold conditions;
- **Williams %R (WR)**: reflects the closing price relative to the highest value for the look-back period;
- **Money Flow index (MFI)**: uses price and volume to measure buying and selling pressure;
- **Moving Average Convergence Divergence (MACD)**: a trend-following momentum indicator that shows the relationship between two moving averages of prices.

We then divide the dataset in samples using sliding window of size 11 and 1 as stride. The values of each sample are normalized between 0 and 1 for each feature using min-max normalization. Each matrix line (y-axis) denotes time, and each column (x-axis) represents a feature. The original matrix of each sample has the shape of $[11 \times 5]$. After that, the samples are reshaped into 3D shape of $[1 \times 5 \times 11]$ as shown in Figure 2. Each day will be viewed as a different image channel and the feature map will be a summarized result of convolution process.

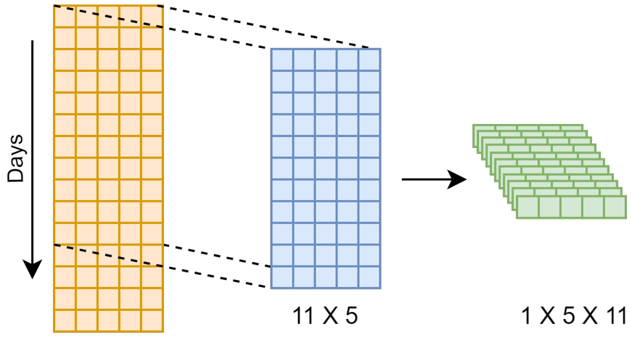

Figura 2. Input samples reshaping process

4.2. Labelling

A supervised training process is used, so all daily samples are labeled as Buy, Sell or Hold according to the Algorithm 1 using the closing prices. We propose this labelling method in order to find troughs and crests in the time series. Differently from the proposal of [Sezer and Ozbayoglu 2018], in this method we aim to identify the minimum and maximum values inside the rolling window over the time series as buy or sell operation point, providing more accurate turning points.

The labeling method divides the time serie into smaller segments. It begins at the first day, and moves forward using a sliding window with stride 1. For each segment, the subsequent days after the minimum and maximum values in current segment are marked as buy and sell, respectively, and the other days are marked as hold. This process repeats until the end of the time series. As a result, the turning points in price trend are marked as a trading entry point.

Algorithm 1 Labelling method

```
procedure LABELLING(prices, windowSize, totalDays)
    winBegin ← 0
    winEnd ← winBegin + windowSize
    countRow ← 0
    while countRow ≤ totalDays do
        minValue ← Min(prices[winBegin:winEnd])
        maxValue ← Max(prices[winBegin:winEnd])
        for i in prices[winBegin:winEnd] do
            if $prices[i] = minValue$ AND $prices[i]$ NOT NULL then
                labels[i] ← "Hold"
                labels[i + 1] ← "Buy"
            else if $prices[i] = maxValue$ AND $prices[i]$ NOT NULL then
                labels[i] ← "Hold"
                labels[i + 1] ← "Sell"
            else if $prices[i]$ NOT NULL then
                labels[i] ← "Hold"
        winBegin ← winEnd + 1
        winEnd ← winBegin + windowSize
        countRow ← winEnd
```

4.3. CNN architecture

The model architecture is presented in Figure 3. It is structured as follows: the input is a tensor of shape $[1 \times 5 \times 11]$ as described before, two convolutional layers ($[32 \times 1 \times 2]$ and $[64 \times 1 \times 3]$), two max pooling layers (pool size of $[1 \times 2]$), a fully connected layer (128 units) and an output layer. ReLu activation function is used in all layers, and the output layer uses Softmax function.

Convolutional layers capture changes over time for each input feature, max-pooling layer extracts larger features and perform dimensionality reduction. The fully connected layer makes the classification for a trading signal.

Differently to the CNN-TA model, this layer arrangement was defined in order to have a summarized representation of feature maps invariant to the order of features. In the first convolutional layer, the kernel is applied for each channel, represented by the third dimension of the tensor. The result of each channel is added and normalized using the ReLu function. As a result, the time dimension decreases from 11 to 1.

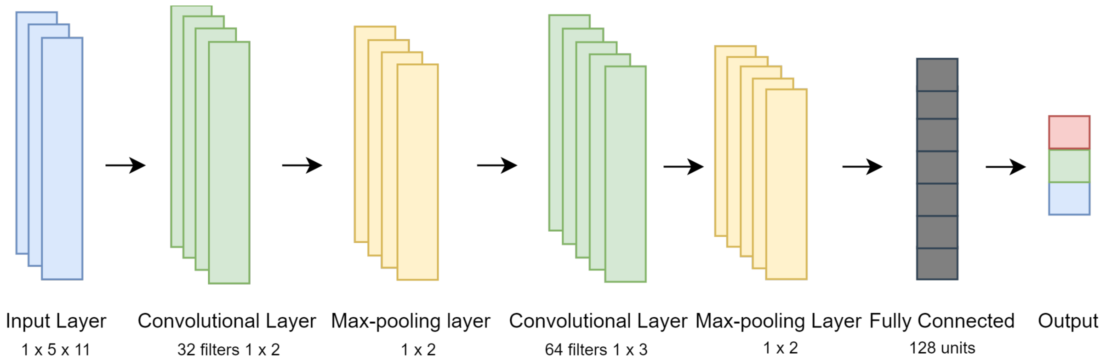

Figura 3. Proposed CNN architecture

5. Experiments

In the experiments, we evaluated our model in Brazilian stock market. We used daily prices of Brazilian Exchange-Traded Fund (BOVA11) and the top 3 most relevant Ibovespa's stock, Itaú (ITUB4), Vale (VALE3) and Petrobras (PETR4). The datasets were obtained from Yahoo Finance[2] in periods from 12-02-2008 to 12-31-2018. Figure 4 shows the generated points from labelling process in BOVA11 dataset as an example.

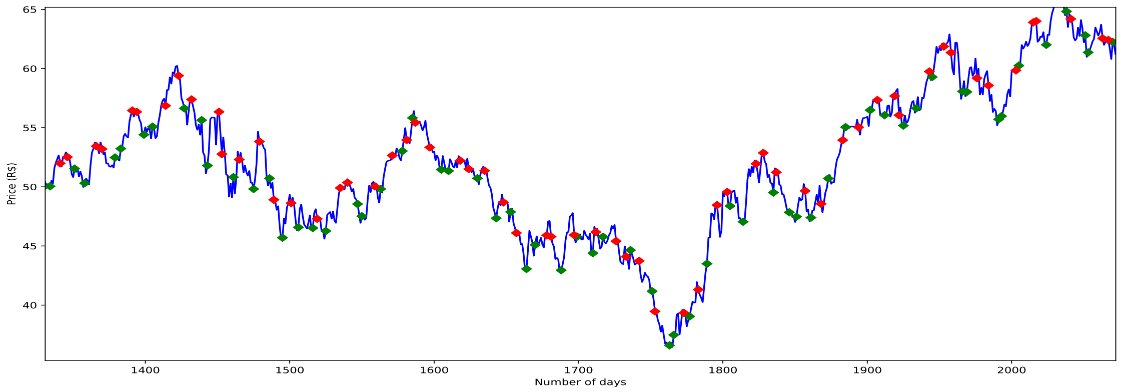

Figura 4. Buy sign with green dots, sell with red dots and hold are not marked.

We implemented the MCNN-TC using Keras[3]. Adam Optimization Algorithm [Kingma and Ba 2014] was used, with 1^{-4} initial learning rate and beta of 0.9. In order to improve optimization and prevent overfitting we used early stopping and model checkpoint monitoring the validation loss. Dropout of 0.5 was used before the fully connected layers. Moreover, due to the unbalanced classes, we used samples-weight, setting the Keras to pay less attention to hold class.

The training sessions were performed in sliding window with retraining approach according to Figure 5. Six years from the dataset were taken and divided 20% for validation crossover. The following year out of sample data was used for computational and financial evaluation. Then we moved 1 year ahead, retrained it and evaluated it again. The proposal is to allow the model to learn different patterns in different periods. The results of each evaluation session are shown in the next section.

[2] `https://finance.yahoo.com`, accessed on March, 2020.

[3] `http://keras.io`, access on March, 2020.

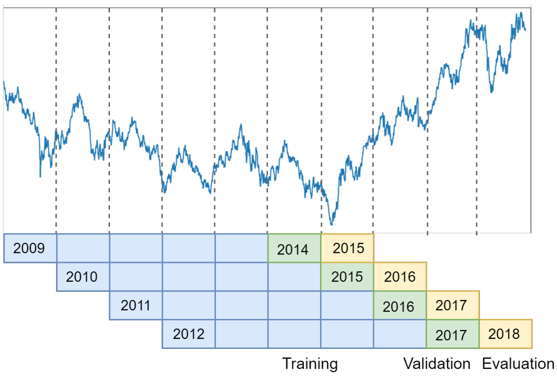

Figura 5. Training period is shown in blue, validation period in green and evaluation period in yellow

6. Results

6.1. Statistical evaluation

Each dataset is evaluated separately, we used weighted accuracy, precision, recall and F1. Furthermore, we compared our proposed model to CNN-TA [Sezer and Ozbayoglu 2018]. Our implementation of the CNN-TA is the best we could get given the information provided in the paper. Table 1 presents the metrics comparison for both DL models.

Tabela 1. Metrics comparison for MCNN-TC and CNN-TA

	MCNN-TC				CNN-TA			
Stock	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
BOVA11	**68.8%**	81.5%	**68.7%**	**72.3%**	63.0%	**81.6%**	63.1%	69.7%
ITUB4	**67.9%**	**81.1%**	**67.9%**	**69.2%**	59.5%	80.0%	59.5%	66.9%
PETR4	**64.9%**	80.1%	**64.9%**	69.4%	63.9%	**81.9%**	63.9%	**70.6%**
VALE3	**69.3%**	80.1%	**69.7%**	**72.7%**	56.9%	**81.0%**	56.9%	64.2%

For a trading system, the accurate classification task are important to make it profitable. In MCNN-TC, the accuracy for all datasets are approximately 66%. Precision rates are around 80% and for Recall, which measures the rate of real positive from all predicted positives of each class, the model achieves rates of approximately 66%. For F1-score, the model resulted in average, rates near to 71%. Most of the buy and sell points are classified correctly. We consider this a good result for time series pattern classification task.

As one can see, the results presented in Table 1 show that our model achieved a better performance compared to CNN-TA. The accuracy and Recall of CNN-TA are around 61%. The F1-score is approximately 67%. Overall, precision rates are tied to our model, in average 81%. The only case in which our model underperformed CNN-TA was in stock PETR4, losing in precision and recall. In summary, we showed that MCNN-TC is more successful in trading signal classification compared to CNN-TA and shows more stable behaviour.

It is important to mention that news expectation and some economic factors may interfere in the stock market making prices go up or down unexpectedly. There is no such historical data pattern in such situations to contribute for a more accurate prediction.

6.2. Financial evaluation

The MCNN-TC trading performance is compared to the BH strategy and CNN-TA model in terms of Return On Investment (ROI), as shown in Figures 6, 7, 8 and 9. In the BH strategy, the stock is bought at the beginning of the test data and sold at the end.

In the Brazilian stock market the default quantity to buy stocks is multiples of 100 and for ETFs is multiples of 10. The operations using DL methods are performed according to default quantity as follows: at first trading signal, the current action at that day is taken. If the label is Hold, no action is taken. If the predicted label is Buy, the stock is bought at that day with all the current available amount (when is not already bought), the leftover goes to cashier. If the label is Sell, all stocks are sold (when is not already sold). The start amount for evaluation is R$10,000.

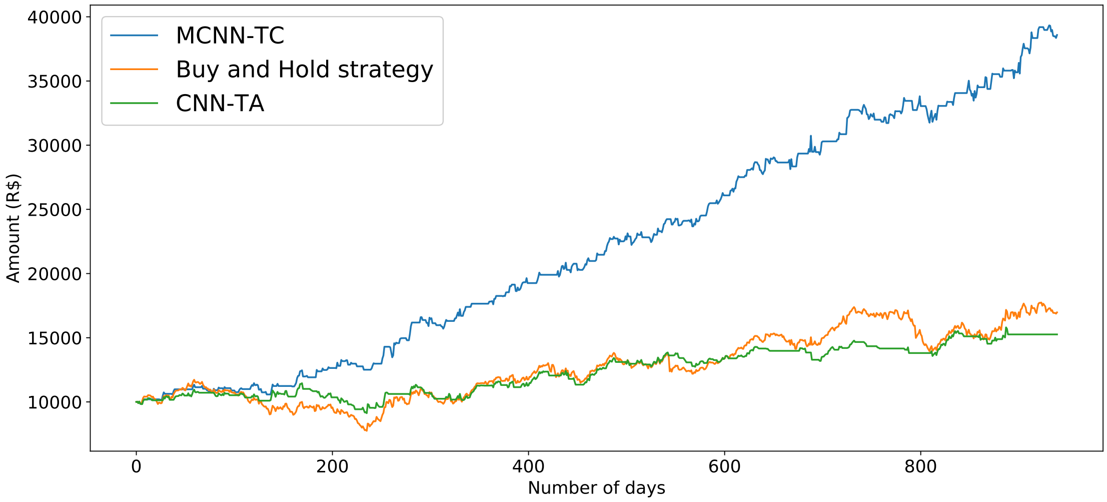

Figura 6. Accumulated ROI for BOVA11

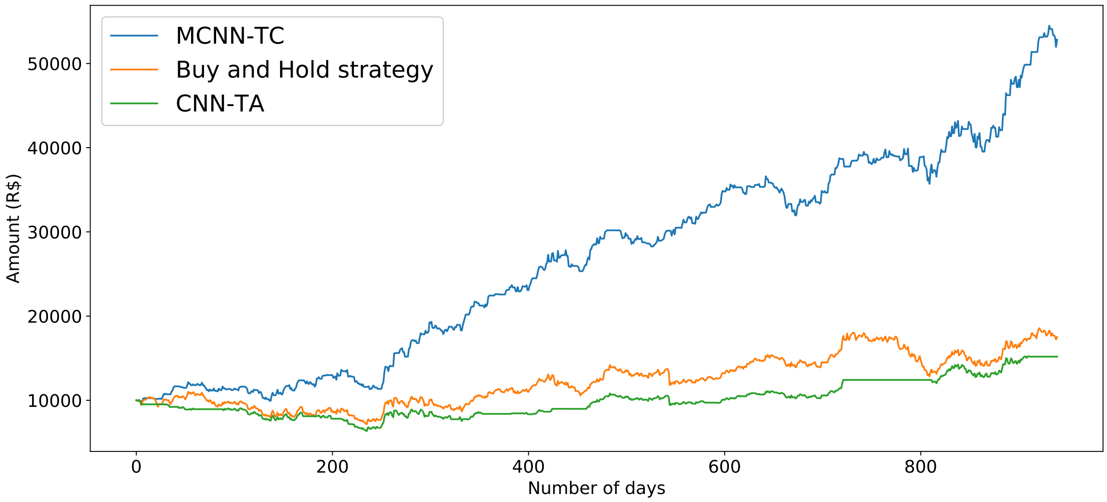

Figura 7. Accumulated ROI for ITUB4

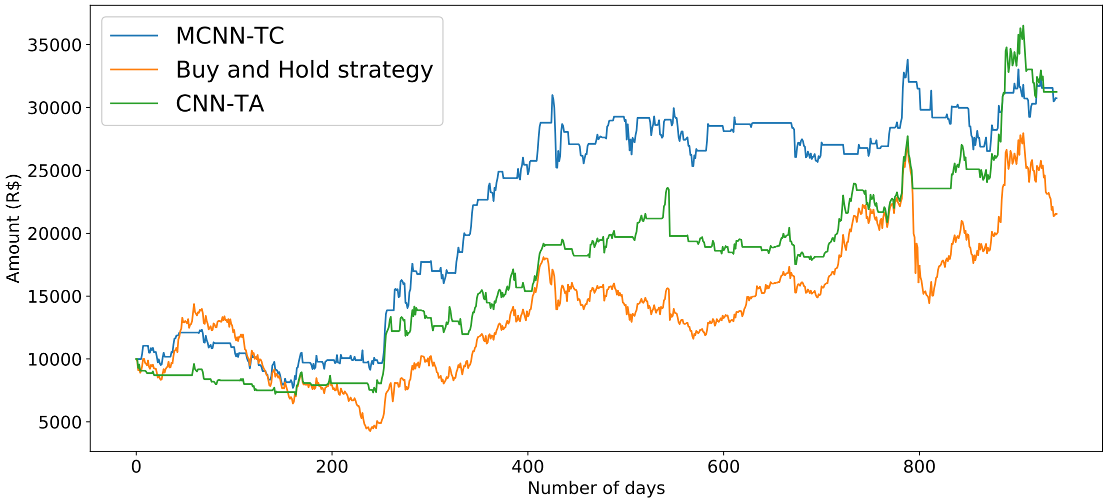

Figura 8. Accumulated ROI for PETR4

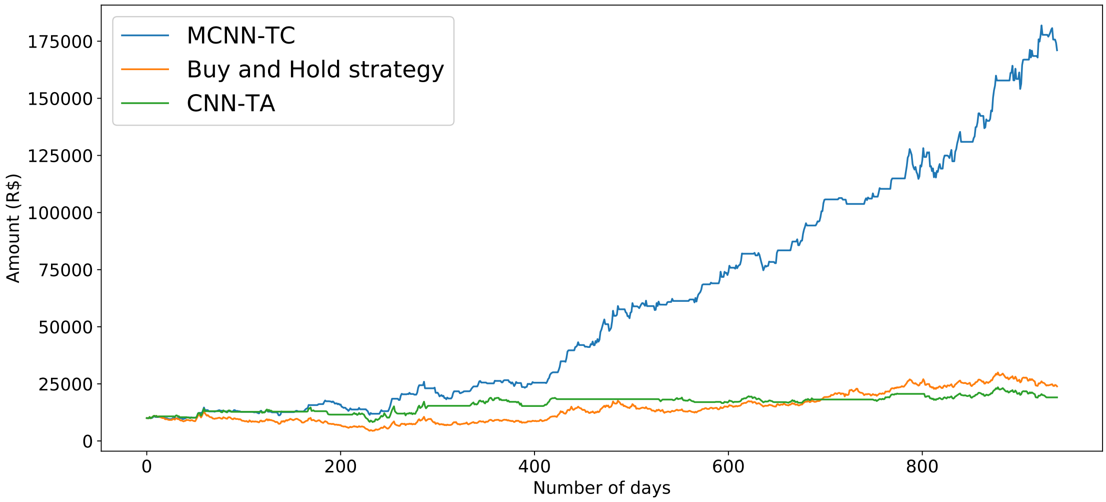

Figura 9. Accumulated ROI for VALE3

The results show that MCNN-TC had a better performance in all stocks/ETF, excepting in PETR4, achieving 207.3% compared to 212.4% from CNN-TA. We believe that news related to PETR4 in the evaluation period may have interfered in trading performance. However, both strategies were more profitable than Buy and Hold.

7. Conclusions and Future works

In this paper we proposed the use of a Multichannel Deep Convolutional Neural Network aiming to predict the best Buy and Sell moments for stock market. Our proposed model, MCNN-TC, was able to analyze time series data and perform a trading signal classification. We evaluated our model in Brazilian stock market and we used the top 3 most relevant Brazilian stocks in Ibovespa (ITUB4, PETR4 and VALE3) and an ETF which replicates Ibovespa portfolio to perform a financial evaluation in order to outperform the Buy and Hold strategy.

Our proposed model showed promising results for an automated trading system, and we believe it can help traders to take decision on the stock market. In future research, we intent to explore more complex model architectures, such as Recurrent Neural Networks, combined with Convolutional Neural Network, in order to have a more accurate prediction.

In addition, we also intent to investigate other features. In MCNN-TC we only considered some input features. We believe that there may exist informative features, such as economics data, that may improve the prediction capability. Also, there are indicators we did not explore in this research that may improve the classification accuracy.

Finally, our labeling method proved to be more profitable when compared to the method proposed by CNN-TA, despite only considering the closing price at that moment. We plan to develop more fine-grained labeling strategy and study its impact on the classification.

8. Acknowledgment

The authors acknowledge the São Paulo Research Foundation (FAPESP Grant 2019/07665-4) and CNPq (Grants 425860/2016-7 and 307027/2017-1) for supporting this project. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

Bao, W., Yue, J., and Rao, Y. (2017). A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PloS one*, 12(7):e0180944.

Boureau, Y.-L., Ponce, J., and LeCun, Y. (2010). A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 111–118.

Cavalcante, R. C., Brasileiro, R. C., Souza, V. L., Nobrega, J. P., and Oliveira, A. L. (2016). Computational intelligence and financial markets: A survey and future directions. *Expert Systems with Applications*, 55:194–211.

Colby, R. W. and Meyers, T. A. (1988). *The encyclopedia of technical market indicators*. Dow Jones-Irwin Homewood, IL.

Fischer, T. and Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Hoseinzade, E. and Haratizadeh, S. (2019). Cnnpred: Cnn-based stock market prediction using a diverse set of variables. *Expert Systems with Applications*, 129:273–285.

Jain, S., Gupta, R., and Moghe, A. A. (2018). Stock price prediction on daily stock data using deep neural networks. In *2018 International Conference on Advanced Computation and Telecommunication (ICACAT)*, pages 1–13. IEEE.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krauss, C., Do, X. A., and Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the s&p 500. *European Journal of Operational Research*, 259(2):689–702.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., and Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems*, pages 396–404.

Malkiel, B. G. and Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, 25(2):383–417.

Mittal, A. and Goel, A. (2012). Stock prediction using twitter sentiment analysis. Standford University, CS229 (http://cs229.stanford.edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis.pdf).

Selvin, S., Vinayakumar, R., Gopalakrishnan, E., Menon, V. K., and Soman, K. (2017). Stock price prediction using lstm, rnn and cnn-sliding window model. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1643–1647. IEEE.

Sezer, O. B. and Ozbayoglu, A. M. (2018). Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. *Applied Soft Computing*, 70:525–538.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Tsai, Y.-C., Chen, J.-H., and Wang, C.-C. (2019). Encoding candlesticks as images for patterns classification using convolutional neural networks. *arXiv preprint arXiv:1901.05237*.

Vargas, M. R., De Lima, B. S., and Evsukoff, A. G. (2017). Deep learning for stock market prediction from financial news articles. In *2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, pages 60–65. IEEE.