A Methodology for Definition and Refinement of a LSTM Stock Predictor Architecture using iRace and NSGA-II

Thiago F. Costa[1], **Elizabeth F. Wanner**[1], **Flávio V. C. Martins**[1], **André R. da Cruz**[1]

[1] Departamento de Computação
Centro Federal de Educação Tecnológica de Minas Gerais,
Av. Amazonas 7675, Nova Gameleira, Belo Horizonte, MG 30510-000, Brasil

thiagofigcosta@gmail.com, {efwanner,flaviocruzeiro,dacruz}@cefetmg.br

Abstract. *This paper presents a novel methodology aiming to define and refine a LSTM architecture applied to predict stock market prices. The methodology, dubbed* STOCK-PRED: THE LSTM PROPHET OF THE STOCK MARKET*, uses iRace and NSGA-II algorithms. The LSTM is built in two steps: (i) initially, iRace determines a robust set of hyperparameters using a compound objective function; afterwards, (ii) one of the best structures is used to define a tiny search space for the NSGA-II populations. In this step NSGA-II optimizes, simultaneously, the Mean Squared Error, in relation to price prediction, and the Accumulated Accuracy Rate for a time horizon of seven days, relative to growth price tendency. The methodology is tested considering stock market tickers from USA and Brazil. Even in a challenging scenario surrounded by possibly turbulent events, the Stock-Pred predicts the prices based on historical data and machine learning techniques. Since the optimization problem is noisy and the objective functions have a high computational cost, we consider a low budget in relation to the number of fitness evaluations. The analysis of the non-dominated solution indicates that the proposed methodology is promising, achieving a MAPE of 1.279%, 1.564% and 2.047% for BVSP, IBM and AAPL stocks respectively.*

Keywords: *Stock market forecasting. LSTM Networks. NSGA-II. iRace.*

1. Introduction

Investments on the stock market are a type of variable income (investment where the profit is not known at the time of application). Those are becoming more popular throughout the years. In Brazil, for example, there were at least 1.9 million new investors from 2019 to 2021. In this period, the investors were more active and the exchanged stock volume increased by 181% when compared to previous years [B3 2021].

The stock market is a complex system with difficult foreseeability. There are many elements which may influence the price, those can be interpreted by a fundamental or a technical analysis [Palepu et al. 2020, Kirkpatrick II and Dahlquist 2010]. The noisy aspect of the financial market is grounded on [Malkiel 1973], which states that stocks follow random and unpredictable paths. This characteristic enables the financial market to be analyzed as a chaotic system [Sahni 2018]. Inversely, the efficient market theory [Fama 1970] affirms that a stock value automatically and instantly reflects all available historical data. Therefore, it would not be possible to obtain returns above the market average, since every agent with access to the same information would perform similarly.

The previous idea supports this paper by considering predictability in the stock market. We aim here to obtain above-average profits by assuming that information about the market is not uniformly distributed among its agents. Moreover, this work suppose that an automatic classifier can obtain a more refined analysis than a human agent, since it is able to detect information implicitly encoded into the stock prices with more precision. A number of models and algorithms, such as regression, ARIMA, support vector machine, fuzzy models, genetic algorithm, random forest, k-nearest neighbor, and neural network, can identify trends on the data and predict the stock values or movement despite the level of randomness and risks on the aforementioned system [Shah et al. 2019].

This work deals with Long Short-Term Memory (LSTM), a type of RNN, [Menon et al. 2019] to predict stock price movements. In general, neural networks have many possible combinations of hyperparameters, resulting in a high number of architectures which, in the majority of the cases, cannot be evaluated one by one using trial and error. Therefore, it is important to fully or partially adopt a good Neural Architecture Search (NAS) technique to assist the choice between models.

The proposal of this work is to present the fine-tuned multi-objective STOCK-PRED – THE LSTM PROPHET OF THE STOCK MARKET – a methodology which specializes in a stock price dataset to predict future closing prices. A case study is presented here, where initially, the offline parameter tuning algorithm iRace [López-Ibáñez et al. 2016] is applied to investigate a good and robust region for the hyperparameters of the LSTM architecture, using a composed fitness function. Following, using a tiny search space defined by a solution, previously obtained, the NSGA-II is adopted to simultaneously optimize the Accumulated Accuracy Rate (AAR) for trend movements and the Mean Squared Error (MSE) for price prediction. The NSGA-II search space is smaller than the iRace one, enabling a fine local search without wasting computational resources.

The approach was developed in Python using the Keras library and it is available under the MIT license through the following link https://github.com/thiagofigcosta/stock-pred-v2. The methodology is illustrated in computational experiments considering stock market tickers from USA and Brazil, using financial data from public sources. Since the optimization problem is noisy and the objective functions have a high computational cost, it is considered a low budget in relation to the number of fitness evaluations. The non-dominated solution set is analyzed and the results indicate that Stock-Pred is promising.

2. Related Work

The stock market forecast was analyzed in [Selvin et al. 2017] using LSTM, Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), in addition to the Auto-Regressive Integrated Moving Average (ARIMA). Their model was tested for pharmaceutical and technology companies, with normalized data from 2014 to 2015. The network was encoded to predict the next 10 minutes using the previous 100. The network was trained for 1000 epochs with a checkpoint callback, using the MSE loss function. The CNN had the best performance, achieving an overall error of 2.36%, 3.63%, and 8.96% for the companies Infosys, Cipla, and TCS, respectively.

A model using 2-layered LSTM followed by 2 dense layers was defined in [Roondiwala et al. 2017]. The dense layers activation functions used were ReLU and Linear. The network was optimized using RMSProp [Xu et al. 2021], and trained with

the normalized Open, High, Low and Close prices as features from 2011 to 2016. They achieved a RMSE of 0.00859 when training the model for 500 epochs.

The authors in [Li et al. 2018] used LSTM with the Attention Mechanism [Bahdanau et al. 2014] to forecast the open prices of stocks. They used data between 2013 and 2017 to train the network with RMSProp. The last 300 days of the dataset were used for validation and testing purposes. The dataset was normalized, and the network was encoded to use the previous 10 days to predict the next one. The results were compared with other models, such as other variations of LSTM networks and DA-RNN (Data Associated RNN). The average MSE of the best model was $1.012 \cdot 10^{-3}$.

Instead of just predicting the stock market movement, the authors of [Chen et al. 2015] encoded their LSTM Network labels into 8 binary classes. With these classes, they were able to predict the stock movement alongside with its magnitude in percentage. The magnitude is an information of "how much" the stock went up or down in comparison with the previous value. They trained 3 LSTM layers connected to a Dense Softmax Layer using RMSprop and a batch size of 64. The dataset ranged from 2013 to 2015, and was encoded to use 10 past days to predict 3 days. The Categorical Cross Entropy loss function was used on the network. The accuracy achieved was 27.2% which seemed low when comparing with other works. However this accuracy cannot be compared with all works, since it might be lower because aside from predicting the movement, it also predicts its magnitude, making the problem a more complex one.

The authors in [Moghar and Hamiche 2020] aimed to forecast the closing price of stocks and compared the results with the ARIMA model. They used 50 past days to predict a day in the future using a network with 4 LSTM layers, 4 dropout layers and one dense layer. The dataset used for the experiments contained data for tickers GOOGL and NKE stocks from 2004 to 2019 and from 2010 to 2019, respectively. They trained the network for up to 100 epochs. The results were presented using the MSE.

The offline tuning method iRace has been successfully applied to find good parameters set for machine learning methods and to solve optimization heuristics. In [Cramer et al. 2017], the authors tuned via iRace and experimented seven classification methods for rainfall prediction in weather derivatives. In [Kampouridis and Otero 2017], iRace was employed to adjust the parameters of a mono-objective genetic algorithm which determines an event-based time scale. The goal was to find a trading strategy maximizing profitability in foreign exchange markets. iRace was also applied in [Mohamed and Otero 2020] to tune implementations of a multi-objective genetic algorithm and a particle swarm optimization to decide when to buy or sell on the stock market.

In the literature, there are works applying the NSGA-II [Deb et al. 2002] to determine LSTM architectures. For example, in [Li et al. 2019] the authors tackled the problem of acoustic scene classification with a CNN model. They achieved improvements using NSGA-II for the NAS, with the accuracy and the computational cost as objectives. In [Chui et al. 2021], the authors estimate the remaining useful life of equipment for optimal predictive maintenance using an ensemble of RNN and LSTM whose weights were tuned using NSGA-II, by minimizing two error measures. The authors in [Chu et al. 2021] used CNN to address the problem of single image super resolution and adopted NSGA-II as the NAS technique with an elastic search space to perform both micro and macro search.

In this work, iRace is adopted to find a tiny and promising region, based on a composite fitness function. The region is then be used as a search region in a second multi-objective optimization phase. Aiming to find a set of non-dominated NAS for the Stock-Pred, this work uses the NSGA-II as the heuristic which optimizes the MSE, in relation to price prediction, and the AAR, relative to growth price tendency.

3. Methodology

This section describes the steps made to develop the Stock-Pred and to carry out the experiments. By following the actions described below, similar results should be achieved.

3.1. Data Crawling

The data used in this paper was obtained from Yahoo Finance[1]. The historical data presents the following features: (i) Date; (ii) Close price; (iii) Open Price; (iv) High price; (v) Low price; (vi) Adj. Close, which is the close price after discounting dividends and other charges; (vii) Volume of exchanged stocks. Available values may vary according to local holidays, in reference to the stock, but they are generally available for all work days. Data from 01/Jan/2016 to 07/May/2021, for the stocks of AT&T (T), IBM (IBM) and Apple (AAPL), and also the Brazilian index (BVSP) were used.

3.2. Data Processing

The next step is to pre-process the data. First, all dates with null or empty values have been filtered out from the data. The Adj. Close feature has been removed from the dataset. At this point, with all unwanted rows and columns removed, the enrichment process takes place. This process consists of generating new features using the existing ones. Inspired by [Nelson et al. 2017], the close price, of a reference day d, was used to compute the Standard Moving Average (SMA) and the Exponential Moving Average (EMA) both for 13 and 21 days, and the Log Return (LR). The features OC, OH, OL, CH, CL and LH, which represents a ratio between the prices of a given stock [Liu et al. 2018] have also been created. The formulas for these features are presented in Eq. (1).

$$\begin{aligned} OL &= (Low - Open)/Open & \quad LH &= (High - Low)/Low \\ OC &= (Close - Open)/Open & \quad CL &= (Low - Close)/Close \\ OH &= (High - Open)/Open & \quad CH &= (High - Close)/Close \\ LR &= \log(Close_d) - \log(Close_{d-1}) \end{aligned} \tag{1}$$

After this step, the final set of features are composed by the date, the OC, the OH, the OL, the CH, the CL, the LH, the Volume, both EMA, both SMA and the LR. All those features have been normalized using the min-max scaling, considering each feature individually, due to their difference in scale.

The last step in the pre-processing pipeline is to adapt the dataset for LSTM [Hochreiter and Schmidhuber 1997]. The design chosen for the networks implies that B sequential data points, or days, of the past will be used to forecast the close price for the next F days. Therefore, the data must be arranged using a rolling window, with all features being present B times on the input, and the following F close prices on the output. This arrangement enables the forecast of a given date to have up to F results.

[1]See: https://finance.yahoo.com/

3.3. LSTM Networks

The goal is to create a LSTM model to forecast the close price of a given stock. For training 56% of the dataset was used, with 24% for the cross-validation. Finally, for the tests 6.2% of the dataset was used, which corresponds to the interval that starts on 07/Jan/2021 and ends on 07/May/2021. The gradients were clipped to 0.5 to avoid the Exploding Gradients issue. On stateless networks, the cell memory is reset after each batch, meanwhile, on our implementation, stateful LSTM networks reset the memory at every epoch. The LSTM hyperparameters can be categorized into two types, Layer and Network. The Network Hyperparameters is a set that configures the model, hence they have just a single instance per model. Layer Hyperparameters define the characteristics of the layer, meaning that every single layer has a unique instance of those. Three extra configurations have been added to the network architecture: (i) "Dense at the end", switches between a dense layer with linear activation or another LSTM Layer at the end of the network; (ii) "unit forget bias", an option that adds one on the bias vector of the LSTM forget gates at initialization; (iii) "go backwards", an option that reverses the layer input sequence.

To assess the performance of the models, some measures have been employed, such as the Mean Absolute Percentage Error (MAPE), the MSE, the cosine similarity, and the R^2. These measures are directly related to the close price of stocks. Another set of measures was also adopted, in this case, the stock price is converted into tendency, i.e. whether the stock will rise or not. Since the stock movement is a binary problem, the measures are the accuracy, the Area Under the Receiver Operating Characteristic Curve (ROC AUC), and the F1-Score. Since each forecast can have up to F results, to compute these measures, the multiple available results for a given day are grouped, and to provide a single output, the average between the first and last available results is calculated. The AAR, a binary measure, is applied to consider every available movement prediction. To compute the AAR, F sequential dates are chosen from the test dataset, then, the accuracy of each movement prediction for each day is calculated using the real value of the previous day as ground truth. Finally the average of all accuracy values results into the AAR.

To reduce the computational cost of the LSTM networks and, at the same time be able to increase the amount of epochs, three training callbacks have been created. The checkpoint callback saves the weights of the network with the best validation loss so far. This checkpoint can be restored later, enabling the model with the best performance during the training to be used. The learning rate callback reduces the learning rate by a factor after a "reduce patience" amount of epochs without improvements on the validation loss. The last one is the early stop callback, that interrupts the training prematurely after a "stop patience" amount of epochs. The loss function adopted the networks was the MSE.

3.4. Wide NAS with iRace

iRace interactively samples distinct parameter configurations and evaluate them across multiple problem instances to find a good and robust parameter set configuration for the problem instances.

$$\min u(\mathbf{p}) \quad s.t.: \ \mathbf{p} \in \Pi \tag{2}$$

$$u(\mathbf{p}) = -\frac{3 \cdot AAR(\mathbf{p}) + F1\_Score(\mathbf{p}) + Accuracy(\mathbf{p})}{5 * MSE(\mathbf{p})} \tag{3}$$

In the absence of any information related to the network architecture, to the hyperparameter search intervals, and to the ideal AAR, an initial search space has been defined based on literature values. The final search space for this step, is created by adding an extra margin to the intervals. The iRace is used to find a robust parameter set $\mathbf{p}$ in the search space Π by solving the tuning problem of Eq. (2) with a single noisy and weighted utility function $u : \Pi \rightarrow \mathbb{R}$ computed as Eq. (3).

Table 1. Wide Hyperparameters Search Space

Hyperparameter	Search Space	Hyperparam. Type
Backward Samples (B)	[5, 60]	Network
Forward Samples (F)	[7, 14]	Network
Hidden LSTM Layers	[0, 5]	Network
Maximum Epochs	[500, 5000]	Network
Patience to Stop	[1000, 5000]	Network
Patience to Reduce Learning Rate	[0, 1000]	Network
Learning Rate Reduce Factor	[0, 0.2]	Network
Batch size	[0, 128]	Network
Stateful	{ True, False }	Network
Optimizer	{ SGD, Adam, RMSPro }	Network
Dense at the end	{ True, False }	Network
Activation Function	{ ReLU, Sigmoid, Tanh, Exponential, Linear }	Layer
Recurrent Activation	{ ReLU, Sigmoid, Tanh, Exponential, Linear, Hard Sigmoid }	Layer
Layer size	[10, 200]	Layer
Dropout	[0, 0.3]	Layer
Recurrent Dropout	[0, 0.3]	Layer
Use bias	{ True, False }	Layer
Unit forget bias	{ True, False }	Layer
Go backwards	{ True, False }	Layer

The iRace is set up to have a maximum budget of 777 and to be a elitist race with two elite individuals per iteration, at which one of them must be a new one after each iteration. The algorithm is also configured to run seven iterations of a non-deterministic race, with a μ of five and a confidence level of 0.95. The LSTM search space Π is presented in Table 1. To specialize the search space of the LSTM concerning to stock price prediction, the AT&T ticker was used in the tuning process. After the race, the selected model was retrained and used to determine the tiny feasible region to be employed in the multi-objective optimization phase for the tickers AAPL, IBM and BVSP index.

3.5. Fine-tunning NAS with NSGA-II

With a partial solution $\tilde{\mathbf{p}}$ coming from iRace, a smaller search space, $\tilde{\Pi}$, around it needs to be defined. The idea is to create new intervals which are plus and minus 10% of the original $\tilde{\mathbf{p}}$ numeric values that remains inside Π. This new region is the feasible set of the multi-objective problem of Eq. (4), in which $u_1 : \tilde{\Pi} \rightarrow \mathbb{R}$ and $u_2 : \tilde{\Pi} \rightarrow \mathbb{R}$ are, respectively, the MSE and AAR. The former is a regressive measure and the latter is binary measure.

$$\min \mathbf{u}(\mathbf{p}) = [u_1(\mathbf{p}), -u_2(\mathbf{p})] \quad s.t.: \ \mathbf{p} \in \tilde{\Pi} \tag{4}$$

Table 2. iRace Best Network Hyperparams.

Hyperparameter	Value
Backward Samples	31
Forward Samples	7
Hidden LSTM Layers	1
Maximum Epochs	3183
Patience to Stop	1827
Patience to Reduce	228
Reduce Factor	0.05265
Batch size	46
Stateful	False
Optimizer	Adam
Dense at the end	False

Table 3. iRace Best Layer Hyperparameters

Hyperparameter	Layer 1
Activation Function	Sigmoid
Recurrent Activation	Tanh
Layer size	67
Dropout	0.24864
Recurrent Dropout	0.15223
Use bias	True
Unit forget bias	True
Go backwards	False

Table 4. Evaluation of the model optimized by iRace

Stock	MSE	AAR	MAPE	Cosine	R^2	F1	Accuracy	ROC AUC
AAPL	0.004862	57.143%	7.645%	0.99774	-	0.735	0.750	0.753
BVSP	0.000814	49.286%	7.397%	0.99989	-	0.928	0.920	0.924
IBM	0.000349	91.287%	2.246%	0.99986	0.913	0.933	0.923	0.918

This second meta-optimization phase is performed by the NSGA-II, using the PyMOO implementation [Blank and Deb 2020]. The NSGA-II was configured to run in parallel, with a population size of 30 for parents and offspring, eliminating duplicates, and with a maximum of 500 function evaluations. The hyperparameters were mapped into numbers, and PyMOO was set to use Real Random, Int Random and Bin Random as initialization algorithms, Real SBX, Int SBX and Bin HUX as crossover methods, and finally Real PM, Int PM and Bit Flip as mutation algorithms.

Three computational experiments were designed to observe the behavior of the Stock-Pred for the tickers IBM and APPL, and BVSP index. Some solutions of the obtained non-dominated set were analyzed according to some quality measures, in addition to those used in the optimization process. It can be noted that the optimization problem is noisy and the objective functions in both phases have a high computational cost. For this reason, a low budget in relation to the number of fitness evaluations has been set.

4. Experiments and Results

The experiments were executed, during 14 days, in a shared server with an Intel Xeon E5-2630 v3 CPU @ 2.40GHz Octa-Core processor, 32 GB of RAM memory and a Nvidia Tesla K40c graphics card. The first experiment was the iRace search on the wider search space. The best obtained individual is described in Tables 2 and 3 and its independent set of evaluations (train and test) for IBM, AAPL and BVSP tickers are shown in Table 4.

IBM stocks had the best results for iRace, reaching a MAPE of 2.246% and being the only stock to obtain a valid value for the R^2, that is, the forecast of the others presented a very different curve from the original one [Nakagawa and Schielzeth 2013]. The reason for this may be the fact that both AT&T, which was used in iRace, and IBM

are dividend aristocrats, which means that these stocks may be similar, and therefore the IBM had a better performance than the others. When comparing results between different models, especially when dealing with different stocks as well, not all measures are appropriate. The MSE, for example, reflects in its value the price scale, the interval considered and the architecture of the model. For a better comparison, in similar conditions, it is recommended to consider the measures MAPE, R^2, Cosine Similarity and AAR.

After the end of first tuning phase, intervals around the best hyperparameters found by iRace were determined for the multi-objective step. It was defined a finer search space to be optimized by NSGA-II. The feasible region is described in Table 5.

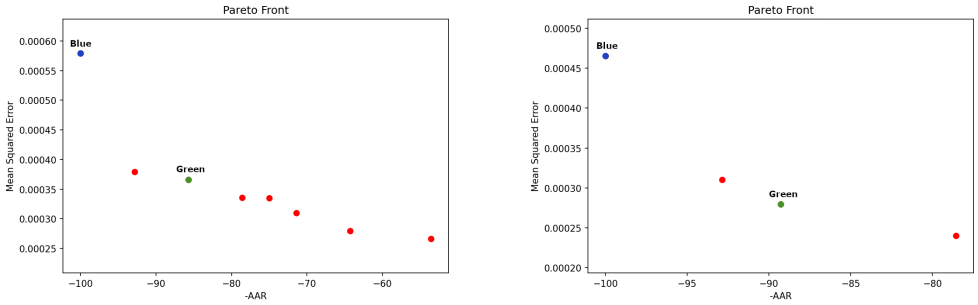
Table 5. Fine Network Search Space

Hyperparameter	Search Space	Hyperparam. Type
Backward Samples (B)	[27, 34]	Network
Forward Samples (F)	7	Network
Hidden LSTM Layers	[0, 2]	Network
Maximum Epochs	[2900, 3400]	Network
Patience to Stop	[1700, 1900]	Network
Patience to Reduce Learning Rate	[200, 250]	Network
Learning Rate Reduce Factor	[0.045, 0.055]	Network
Batch size	[40, 50]	Network
Stateful	False	Network
Optimizer	{ Adam, RMSPro }	Network
Dense at the end	{ True, False }	Network
Activation Function	{ ReLU, Sigmoid, Tanh }	Layer
Recurrent Activation	{ ReLU, Sigmoid, Tanh }	Layer
Layer size	[60, 74]	Layer
Dropout	[0.20, 0.28]	Layer
Recurrent Dropout	[0.12, 0.17]	Layer
Use bias	{ True, False }	Layer
Unit forget bias	{ True, False }	Layer
Go backwards	{ True, False }	Layer

Table 6. Results NSGA-II

Model & Stock	MSE	AAR	MAPE	Cosine	R^2	F1	Accuracy	ROC AUC
Green - AAPL	0.000747	96.429%	2.047%	0.99978	0.932	0.830	0.827	0.827
Green - BVSP	0.000279	89.286%	1.279%	0.99993	0.899	0.847	0.824	0.821
Green - IBM	0.000366	85.714%	1.564%	0.99989	0.905	0.831	0.796	0.783
Blue - AAPL	0.005202	100.00%	5.564%	0.99968	-	0.462	0.462	0.462
Blue - BVSP	0.000465	100.00%	1.644%	0.99993	0.124	0.909	0.896	0.896
Blue - IBM	0.000579	100.00%	2.046%	0.99988	0.689	0.689	0.620	0.602

NSGA-II was run for each ticker. Some Pareto fronts can be found in the Figure 1. Two solutions, the blue and green ones, were chosen to be analyzed for each non-dominated set. The blues have the AAR measure maximized, while the greens have a more balanced profile. These solutions were tested and the results are presented in Table 6. The green models achieved a better overall performance regarding the majority



(a) IBM stock **(b)** BVSP Stock

Figure 1. Pareto front for multiple stocks, all dots are non-dominated solutions, the blue and green dots represents the chosen models

Table 7. NSGA-II Optimum Green Network Hyperparameters

Hyperparameter	AAPL	BVSP	IBM
Backward Samples	31	30	29
Forward Samples	7	7	7
Hidden LSTM Layers	1	2	1
Maximum Epochs	3166	3027	3282
Patience to Stop	1782	1711	1878
Patience to Reduce	240	234	236
Reduce Factor	0.051925	0.04762	0.05008
Batch size	48	44	40
Stateful	False	False	False
Optimizer	RMSProp	Adam	RMSProp
Dense at the end	True	True	False

measures due to its balanced profile. The binary measures F1-Score, Accuracy and ROC AUC were the only ones that in some cases showed a slight worsening when comparing the iRace results with the green models, except for AAR, R^2 and MSE small changes on IBM stocks due to the AT&T similarity. These measures are calculated using the average between the first and last available predictions. Since they are binary and do not use all the calculated predictions, they can be more sensitive and may not fully reflect the quality of the model. It is therefore important to analyze all measures as a whole.

The hyperparameters for the green models are highlighted in Tables 7 and 8. Each hidden layer of the models are represented by the letter L followed by the number of the layer on the Table 8. The best MSE and MAPE across the evaluated stocks were achieved by the Green BVSP model. The forecasts for this model can be seen in Figure 2, that shows how close the forecasts were to the actual values.

Different from the partial results of iRace, in which only one of the evaluated stocks presented good results, now that the fine search was made for each stock in specific, all of them can reach good results, and NSGA-II proved to be efficient. The iRace also fulfilled its role by reducing the search space in a faster run, so that NSGA-II could improve the results. The two NAS methods are not being compared, in this work they are complementary, both have strengths and weaknesses, and both can be used for NAS.

Table 8. NSGA-II Optimal Green Layer Hyperparameters

Hyper-parameter	AAPL-L1	BVSP-L1	BVSP-L2	IBM-L1
Act. Function	Sigmoid	ReLU	ReLU	Sigmoid
Rec. Act.	Tanh	Sigmoid	Sigmoid	Tanh
Layer size	71	61	73	65
Dropout	0.24019	0.23468	0.26155	0.25468
Rec. Dropout	0.14546	0.14437	0.13878	0.14771
Use bias	False	True	False	False
Unit forget b.	True	False	True	True
Go backwards	True	True	False	True

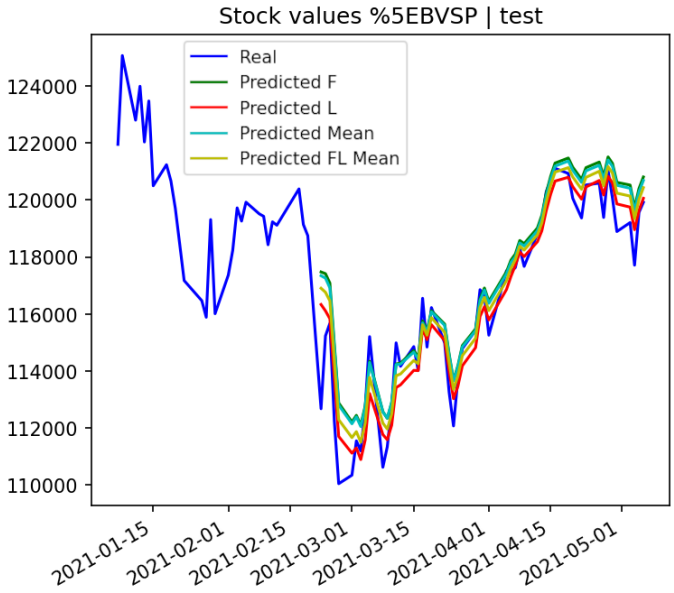


Figure 2. Green model forecasts for BVSP, dates in format yyyy/MM/dd

5. Conclusions and Further Work

Stock market forecasting is not simple, however it has been shown to be an achievable task. In the present work the Stock-Pred was presented. It uses tuned LSTM networks to forecast the closing price of stocks, for the tickers AAPL, IBM and the BVSP index. The results were satisfactory, with outstanding measures such as a MAPE of 1.28% and a R^2 of 0.932. The methodology used for adjusting the hyperparameters also proved to be efficient. Even though promising results were achieved, there is still much that can be improved both in terms of LSTM architecture and NAS networks.

The measures are crucial because they reflect the quality of the model. They determine whether the results are satisfactory or not, and also allow to compare multiple models. Considering this, the measures assume an even more important role when performing a NAS on the problem. Since they determine how the search will be conducted, impacting the results directly. With these considerations in mind, it is worthwhile to test new measures, such as simulated profit, on future work, especially those that, similar to MAPE and R^2, are not affected by scale, interval, or architecture. Other methods for ag-

gregating multiple predictions can be tested to calculate binary measures, instead of using the average between the first and last available prediction.

There are also improvements that can be made on the network architecture, such as increasing the number of features, by adding more market indicators through the TA-lib[2]. We can choose which features should be included on the final dataset by either selecting them during the NAS or using some other feature selection technique. Since a given date can have up to F predictions, a significant improvement would be to connect the multiple predictions to another neural network, in order to retrieve only a single result per date.

Although the blue models reached the highest AAR, their performance in other measures was penalized, even considering that all Pareto front solutions should be equivalent to each other. Extending the experiments by evaluating more stocks, including ones from different categories, can be beneficial not only to have more meaningful results, but also to detect failure points. And, finally, we consider for future works, the application of methods for many-objectives [Ishibuchi et al. 2008] due to the various measures that can be adopted as objective criteria during NAS.

References

B3 (2021). Uma análise da evolução dos investidores na b3. (visited on 2021-11-03).

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Blank, J. and Deb, K. (2020). pymoo: Multi-objective optimization in python. *IEEE Access*, 8:89497–89509.

Chen, K., Zhou, Y., and Dai, F. (2015). A lstm-based method for stock returns prediction: A case study of china stock market. In *IEEE international conference on big data*.

Chu, X., Zhang, B., Ma, H., Xu, R., and Li, Q. (2021). Fast, accurate and lightweight super-resolution with neural architecture search. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 59–64. IEEE.

Chui, K. T., Gupta, B. B., and Vasant, P. (2021). A genetic algorithm optimized rnn-lstm model for remaining useful life prediction of turbofan engine. *Electronics*, 10(3):285.

Cramer, S., Kampouridis, M., Freitas, A. A., and Alexandridis, A. K. (2017). An extensive evaluation of seven machine learning methods for rainfall prediction in weather derivatives. *Expert Systems with Applications*, 85:169–181.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*.

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

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Ishibuchi, H., Tsukamoto, N., and Nojima, Y. (2008). Evolutionary many-objective optimization: A short review. In *2008 IEEE congress on evolutionary computation (IEEE world congress on computational intelligence)*, pages 2419–2426. IEEE.

[2]See: https://ta-lib.org/

Kampouridis, M. and Otero, F. E. (2017). Evolving trading strategies using directional changes. *Expert Systems with Applications*, 73:145–160.

Kirkpatrick II, C. D. and Dahlquist, J. A. (2010). *Technical analysis: the complete resource for financial market technicians*. FT press.

Li, H., Shen, Y., and Zhu, Y. (2018). Stock price prediction using attention-based multi-input lstm. In *Asian conference on machine learning*, pages 454–469. PMLR.

Li, J., Liang, C., Zhang, B., Wang, Z., Xiang, F., and Chu, X. (2019). Neural architecture search on acoustic scene classification. *arXiv preprint arXiv:1912.12825*.

Liu, S., Liao, G., and Ding, Y. (2018). Stock transaction prediction modeling and analysis based on lstm. In *IEEE Conference on Industrial Electronics and Applications*.

López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L. P., Birattari, M., and Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43–58.

Malkiel, B. G. (1973). A random walk down wall street. w. w.

Menon, A., Singh, S., and Parekh, H. (2019). A review of stock market prediction using neural networks. In *2019 IEEE International Conference on System, Computation, Automation and Networking (ICSCAN)*, pages 1–6. IEEE.

Moghar, A. and Hamiche, M. (2020). Stock market prediction using lstm recurrent neural network. *Procedia Computer Science*, 170:1168–1173.

Mohamed, I. and Otero, F. E. (2020). A multiobjective optimization approach for market timing. In *Proceedings of the Genetic and Evolutionary Computation Conference*.

Nakagawa, S. and Schielzeth, H. (2013). A general and simple method for obtaining r2 from generalized linear mixed-effects models. *Methods in ecology and evolution*, 4(2).

Nelson, D. M., Pereira, A. C., and de Oliveira, R. A. (2017). Stock market's price movement prediction with lstm neural networks. In *2017 International joint conference on neural networks (IJCNN)*, pages 1419–1426. IEEE.

Palepu, K. G., Healy, P. M., Wright, S., Bradbury, M., and Coulton, J. (2020). *Business analysis and valuation: Using financial statements*. Cengage AU.

Roondiwala, M., Patel, H., and Varma, S. (2017). Predicting stock prices using lstm. *International Journal of Science and Research (IJSR)*, 6(4):1754–1756.

Sahni, R. (2018). Analysis of stock market behaviour by applying chaos theory. In *9th international conference on computing, communication and networking technologies*.

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 *international conference on advances in computing, communications and informatics*.

Shah, D., Isah, H., and Zulkernine, F. (2019). Stock market analysis: A review and taxonomy of prediction techniques. *International Journal of Financial Studies*, 7(2):26.

Xu, D., Zhang, S., Zhang, H., and Mandic, D. P. (2021). Convergence of the rmsprop deep learning method with penalty for nonconvex optimization. *Neural Networks*, 139:17–23.