ORIGINAL ARTICLE

AURORA: an autonomous agent-oriented hybrid trading service

Renato A. Nobre[1,6] · Khalil C. do Nascimento[1] · Patricia A. Vargas[2] · Alan Demétrius Baria Valejo[3] ·
Gustavo Pessin[4] · Leandro A. Villas[5] · Geraldo P. Rocha Filho[1]

Abstract
Stock markets play an essential role in the economy and offer companies opportunities to grow, and insightful investors to make profits. Many tools and techniques have been proposed and applied to analyze the overall market behavior to seize such opportunities. However, understanding the stock exchange's intrinsic rules and taking opportunities are not trivial tasks. With that in mind, this work proposes AURORA: a new hybrid service to trade equities in the stock market, using an autonomous agent-based approach. The goal is to offer a reliable service based on technical and fundamental analysis with precision and stability in the decision-making process. For this, AURORA's intelligence is modeled using a rational agent capable of perceiving the market and acting upon its perception autonomously. When compared with other solutions in the literature, the proposed service shows that it can predict the gain or loss of value at the price of a stock with an accuracy higher than 82.86% in the worst case and 89.23% in the best case. Furthermore, the proposed service can achieve a profitability of 11.74%, overcoming fixed-income investments, and portfolios built with the Markowitz Mean-Variance model.

Keywords Time-series · Stock market prediction · Long short-term memory · Recurrent neural network

1 Introduction

Financial markets play an essential role in the behavior of a country's economy. Such markets make it easy for buyers and sellers to trade their financial assets, helping companies grow and investors make profits. The study and potential predictions of exchanges allow these investors to increase their profits and better understand their investments' behavior. For that reason, trading services based on decision-making models are getting more focus in multiple financial markets globally [21].

Among different types of financial markets, the stock market stands out in popularity. This type of market

✉ Geraldo P. Rocha Filho
geraldof@unb.br

Renato A. Nobre
renato.avellarnobre@studenti.unimi.it

Khalil C. do Nascimento
khalilcarsten@gmail.com

Patricia A. Vargas
p.a.vargas@hw.ac.uk

Alan Demétrius Baria Valejo
alanvalejo@ufscar.br

Gustavo Pessin
gustavo.pessin@itv.org

Leandro A. Villas
leandro@ic.unicamp.br

1   Department of Computer Science, University of Brasília
    (UnB), Brasilia, DF, Brazil

2   Edinburgh Centre for Robotics, Heriot-Watt University,
    Edinburgh, United Kingdom

3   Department of Computing, Federal University of São Carlos
    (UFSCar), São Carlos, Brazil

4   Vale Institute of Technology, Robotics Laboratory,
    Ouro Preto, MG, Brazil

5   Institute of Computing, University of Campinas
    (UNICAMP), Campinas, SP, Brazil

6   Computer Science Department, Università degli Studi di
    Milano, Milano, Italy

negotiates fractions of company assets, denominated stocks, or other public listed companies' financial instruments. When a specific number of stocks are bought or sold, they can be termed shares, while a generic amount being traded will be termed equities. Investors trading their equities in the market are guided by some form of prediction or analysis, such as studying the price's behavior or reading the company economic report.

The use of intelligent agents combined with Machine Learning (ML) techniques is one way to provide services related to predicting a particular asset in the financial market. Predicting those markets has been a growing field of study in the area of machine learning [9]. These advances, attributed to the expansion of the economy, might offer possibilities for everyone who seeks to earn profits from savings. Recent studies have been conducting research to predict financial markets using multiple methodologies and ML models [21]. These studies are categorized into two groups: (i) fundamental analysis, the company that underlies the stock itself instead of actual stock; and (ii) technical analysis, focused on predicting the future price of stocks by studying the trend in the past and present of a given stock price [15].

Some studies recommend sentiment analysis to predict the value of an asset [21, 24]. Other works use behavioral variables of economic aggregates (known as macroeconomic variables) or indices from markets around the globe [17, 21, 22]. However, most of the works use technical data of time-series from the asset to create models, or technical indicators [3, 4, 6, 23]. Furthermore, most of the works to date do not use companies' stock market equities in their research. Instead, they research on other financial assets, such as cryptocurrencies [7], foreign exchange market [4, 8, 13], stock market indices [1, 6, 17] and commodities [8]. Moreover, when a study is focused on a stock market, it is usually carried out on the Asian or European markets [21]. Notwithstanding, all the works mentioned so far do not explore a hybrid service based on technical and fundamental analysis and do not provide a mechanism for acting upon decisions, as explored in this research.

Overcoming the challenges and limitations mentioned above, this work supports the hypothesis that it is possible to model service with high accuracy to predict the stock's movement and stability to trade assets in the stock market based on technical and fundamental analysis.

With this in mind, this work proposes a new hybrid service to trade equities in the stock market: AURORA (**AU**to-nomous **R**ati**O**nal t**RA**der), using an autonomous agent-based approach. AURORA is based on a rational agent, a computational entity capable of perceiving the market and acting upon its perception autonomously. Rationality here means that the best actions will be taken to increase profit and reduce risks. For this, AURORA was

designed into three modules: (i) the insider, a predictor module that can predict the stocks' movements; (ii) a risk management module to deal with market volatility through the news; and (iii) an agent module, a rational actuator that aims to allocate the resources available to the agent and act upon them. Therefore, AURORA provides an end-to-end service, from the data analysis to the brokerage's trade orders.

As proof of concept, AURORA was designed to operate on the B3 stock exchange, the Brazilian market, previously known as BM&FBovespa. The considered portfolio has shares of 3 Brazilian companies: Petrobras (PETR3), Ambev (ABEV3), and Vale S.A. (VALE3). The experimental results show that the proposed service can predict the gain or loss of value at the price of a stock with an accuracy higher than 82.86% in the worst case and 89.23% in the best case compared with other solutions in the literature. Furthermore, the proposed service can achieve a profitability of 11.74%, overcoming fixed-income investments and portfolios built with the Markowitz Mean-Variance model. These results corroborate the hypothesis that AURORA is a reliable service to trade assets in the stock market with high precision and stability.

The remainder of this work is organized as follows. Sect. 2 shows related works in the field of financial market prediction in both fundamental and technical analyses approach. Section 3 proposes and explains the AURORA while laying out the theoretical foundation necessary for understanding its behavior. Section 4 explains the methodology used to validate AURORA and presents the results obtained from the experimental procedure. Finally, Sect. 5 presents the main conclusions and future research.

2 Related works

In the last few years, several works have been published in stock market prediction using machine learning [4, 6, 21–23, 29]. This section presents the challenges in this area. These studies are presented in two key categories, technical and fundamental analysis. Despite the progress made in this area, so far was not found a hybrid service based on technical and fundamental analysis to trade assets in the desired stock market [21].

For the fundamental analysis approach, Malagrino *et al.* [17] investigates the usage of Bayesian networks to verify the extent to which stock market indices from around the globe influence the main index at the B3 stock exchange in Brazil. For this, index direction was used as inputs to a network, moving through 24 and 48 hours cycles and outputting the next day closing direction for the index. The modeled Bayesian Networks allowed a further advantage of being more straightforward and tractable for its users

than related literature. However, doing full Bayesian learning is computationally costly, and its networks tend to perform poorly on high dimensional data. Nti *et al.* [22] proposes a random forest-based model for feature selection of macroeconomic variables with a leave-one-out cross-validation tactic and an Long Short-Term Memory (LSTM) for stock market prediction enhanced. The purpose was to examine the degree of significance between historical prices, different sectors, and macroeconomic variables to predict a monthly stock price. However, the proposed model provides little control over the stock market prediction's decision-making process since the random forest can feel like a black-box technique.

Still focused on the fundamental analysis, researchers' primary approach is a sentimental analysis of social network sites, such as Twitter and Facebook [21]. In this context, Preis *et al.* [24] suggests that massive new data sources resulting from human interaction with the internet may offer a perspective on the behavior of a market. The study found patterns that might be interpreted as "early warning signs" of stock market moves by analyzing changes in Google query volumes for search terms related to finance. Notwithstanding, the research choose as a premise that an increase in the search queries indicates an inopportune event that will decrease the stock value. However, such an assumption might not be valid in every occurrence.

Other researchers use technical analysis as a solution to predicting the future price of stocks. Paiva *et al.* [23] proposes a decision-making model for day trading investments using a fusion approach of a support vector machines (SVM) classifier, as well as the Markowitz's mean-variance method (MV), for portfolio selection. For this, the proposed model was divided into two stages: (i) SVM selects the stocks with the potential to gain expected return; and (ii) MV model is used to define the proportion of the resource that goes for each portfolio. However, even though MV has been widely recognized as one of the cornerstones of modern portfolio theory, the model has drawn many criticisms and proposals of alternative or more refined models [3]. In Chandrinos *et al.* [4], a risk management system using machine learning was proposed, named ARIMS. The main target of the AIRMS is to improve the performance of two portfolios, preventing them from losses. In the study, they use two models: artificial neural networks (ANN) and decision trees. The experiments were applied to the five major currency pairs from the years 2010 to 2016. It was observed the AIRMS with a decision tree, and AIRMS with a neural network succeed in increasing the total return from portfolios, showing evidence that AIRMS can turn losing years into profitable ones improving the negative results. However, it should be noted that the AIRMS tool was applied only on profitable portfolios, which might not represent the same behavior when applied to not as profitable or loss-making ones.

Based on the premise that a stock can move following the trend of other assets with the same similarity, [12] proposes a framework that incorporates the inter-connection of firms to forecast stock prices. An autoencoder was used to reduce the dimension of stock fundamental information and group the stocks in a graph. Then, a hybrid model of graph convolutional network and LSTM to stock market forecasting is proposed. On the same research front, [5] proposes using a convolutional graph based on Convolutional Neural Networks (CNN) for stock trend prediction. The objective is to consider the stock market information and individual stock information to make the prediction. Other works deal with the problem of stock prediction, exploring the discriminatory capabilities of the CNNs when dealing with Gramian Angular Fields [2] or using an LSTM [20]. However, the previous works do not explore a risk management and resource allocation mechanism to decide whether the stock could compromise the investor's profitability.

Chung and Shin [6] propose a genetic algorithm (GA) to optimize an LSTM model using technical indicators from the Korean stock price index as the experimental data. For this, several window sizes and different numbers of LSTM units were applied to evaluate the GA's fitness. The metric for comparing the benchmark and the proposed model was the mean absolute error (MAE), mean absolute percentage error (MAPE), and mean squared error (MSE), showing that the GA-LSTM presents better performance than the benchmark in all error measures. Such results suggest that the appropriate hyper-parameters of the LSTM is an essential condition for better performance. However, GAs are used for optimization problems such that the quality of the solution depends on the processing time. In other words, GA-based solutions are slower than traditional methods and, therefore, can influence stock market prediction.

Despite the advances made in the stock market, there are still many challenges and problems in this area that this research addresses, differentiating from related works in the following Aspects:

1. An autonomous, rational and agent-based, investment service to deal with the proper allocation of resources in portfolio selection.
2. A methodology to improve the stock market prediction, independently of the stock market.
3. A mechanism that uses LSTM to predict the stock's behavior and uses the Deep Deterministic Policy Gradient (DDPG) to manage the financial resources of the portfolio.

Hence, this work brings novel contributions to the literature as its presented aspects are distinguished. The Aspect 2 offers an autonomous investment system, unlike most other works cited [6, 17, 22, 23] which proposes focused studies only on a specific model. It is also different from Chandrinos *et al.* [4], which offer a tool designed for risk control. Aspect 2 distinguishes itself when related to Malagrino *et al.* [17], Chung and Shin [6], and Paiva *et al.* [23] which uses only the market index, and Chandrinos *et al.* [4] that focus its studies in the currency market. Finally, Aspect 2 is found to be a difference when comparing to Malagrino *et al.* [17]; Paiva *et al.* [23]; Chandrinos *et al.* [4], which uses SVM, ANN, and Bayesian network, respectively. Such aspect is also distinguished when compared to the [7, 13], in which use agent-based techniques but do not exploit deep reinforcement learning, nor do they use reinforcement learning for resource management. For these reasons, the proposed service will be presented in the next section.

3 AURORA: autonomous rational trader

This section describes AURORA[1], a hybrid, autonomous, agent-oriented service to trade equities in the stock market. For that, AURORA takes advantage of time series from historical data of assets and fundamental analysis variables in its investment process. Through an LSTM, time series are used to predict a given asset's value. On the other hand, the fundamental variables observe the existence of possible risks in the asset. Such information is used as input to a deep reinforcement learning model to manage the investor's resources in his portfolio. As a result, AURORA was designed into three modules: (i) the insider, a predictor module that can predict the stocks' movements, described in Sect. 3.2; (ii) a risk management module to deal with market volatility through the news, presented in Sect. 3.3; and (iii) an agent module, a rational actuator that aims to allocate the resources available to the agent and act upon them, described in Sect. 3.4. The following shows an overview of AURORA, as well as the mechanisms modeled for its operation.

3.1 Overview

Figure 1 shows an overview of the service provided by AURORA and its operation. For this, AURORA has the predictor module, denominated Insider, which indicates if the studied stock will increase or decrease in value through historical data. The agent uses such prediction, along with the likelihood of possible risk, to decide whether it is more profitable to maintain the asset, buy more stocks, or liquidate it. This process is performed for each asset that the agent has, aiming not only at the highest possible profit from a single stock but also from the portfolio.

AURORA is modeled to operate with technical and fundamental databases, represented at the top of Fig. 1. Information from technical data, such as the asset's price, is used to feed the Insider Module (as seen in label B of Fig. 1, which is responsible for making its asset predictions. Furthermore, fundamental data are used as an input to a Risk Management Module (or RMM, as seen in Label C of Fig. 1), responsible for finding real-time issues with the portfolio's stocks. The Insiders and the RMM outputs are used as input to the Resource Allocation Model (ResAM), which is responsible for deciding which stocks will be bought or sold and the corresponding amount. After the ResAM makes its decisions, all that is left to the agent (Label A of Fig. 1) is to act upon it. A series of requisitions to the desired brokerage is then made following the agent's proposed actions (i.e., broker API).

It is worth noting that the stock market and its investors are full of irrational behaviors. Recent works have been studying that some of these behaviors might generate further profits. Wang *et al.* [30] shows that investors with loss aversion tend to have a better performance than traditional rational investors. However, AURORA's metric of rationality aims to increase profit and reduce risks. Therefore, if loss aversion can yield higher profits, AURORA would consider it a rational behavior. As an extension, other behaviors such as herding and regret aversion would also be considered rational to provide higher profitability.

AURORA was implemented using the Python 3 programming language in conjunction with the Pandas, Keras, Gym, and Spinning Up libraries. The Pandas were used to read and process data. The Keras was used to design the learning models with its high-level neural network features with the TensorFlow backend. The Gym was used to generate the environment in which the agent interacts during reinforcement learning. The Spinning Up to implement deep reinforcement learning algorithms. Conjointly, MongoDB, a general-purpose document-based database, was used to structure the data files. The choice of a non-relational database allows a more direct approach combined with Pandas data-frames. Furthermore, the complexity of the data usage is low, sparing the necessity of a relational model. The Insider module of AURORA is presented below.

---

[1] Available at https://github.com/EmpyreanAI/AURORA.
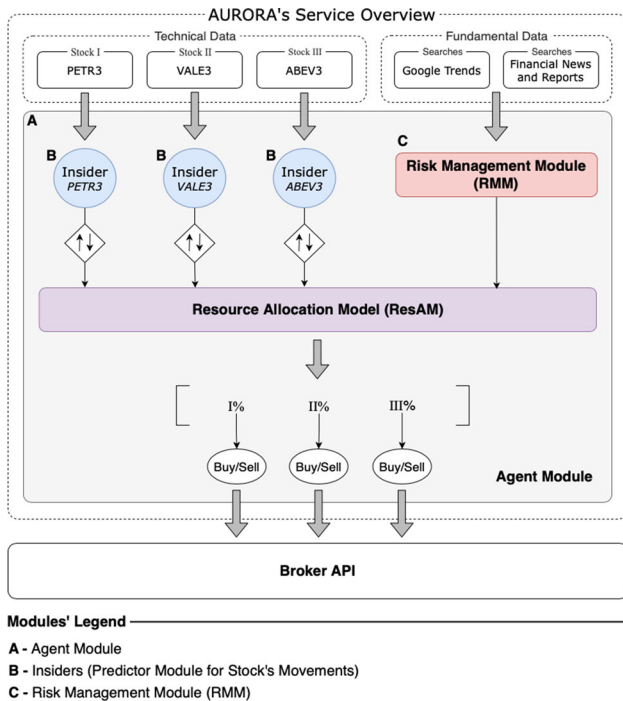
Fig. 1 AURORA's operational scenario is presenting the process flow, from data gathering to market interaction. The Figure highlights the Agent Module and its submodules, together with the Agent's ResAM. The Insider and RMM submodules and the ResAM are presented with additional details in Figs. 2, 4, and 5
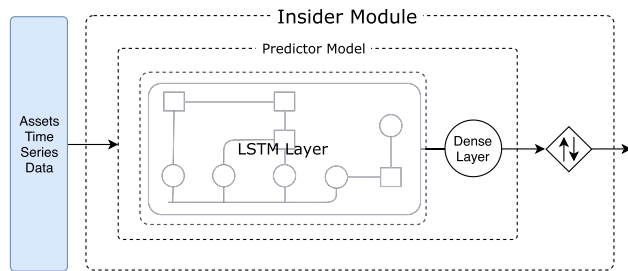
Fig. 2 Overview of Insider Module. The Predictor Model is exchangeable to any desired learning method

3.2 Insiders — predictor module for stock's movements

One of the main AURORA's rational modules is the insider module. Its task is to support the resource allocation model, predicting the movement of a specific stock. Each Insider is an individualized entity specialized in a given stock and programmable to use any desired learning model. Insider units gather their data from the database of a stock exchange, as presented in Fig. 2. In this data, multiple information is available to be used as input, such as closing and opening price and volume. The input is passed through the learning model, and a prediction is created. The

prediction made by the Insider takes the shape of a binary value, 1 indicating that a stock is increasing in value, and 0 showing a decrease.
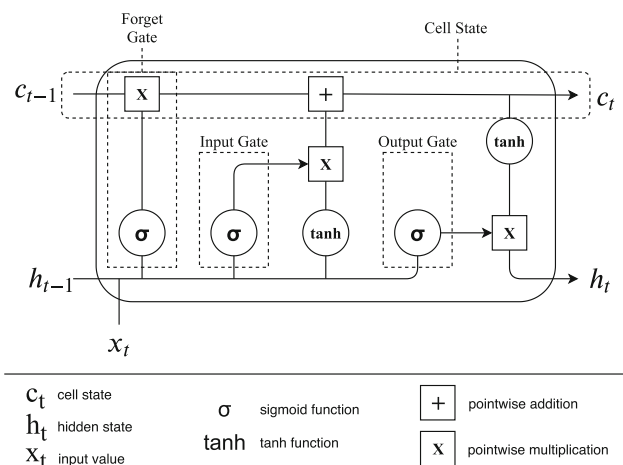
To predict the stock's movement, AURORA's insiders were modeled based on an LSTM network, a gated RNN network. Gated RNNs were modeled in our service to be based on the idea of creating paths through time that have derivatives that neither vanish nor explode [10]. With this, the LSTM can decide what information it wants to keep or forget, aiming at the improvement of the decision-making process's accuracy. The mechanism for asset prediction is presented below.

3.2.1 Mechanism for asset prediction

An LSTM network distinguishes being a gated RNN architecture that can be applied to temporal data prediction. Besides being the most effective sequence model [10], LSTM has shown substantial results by exploiting long-term inter-correlations. The usage of a gated model in AURORA overcome the RNN's difficulty in learning long time dependencies that are more than a few time-steps in length [11], addressing the stock's time series problem with a viable approach.

LSTM's cell topology adopted in the proposed service is illustrated in Fig. 3. In our service, a cell comprises a cell state c_t, which transmits information through the sequence chain, hence implying in reducing the effects of short-term memory; and gates, internal mechanisms that regulate the flow of information.

Gates mechanisms (see again, Fig. 3) allow information to be added or removed to the cell state through the learning process via *sigmoid* functions. This type of mathematical function has a domain of all real numbers, with return value increasing from 0 to 1 or −1 to 1. Thus, the activation function adopted in the modeled LSTM was

c_t	cell state	σ	sigmoid function	+	pointwise addition
h_t	hidden state	tanh	tanh function	x	pointwise multiplication
x_t	input value				

Fig. 3 An LSTM cell

the logistic function: $g(z) = \frac{1}{1+e^{-z}}$, where z is the real number to be scaled and e the Euler's number.

The LSTM cell implemented in our service has three main gates with specific tasks, as seen in Fig. 3. The forget gate is the one that decides the permanency of information. Values from the previous state c_{t-1} and current input x_t are passed to the *sigmoid* function, generating a forget value. The input gate is responsible for updating the cell state with information from the current input x_t and the previous state c_{t-1}. Those values are passed to a *sigmoid* function that decides which ones to be updated, and further being passed to a *tahn* function to help regulate the network. The output from the *sigmoid* decides which information to keep from the *tahn* by a multiplication operation. Lastly, the output gate decides what the next hidden state h_t should be. The hidden state contains information about previous inputs and is also used for prediction. Its value is calculated passing the previous hidden state h_{t-1} and current input x_t into a *sigmoid* function giving a value out_t, then it is calculated the current cell state c_t into a *tahn* function which is later multiplied to out_t to decide which information the new hidden state h_t should carry.

After treating the new input x_t in gates, the cell state value is updated. Its value is obtained by first multiplying the previous cell state c_{t-1} by the forgetting factor, possibly dropping unnecessary values, then making a pointwise addition to the resulting value of the input gate, updating the cell state c_{t-1} to a new value c_t. The new cell state c_t and hidden state h_t is then carried over to the next time-step until all time-steps are completed and the model learned.

AURORA implements an LSTM which receives as an input the historical data of a given asset. The input x_t in an LSTM cell is a three-dimensional array with the format $x_t = \langle\text{batch size, time-steps, features}\rangle$. The batch size defines the number of samples that will be propagated through the network; the time-steps represent how long in time each of the samples is; and finally, the features are the number of dimensions fed at each time-steps, in which each dimension is a learning characteristic. AURORA represents each time-step as a day, e.g., if the input shape consists of 12 time-steps with a 32 batch size and a single feature being the closing price, the input would be 32 samples of the closing price of each of the 12 days.

AURORA's service uses the LSTM considering multiples units. Doing so, the LSTM consists of n independent copies of itself. Each copy will have an identical structure but will be initialized with different weights, therefore computing differently. By using n units, the LSTM layer will produce n outputs. For that reason, a densely-connected neural network layer, or dense layer for simplicity, was added to the model. Finally, the network calculates the Dense layer and outputs its result.

However, the time-series analysis by itself might not be sufficient to predict some anomalies, given the complexity of the market and its real-time volatility to external events. To overcome this issue, the agent is provided with a risk management module to detect market's disturbances, which will be described in the next section.

3.3 Risk management module

The use of time series in the stock market can lead to promising information in predictions [11], but it is not sufficient to generate robust models. To address market volatility about external influences, the RMM was modeled into AURORA. The RMM's goal is to make a fundamental analysis using multiple sources of information to reflect an asset's present situation in the market.

Combining sizable behavioral data sets, for instance, search query volumes on Google Trends, may open up new insights into different stages of decision-making [24]. For this, in this research, the Preis *et al.* [24] approach was adopted. This approach uses Google Trends with the premise that an increase in search queries is related to a risk on the stock. Its results showed that query volumes are able to anticipate peaks of trading by one day or more. With this base, AURORA uses an asset's company name's search volume as input to ResAM.

The inputs are used to feed the AURORA's mechanism that produces the probability of risk, as presented in Fig. 4. Given the inputs, the mechanism generates a vector of risk probability, being each position in the vector the probability of a specific asset being a risk. In this research, RMM is modeled using a threshold function, where values above the determined threshold would indicate a possible risk. For this purpose, AURORA uses the volume of queries on the name of an asset on Google Trends. The data used from Google Trends is provided in weekly values of interest for the desired term. Such value comes as an integer value from 0 to 100, where 0 represents no interest and 100 represents extreme interest. When the research interest is below a threshold, the mechanism does not pose a risk to the asset. On the other hand, when the interest passes the
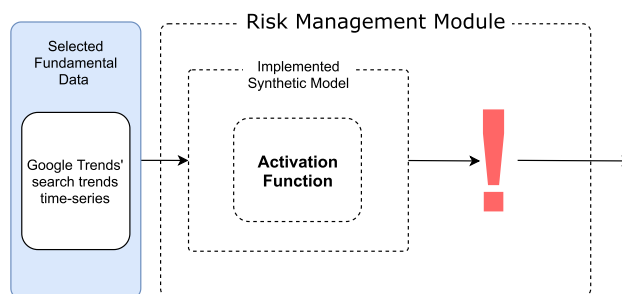

Fig. 4 Risk management module

threshold, a risk is indicated. This mechanism's modeling is based on the premise that any increase in research indicates an asset risk. Finally, RMM's output (the probabilities of risks) and the Insiders' predictions are carried on as inputs to the agent's Resource Allocation Model (ResAM), which is now responsible for making the most profitable decisions, which will be described in the next section.

3.4 Agent module and resource allocation model

AURORA's service has an agent-based structure in which the agent can be seen as a trader, which aims to maximize its profit and reduce risks, making decisions in its user's portfolio. In this article, an agent is understood as a computational entity capable of perceiving the environment through sensors and acting autonomously upon it through actuators [25]. AURORA core is based on an agent that senses the environment (the desired stock market) through its assets' information. It acts on the market rationally based on the output of its allocation model.

The agent has a wallet, where its unallocated money is stored. Also, there is a portfolio, a set of user-chosen assets in which the agent invests the available money in its wallet. An agent considered rational is the one who seeks the maximum value in its portfolio and the minimum amount of money in the wallet. In other words, the maximum investment profit.

Decisions made by the agent are supported by its Resource Allocation Model (ResAM). The module receives as input information of the direction of the stocks provided by its Insiders and information from the RMM, which decides whether an asset is a liability that could place the agent's profit in jeopardy. The output is then given as a vector of percentages, each element the percentage of the asset that should be sold or bought. The mechanism for ResAM is presented below.

3.4.1 Mechanism for resource allocation

AURORA's ResAM was modeled based on a deep reinforcement learning model, denominated Deep Deterministic Policy Gradient (DDPG). DDPG was chosen because it is inherent to this type of problem, given that the apprentice entity does not have any knowledge about the environment and needs to discover by itself through exploration and rewards [28]. In DDPG, the entity will need to learn how to map the market's state with respective responsive actions, aiming to increase its profits and reducing risks. Therefore, in the beginning, the entity will have no information about the environment and will not know which actions to perform, thus needing to explore the cost of its activity as a mean of learning [25].

With this in mind, The DDPG was implemented using OpenAI's deep reinforcement learning library, Spinning Up[2]. The algorithm was also chosen for the approach for being model-free, off-policy, and actor-critic, which uses deep functions approximation to learn policies in a high-dimensional action space [16]. The DDPG is a development of the deterministic policy gradient algorithm [27], combined with successful ideas of the Deep Q-Network [19].

As it is not possible to apply Q-Learning directly to continuous action-space [31], the DDPG appeal for an implementation based on the deterministic policy gradient algorithm [27]. Therefore, the DDPG is an actor-critic algorithm, which interchanges in learning an action-value function Q and a policy [16], and it does so using deep neural networks for those function approximations [16]. The actor uses off-policy data and a Bellman equation to learn the Q function, while the critic uses the Q function learned to develop it's policy [16].

Additionally, for the DDPG to function properly in ResAM, intrinsic characteristics of reinforcement learning algorithms need to be defined: state space, action space, and rewards. For this, the entity takes an action a_t defined by the action space and will receive as feedback from the environment an state s_{t+1} and a reward r_{t+1} [28]. Figure 5 shows the interaction between the entity with the DDPG algorithm and the learning environment. Such an environment is also proposed as a part of the AURORA's learning mechanism. Its development was made using the OpenAI's Gym library[3], and its environment is presented below.

3.4.2 B3 environment

To interact with the DDPG algorithm, it was necessary to develop an environment simulating the B3 stock market. This environment is designed for training and testing the DDPG model in the AURORA's agent. As said, the environment defines the state space, action space, and the reward function. The following definitions were used:

– The States were developed envisioning the agent variables and the output results from the Insiders and the RMM. Therefore, space contains the quantity of deallocated money an agent has and how much its portfolio is yielding. Additionally, the space also has the price, prediction, profitability, amount in the portfolio, and risk, for each of the desired invested assets.
– The Action Space is represented as $A_t = [a_{0,t}, \ldots, a_{n,t}]$, where each $a_{i,t}$ represents the action that the agent should make for the asset i in the time t. The value of an

---

[2] Available at: https://spinningup.openai.com.
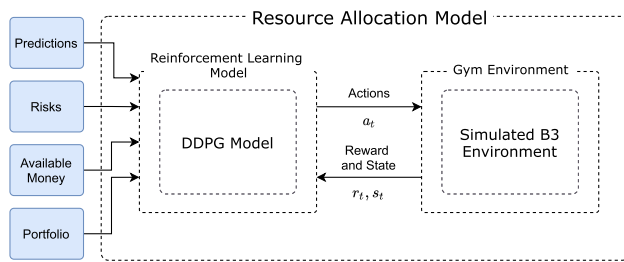
[3] Available at: https://gym.openai.com.

Fig. 5 Overview — resource allocation model (ResAM)

action $a_{i,t}$ lies between $[-1,1]$, where: when between $[-1,0)$ the value represents the percentage of the available money which will be used to buy the asset; when 0 the agent waits for that asset; when between $(0, 1]$ the agent sells all of his positions for that asset.

– The Reward is attributed for each of the possible actions the agent can make for each of his assets. Therefore, the following actions are possible, and the respective reward was attributed:

  – **Profitable Sale:** For each time-step t, if a profitable sale happened, this reward is assigned as the difference of the time t price with the bought price;
  – **Profitable Wait:** For each time-step t, if holding an asset has resulted in a profit, this reward is assigned as the difference of the portfolio price at time t with its initial investment;
  – **Buy Action:** Constant negative reward, indicating that money is being spent;
  – **Unavailable or Loss Action:** Increasing negative reward attributed every time the agent make an unprofitable investment or an unavailable action. This reward is negatively increased based on the time-step, the further on the simulation higher the penalty;
  – **End of Episode:** Difference of the portfolio's final income with the initial investment amount.

Adding all given rewards, the final reward for a time-step t is defined. For each asset in the agent's portfolio, the respective reward will be added. After all the asset's rewards have been added, the final reward is divided by a constant value, avoiding overly scaled rewards and helping the model learn. In the next section, we present a performance evaluation to validate the AURORA.

4 Performance evaluation and methodology

This section presents the methodology adopted and the results obtained to validate the AURORA. For this, AURORA was validated into three steps: (i) performance evaluation to uncover the best set of AURORA's

hyperparameters; (ii) performance evaluation to predict the stock's movement to validate the implantation of AURORA in a real system; and (iii) performance evaluation of AURORA in resource allocation to manage a portfolio with one or more assets in a stock market. It is worth noting that AURORA was compared with other benchmarking solutions in the literature.

4.1 Methodology

In order to evaluate the AURORA, quantitative and qualitative assessments were performed. Each proposed solution has its particularities. Therefore, the performance evaluation for each solution becomes singular for each system [14]. With that in mind, the dataset used and how the data was preprocessed will be presented. Then, the modeled scenario and the chosen metrics will be described.

4.1.1 Dataset

The dataset used to feed the service of AURORA was obtained manually from the B3 official website[4]. The website provides daily information from assets since 1986, such as company name and code, stock code, market type, pricing (previous, open, minimum, average, maximum, closed), number of trades, and volume traded with paper, among many other available data. The data preprocessing used in AURORA will be presented below.

4.1.2 Data preprocessing

AURORA's model uses a semester time window for its training, validation, and testing. The following assets were used to compose the agent's portfolio: VALE3, PETR3, and ABEV3. The database's closing price was chosen as it reflects the day's movements in the market. The database was then filtered with the closing price in the first six months of 2014 for each of the assets in the portfolio, and its values were normalized. The normalization process was made with a Min-Max approach in a [0, 1] interval. Due to the outliers in the stock market, a mean value was inserted between two consecutive historical data. In this way, the amount of data can be increased without losing the feature's behavior.

To complete the database, a label indicating the market's correct direction was created. The labels were created using a discrete approach for each asset at each step of the time. Such an approach was chosen because predicting whether an asset will gain or lose value is a task with a higher probability of success than the real value prediction. Therefore, the labels are determined as 0 if the day's

---
[4] Available at http://www.b3.com.br/.

closing price was less than the average of the window of previous days, or as 1 otherwise.

4.1.3 Experimental setups

Hyperopt[5] library was used to select the best hyperparameters in the experiments. Hyperopt is a python library that implements Sequential Model-Based Optimization (SMBO), also known as Bayesian optimization. SMBO is applicable where the minimization of the value for some $f(x)$ function leads to a high cost due to the complex evaluation process. Such a method uses a search algorithm to determine hyper-parameters through search interactions denominated trials.

The usage of Hyperopt library is based on defining three main characteristics: (i) an objective function value to be minimized with the optimization; (ii) a search algorithm used to find the optimization set; and (iii) the search space for the selected algorithm. The selection of the characteristics is then used to conduct the desired experiments to find the best hyperparameters.

To generate and evaluate the models in AURORA, the database was divided into training, validation, and testing. Training and validation correspond to 70% of the six months contained in the database. The other 30% was used for the test phase. To assess the model's generalizability, the Nested K-Fold was used. Nested K-Fold consists of dividing the total data into k subsets. Thus, the training and testing phase is performed k times. Each time the k is increased, the training base is increased and trained in the next subset. In this research, cross-validation was performed with $k = 6$. To analyze the experiments, the following metrics were used: (i) sensitivity; (ii) precision; (iii) specificity; (iv) accuracy; and (v) F1 Score. The experiments were conducted using a Linux virtual machine hosted in Google Cloud. The machines were provided with 4 CPUs and 3.8 GB of main memory.

4.2 Performance evaluation of AURORA's hyperparameters

To uncover the best hyper-parameters to the LSTM of AURORA, a Hyperopt experiment was conducted. The performance of this experiment requires the definition of Hyperopt's operating characteristics. For this, the following characteristics were used: (i) the objective function was the F1-Score value returned from the LSTM model; (ii) the search algorithm was the Tree Parzen Estimator - TPE; and (iii) the search space is presented in Table 1. The experiments were executed with 1000 trials for each asset in the

portfolio (i.e., PETR3, VALE3, and ABEV3). In the training process, a log was generated to enable further analysis of the TPE parameter choices. The best set of parameters for each stock is presented in Table 2.

Figures 6 and 7 show the behavior of each stock based on the parameters in Table 2. Based on the results, we noted a need to develop specific models for each asset. This is because the models analyze the time series patterns and each asset's hyperparameters are different. Another essential aspect being mentioned is presented at Fig. 6, representing the histograms from each asset experiment. For the case of PETR4 and ABEV3, it was noted that the most selected parameters do not necessarily correspond to the best parameters chosen by Hyperopt, as shown in Table 2. This occurs due to the hyper-parameter selection algorithm seeks to prioritize the exploration of values that have obtained worse results instead of taking advantage of the best results found. These results indicate that the most selected parameters during the search will not necessarily be part of the best parameters at the end of the experiment (Fig. 7). Furthermore, it is noted that the results obtained showed an increase in the slope in the objective function independent of the asset. Therefore, observing the F1 Score, it is pointed out that the models converged for better results.

4.3 Performance evaluation to predict the stock's movement

This section assesses AURORA's performance concerning its ability to predict the movement of a stock. The objective is to determine if it is feasible to model an LSTM in a decision system like AURORA. For that, each stock was trained using the hyper parameterization process's best settings, as shown in Table 2. The experiment consisted of K-fold cross-validation, with $k = 6$, as shown in Sect. 4.1.3. Figure 8 presents the loss value behavior in 5000 epochs of training. In the top-right, the box-plot shows the dispersion of the cross-validation accuracies for each time-step size.

Analyzing Fig. 8, it can be noticed that over time, there is a reduction in the number of errors and an increase in the accuracy of the models, regardless of the stock. Such models converged and stabilized before completing 500

Table 1 Values used as input for the hyper-parameters' exploration

Batch size	1	2	32	64	128	256
LSTM units	1	50	80	100	150	200
Time-steps	1	3	6	9	12	–
Optmizer	Adam	SGD	RMSprop	–	–	–

Table 2 Best parameters found by Hyperopt for each stock

	VALE3	PETR3	ABEV3
Batch Size	2	2	128
LSTM units	200	80	1
Time-steps	6	9	6
Optimizer	RMSprop	RMSprop	Adam
Best F1 Score	0.78781	0.83006	0.65109

training seasons. These findings indicate the ability to generalize models to predict the movement of a given stock without overfitting. Also, the uniform convergence of the loss can confirm these results. Therefore, such results showed that it is possible to model an LSTM in AURORA with a high accuracy rate and a stable prediction of the stock's movement due to the boxplot's interquartile amplitude.

After such observations, we analyzed how AURORA's LSTM model behaved in a real market situation, as presented in Fig. 9. The goal is to understand the behavior of the predictions over the days. As can be seen in the results, the accuracy was satisfactory for all evaluated stocks. In the worst case, Fig. 9a, an accuracy of 80% for VALE3, was obtained. In contrast, ABEV3 (Fig. 9c) and PETR3 (Fig. 9b) achieved an accuracy of 88% and 93%, respectively. Despite the promising results, it is noted that the LSTM has a limitation predicting sudden movement direction changes. It is worth noting that such limitation can be seen as an advantage since this limitation might introduce an undesirable risk when the model is applied to a market with high volatility. Also, in the stock's continuous movements, the AURORA's modeled LSTM showed great accuracy and ensured that its previous errors did not affect its performance.
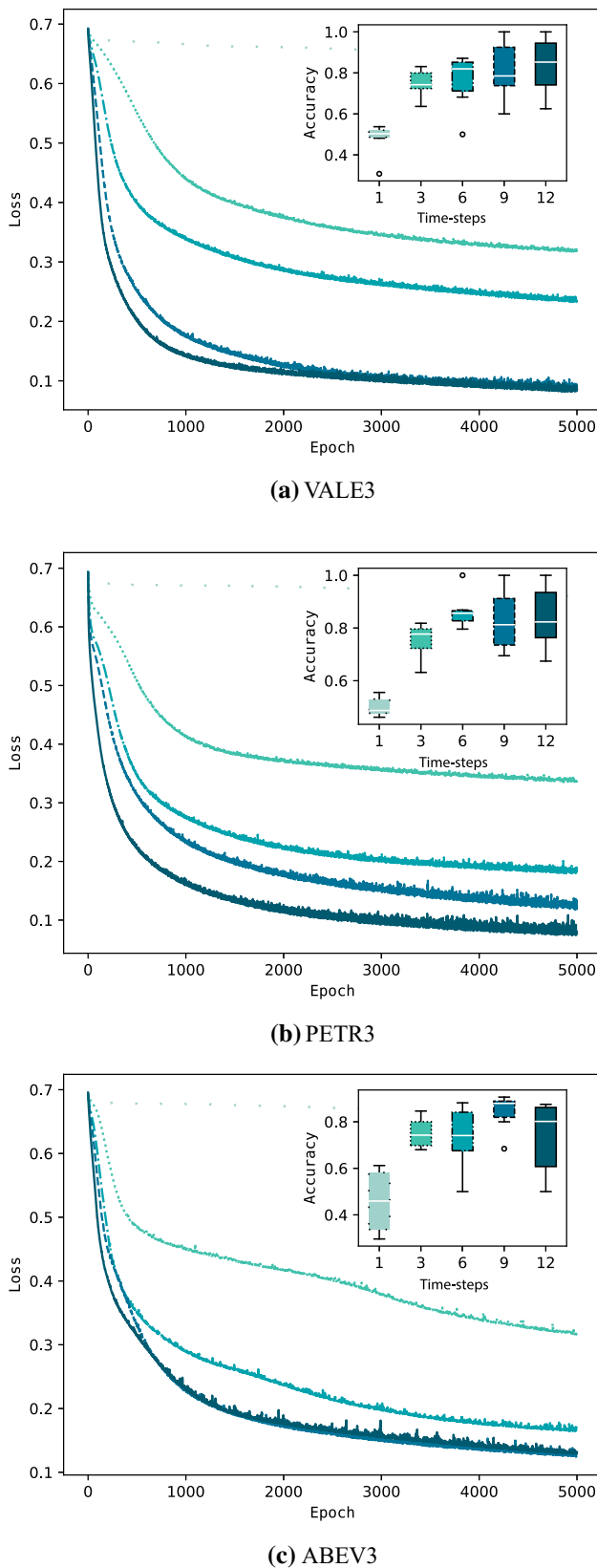

(a) VALE3

(b) PETR3

(c) ABEV3
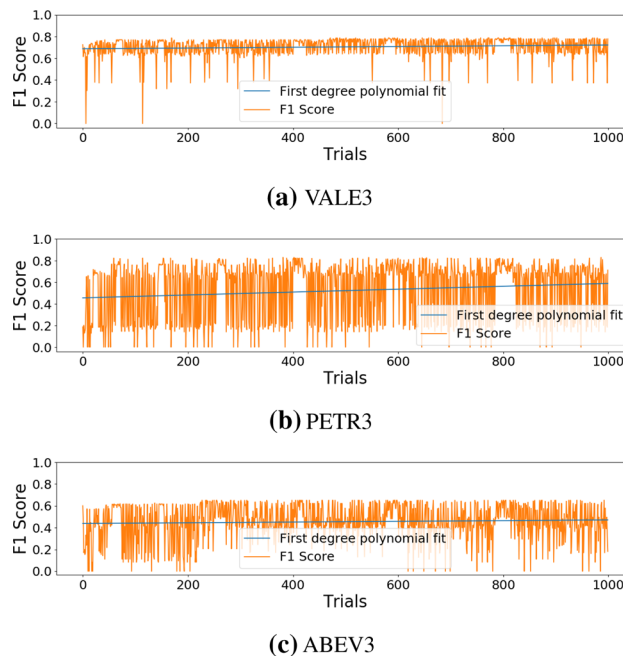
Fig. 7 The progression of the F1 Score through the 1000 trials on the Hyperopt experiment

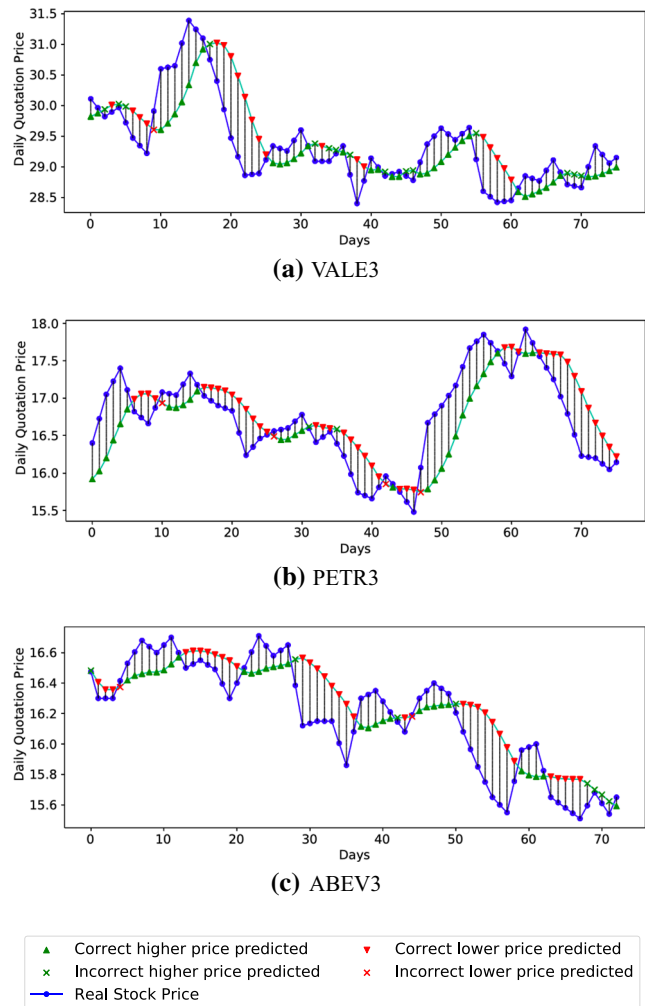### 4.3.1 Analysis of AURORA with other solutions in the literature

To validate LSTM's generalizability in AURORA, a comparative analysis was performed with the following classification algorithms: (i) KNN - K-Nearest Neighbors; (ii) SVM - Support Vector Machine; (iii) RNN - Recurrent Neural Network; and (iv) GRU - Gated Recurrent Unit. To make a fair comparison, the selected algorithms were subjected to an experiment similar to the one performed for the calibrated LSTM, as presented in Sect. 4.2. For that, SVM was parameterized with a radial basis function kernel, thus avoiding overfitting situations. SVM was also

(a) VALE3   **(b)** PETR3   **(c)** ABEV3
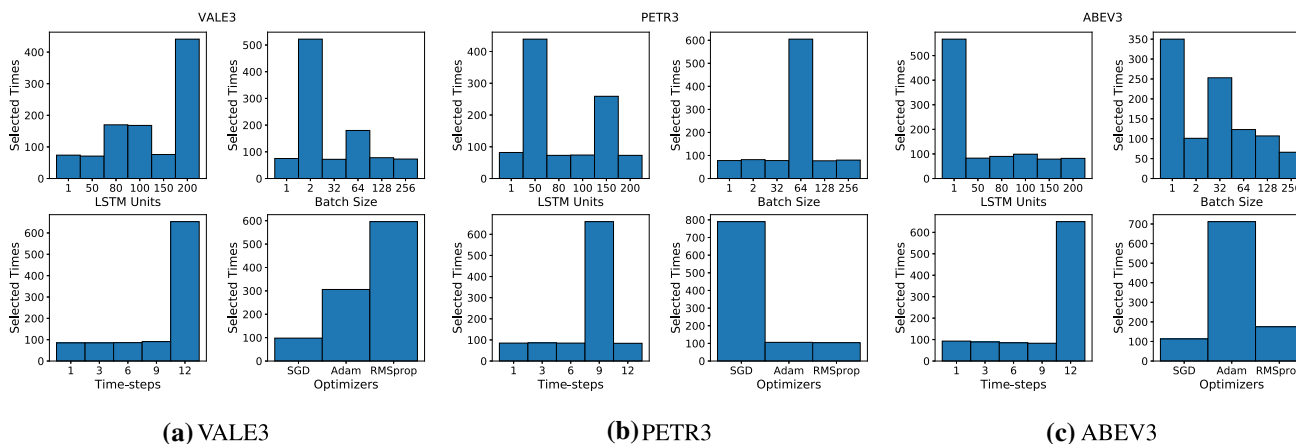
Fig. 6 Histograms from Hyperopt experiment. Presents the most selected parameters during the 1000 trials

(a) VALE3

(b) PETR3

(c) ABEV3

Fig. 8 Performance evaluation of decision-making process for multiple time-steps

(a) VALE3

(b) PETR3

(c) ABEV3

▲ Correct higher price predicted   ▼ Correct lower price predicted
× Incorrect higher price predicted   × Incorrect lower price predicted
● Real Stock Price

Fig. 9 Predictions made by AURORA's LSTM model for each stock. The cyan line represents the average price in certain time-steps. The blue line is the closing price of the stock on that day. The markers represent the correctness of the predictions

calibrated regarding its gamma parameter, which defines the extent to which an only training example impacts the learning process. In the case of KNN, the experiments were carried out with a k value set to 1. To adapt it to the time series problem, the model was parameterized with a Dynamic Time Warping (DTW) distance metric. For deep neural networks (i.e., RNN and GRU), a time window of size 6 was used. The networks were all trained with one unit for a total of 5000 epochs.

To analyze the experiments, the following metrics were used: (i) accuracy represents how close the models are to the correct value, Equation 1; (ii) precision represents how correct the positive predictions of the models were classified as correct, excluding false negatives, Equation 2; and (iii) sensitivity represents how correct the positive predictions of the models were rated correct, excluding false positives, Equation 3. The results could be in one of the

Table 3 Comparative Analysis of AURORA's LSTM model with other algorithms

Algorithms	Accuracy	Precision	Sensitivity
SVM	55.17 ± 2.1	47.73 ± 1.9	87.50 ± 2.3
KNN	55.17 ± 1.8	47.73 ± 1.5	87.50 ± 1.9
RNN	67.74 ± 1.2	87.50 ± 1.4	63.64 ± 1.3
GRU	41.94 ± 2.4	83.33 ± 2.2	22.73 ± 2.6
AURORA's LSTM	84.92 ± 1.5	82.86 ± 1.4	89.23 ± 1.1

following groups of prediction: (i) correct positive prediction, True Positives - TP; (ii) correct negative prediction, True Negatives - TN; (iii) incorrect positive prediction, False Positives - FP; and (iv) incorrect negative prediction, False Negatives - FN.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \qquad (1)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad (2)$$

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \qquad (3)$$

Table 3 shows the results obtained from the evaluated algorithms. SVM[6] and KNN[7] obtained equal results, with an accuracy of 55%, well below AURORA's LSTM. It is believed that such behavior may be the result of an unbalanced amount of ups and downs in the assets' time series. Regarding the neural network results, it can be observed that the LSTM has superior results, having in the worst case a 25.36% gain for the accuracy metric. For the precision metric, the GRU and LSTM are statistically equivalent. However, LSTM outperforms other metrics with a gain of 2 times in worst case for accuracy metric. Therefore, it is observed that AURORA's LSTM has stability in the results compared to other solutions and obtaining a high hit rate.

The resulting predictions for the KNN, SVM, GRU, and RNN are presented in Fig. 10. The results obtained by KNN (Fig. 10a) and SVM (Fig. 10b) show the limitations of the models to predict during an alternating variation in price. This confirms the results presented in Table 3. Although AURORA's LSTM has this limitation, the predictions in the same circumstances are quickly adaptable. As a result, AURORA's LSTM makes fewer mistakes in transitions, as presented in Fig. 9. It is worth noting that the KNN and SVM make mistakes predicting prices that follow the same direction in time, unlike AURORA's LSTM. As observed from GRU (Fig. 10c) and RNN (Fig. 10d),

the results obtained show an issue in the predictions related to the average trend. When the average trend starts to change in direction, it is coherent that the model understands that the next day's prediction will also change direction. However, both the GRU and the RNN fail to predict so. Such an issue is not present in AURORA's LSTM that presents consistency in predictions.

4.4 Performance evaluation of AURORA in resource allocation

This subsection presents AURORA's efficiency in resource allocation. The objective is to prove the learning of AURORA to manage a portfolio with one or more assets in a stock market, in addition to measuring its profitability. Therefore, AURORA was compared with the following strategies: (i) Savings Profitability[8], (ii) Interbank Deposit Certificate[9] (CDI), and (iii) buy and hold[10]. Table 4 presents the set of parameters established to carry out the experiments. It is worth mentioning that to avoid propagating errors in the model, the correct prediction labels were provided by AURORA. To measure the efficiency of AURORA, the following performance measures were used: (i) profit; (ii) profitability; (iii) return; and (iv) yield. The experiments were carried out increasingly in complexity, first carrying out individual experiments of the assets, and ending with the complete portfolio experiment, as shown below.

4.4.1 Impact of individual asset allocation

To analyze how the DDPG would behave in a simpler environment of AURORA, a study was carried out using individual assets. For this, each asset's study was carried out (i.e., PETR3, VALE3, and ABEV3), considering an initial investment of R\$ 10, 000.00. During the experiments, it was observed that even with profits on assets, the return remained mostly negative, as shown in Fig. 11. Such behavior reflects the formulation of the reward function that aims to maximize the reward's accumulated value over a trajectory. The highest final return amount was 82, 272 for PETR3, while ABEV3's −68.70 was the lowest. Another critical aspect, observed in the result, is related to the agent who, instead of switching between purchases, sales and waits, perfor-med a specific action[11] Despite this

---

[6] Parameterized with $C = 0.1$ and $\gamma = 1$.

[7] Parameterized using the DTW metric.

[8] Values from: http://www.yahii.com.br/poupanca.html.

[9] Values from: http://www.yahii.com.br/cetip13a21.html.

[10] Strategy used by investors who decide to buy an asset for the long term.

[11] It is worth emphasizing that not all selected actions are performed. If a purchase action is required, and the agent does not have the amount of money to buy the asset, it will not be performed. With this, the agent receives punishment for the action performed incorrectly. The same logic applies similarly to sales.

behavior, AURORA managed to deliver profitable results with profits of R\$ 1, 377.00 and R\$ 603.00 for PETR3 and VALE3 assets, respectively, as shown in Table 5.

Table 5 presents the comparative profits, profitability, and yields about savings, CDI, and buy and hold for each asset (i.e., PETR3, VALE3, and ABEV3). It is observed that the models of PETR3 and VALE3 obtained the best results with a return of 13.77% and 6.03% respectively. Such returns are more significant than savings, CDI, and buy and hold strategy, reaching more than double these investments for PETR3's assets. On the other hand, the ABEV3 did not present any profit type, consequently, having no profitability. Therefore, it is possible to infer that, even with the asset devaluing 9.30% in the analyzed period, the agent's strategy does not generate losses to the investor with the negotiation of the asset.

4.4.2 Impact of portfolio allocation

This subsection assesses the performance of AURORA with a diversified portfolio of assets. The portfolio was built with the assets used in Sect. 4.4.1 and with the parameters of Table 4. To increase the number of assets that the agent is able to invest, the initial investment amount was defined as R \$50, 000.00. Figure 12 shows the behavior of the average return during the experiments. Compared to individual assets, the portfolio's return was higher and offered a more significant amount of resources inserted in the system. It is also noted that there is a convergence of the return, obtaining a final value of 404.24. This makes sense since AURORA chooses a better reward with DDPG due to the system's more significant amount of resources.

To compare AURORA's profit to a buy and hold strategy, is necessary to create such comparison portfolio. When a investment with more than one asset is created, the possibility of portfolios compositions are endless. Therefore, techniques have been developed in the literature to assess the issue of creating an optimal portfolio. In this paper, the comparison portfolio created was based on the Markowitz Mean-Variance theory [18]. It states that the process of construction of an optimal portfolio, takes in consideration the quantification of the profit and risk of each individual asset with the objective of creating a frontier of the most efficient portfolios [18]. This frontier is presented with the set of portfolios that theoretically provides the higher return with the lowest risk.
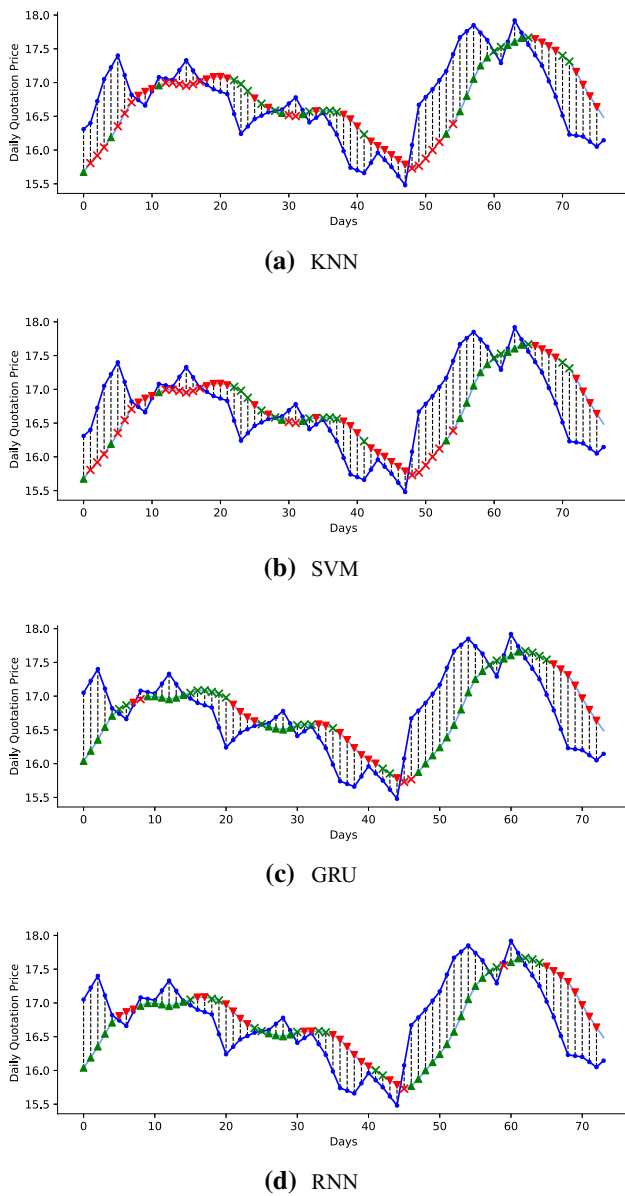
Markowitz's efficient frontier for the selected assets composition is shown in Fig. 13. To obtain such portfolios, the monthly closing prices of each asset from 2011 to 2013 were used as a basis. With the price information, monthly returns, and risks were calculated and used to compose the portfolio. The color of each portfolio combination is

related to an indicator that helps to find a balance point between risk and return, called Sharpe ratio [26]. The risk-free application chosen to calculate the ratio was the CDI return from 2013, of 8.06%. The efficient frontier was then found with a set of 25, 000 random simulations. For this comparison, we consider two portfolios, the one with the lowest risk and the one with the highest Sharpe ratio, represented by the green and red markers. The lowest risk portfolio estimates a half-yearly return of 1.18%, with a risk of 45%. On the other hand, Sharpe's highest ratio portfolio has a potential return of 4.28% with a risk of 80%.

Table 6 shows the results obtained from the AURORA portfolio, comparing it with the efficient frontier portfolios concerning profit, profitability, and yield. If the investor decided to migrate to the lower risk portfolio, the investor would obtain a profit of R\$520.00 in Petrobras but would lose R\$ 1, 914.00 with Vale and R\$1, 606.00 with Ambev. Obtaining a total yield of − R\$3000.00 of the value by investing and totaling your portfolio with R\$47, 000.00. The investor who decided to migrate to the portfolio with the highest Sharpe ratio would earn R\$104.00 in Petrobras, R\$0.00 in Vale, and would be reduced by a loss of −R\$4, 088.00 at ABEV3. The investor would lose R\$3, 984.00 of the money invested, totaling R\$46, 016.00 in the portfolio. In contrast, AURORA made a profit of R\$5, 874.00, a return of 11.74%, a return more significant than the actual and projected returns of the portfolios considered efficient. The yield obtained is 250.44% greater than savings and 141.06% more significant than the CDI. Note that the profitability achieved is slightly lower than the profitability of the PETR3 asset. This difference can be justified because Petrobras's asset is the only one that appreciates in the evaluated period, being possible that the model understands the risk of investing in other assets. Therefore, AURORA, in addition to learning to manage a portfolio, can generate a higher yield than safer fixed yield investment options.

5 Conclusion

In the development of this research, it was clear that stock markets play an essential role in the economy and offer opportunities for companies and corporations to grow and investors to make profits. Due to its complexity, volatility, and opportunities, studying its behavior is a growing research concern. However, understanding the stock exchange's intrinsic rules and taking opportunities are not trivial tasks. This work proposed AURORA, an autonomous rational trader, a hybrid, autonomous, agent-oriented service to trade equities in the desired stock market. AURORA is based on a rational agent, a computational entity capable of perceiving the market and acting upon its
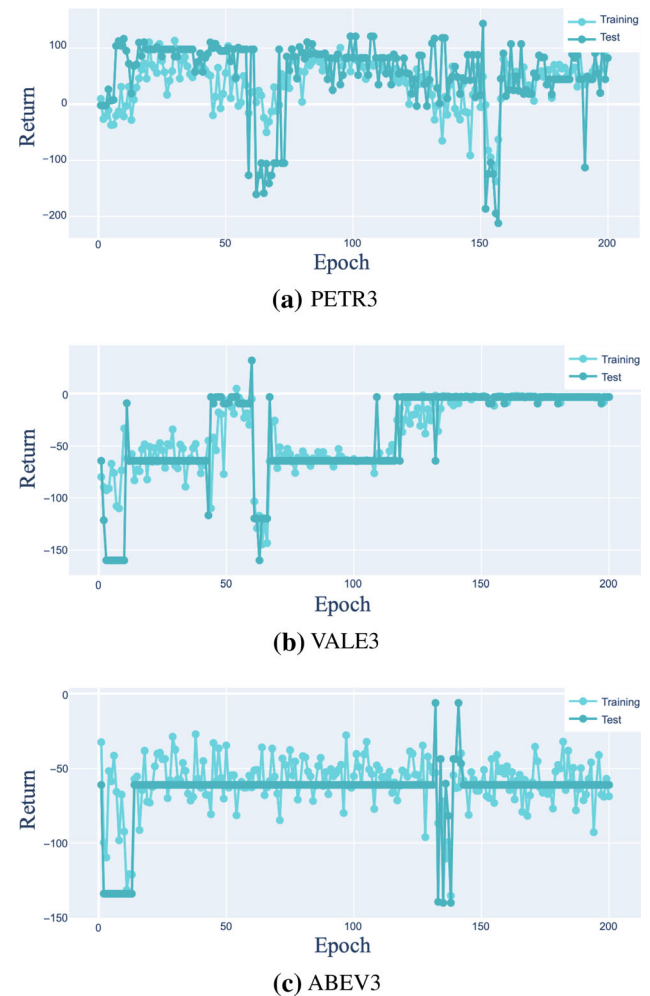
(a) KNN

(b) SVM

(c) GRU

(d) RNN

Fig. 10 Predictions made by the KNN, SVM, GRU, and RNN models

Table 4 Set of selected parameters

Parameter	Standard value
Epoch	200
Steps by epoch	1000
Policy learning rate	0.0005
Q-Value learning rate	0.0001
Buffer	500,000
Batch Size	100
Initial Steps	10
Noise	1.0
Hidden Layer	(16, 16)
Activation function - Hidden Layer	ReLU
Activation function - Output Layer	Tanh
Test episodes	10

(a) PETR3

(b) VALE3

(c) ABEV3

Fig. 11 Performance evaluation of the average return for each stock

perception autonomously through a mechanism for asset prediction and resource allocation.

An extensive evaluation made it possible to assess AURORA's efficiency through the B3 stock exchange. As proof of concept, AURORA was designed to operate in the B3 with a portfolio considering different stocks of three famous Brazilian companies: PETR3, VALE3, and ABEV3. The experimental results show that the proposed service can predict the gain or loss of value at the price of a stock with an accuracy higher than 82.86% in the worst case and 89.23% in the best case compared with other solutions in the literature. Furthermore, the proposed
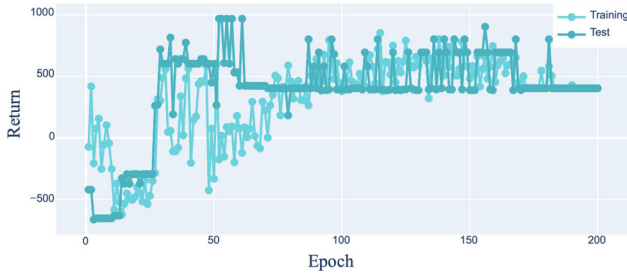
Table 5 Evaluation of AURORA about profits, profitability, and yield compared to savings, CDI, and buy and hold

		PETR3	VALE3	ABEV3
Buy-Hold	Profit	R$624.00	−R$957.00	−R$730.00
	Profitability	6.24%	−9.57%	−7.30%
AURORA	Profit	R$1377.00	R$603.00	R$0.00
	Profitability	13.77%	6.03%	0%
	Yield - Savings	311.04%	80.00%	–
	Yield - CDI	182.75%	23.81%	–

Fig. 12 Performance evaluation of the average return of portfolio
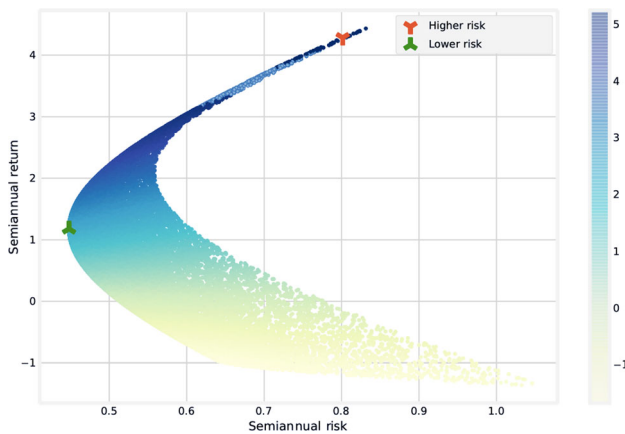
Fig. 13 Portfolio based efficient frontier

service can achieve a profitability of 11.74%, overcoming fixed-income investments, and portfolios built with the Markowitz Mean-Variance model. These results corroborate the hypothesis that AURORA is a reliable service to trade assets in the stock market with high precision and stability.

As future work, we intend to propose a space-time text collection mechanism to assist in the proposed service. Also, we intend to extend our proposal in a real case study to the derivative markets. Finally, we intend to discretize the action space to implement the Deep Q-Network algorithm and use sentiment analysis in the short text to predict the stock market.

Table 6 Evaluation of AURORA compared to the portfolios efficient frontier: Sharpe ratio - lower and higher risk

		Estimated	Real
Lower risk	Profit	R$590.00	−R$3, 000.00
	Profitability	1.18%	−6%
Higher risk	Profit	R$2, 140.00	−R$3, 984.00
	Profitability	4.28%	−7.96%
AURORA	Profit	–	R$5, 874.00
	Profitability	–	11.74%
	Yield - Savings	–	250.44%
	Yield - CDI	–	141.06%

Declarations

Conflicts of interest/Competing interests The authors (Renato A. Nobre, Khalil C. do Nascimento, Patricia A. Vargas, Alan D. B. Valejo, Gustavo Pessin, Leandro A. Villas and Geraldo P. Rocha Filho) declare that there is no conflict of interest.

Availability of data and material All used data are public available on the B3 stock market website: http://www.b3.com.br/.

Code availability Every code developed is public available on Github: https://github.com/EmpyreanAI.

References

1. Araújo R.d.A, Nedjah N, Oliveira A.L, Silvio R.d.L (2019) A deep increasing-decreasing-linear neural network for financial time series prediction. Neurocomputing 347:59–81
2. Barra S, Carta SM, Corriga A, Podda AS, Recupero DR (2020) Deep learning and time series-to-image encoding for financial forecasting. IEEE/CAA J Automat Sinica 7(3):683–692
3. Cesarone F, Scozzari A, Tardella F (2011) Portfolio selection problems in practice: a comparison between linear and quadratic optimization models. arXiv preprint arXiv:1105.3594

4. Chandrinos SK, Sakkas G, Lagaros ND (2018) Airms: a risk management tool using machine learning. Expert Syst Appl 105:34–48

5. Chen W, Jiang M, Zhang WG, Chen Z (2021) A novel graph convolutional feature based convolutional neural network for stock trend prediction. Inf Sci 556:67–94

6. Chung H, Shin KS (2018) Genetic algorithm-optimized long short-term memory network for stock market prediction. Sustainability 10(10):3765

7. Cocco L, Concas G, Marchesi M (2017) Using an artificial financial market for studying a cryptocurrency market. J Econ Inter Coord 12(2):345–365

8. Dixon M, Klabjan D, Bang JH (2017) Classification-based financial markets prediction using deep neural networks. Alg Fin 6(3–4):67–77

9. Enke D, Thawornwong S (2005) The use of data mining and neural networks for forecasting stock market returns. Expert Syst Appl 29(4):927–940

10. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT press, USA

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

12. Hou X, Wang K, Zhong C, Wei Z (2021) St-trader: a spatial-temporal deep neural network for modeling stock market movement. IEEE/CAA J Automat Sinica 8(5):1015–1024

13. Huang CY (2018) Financial trading as a game: a deep reinforcement learning approach arXiv preprint arXiv:1807.02787

14. Jain R (1990) The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling. Wiley, USA

15. Lam M (2004) Neural network techniques for financial performance prediction: integrating fundamental and technical analysis. Decision Support Syst 37(4):567–581

16. Lillicrap T.P, Hunt J.J, Pritzel A, Heess N, Erez T, Tassa Y, Silver D, Wierstra D (2015) Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971

17. Malagrino LS, Roman NT, Monteiro AM (2018) Forecasting stock market index daily direction: a bayesian network approach. Expert Syst Appl 105:11–22

18. Markowitz H (1952) Portfolio selection, journal of finance. Markowitz HM–1952 77–91

19. Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M (2013) Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602

20. Moghar A, Hamiche M (2020) Stock market prediction using lstm recurrent neural network. Procedia Comput Sci 170:1168–1173

21. Nti IK, Adekoya AF, Weyori BA (2019) A systematic review of fundamental and technical analysis of stock market predictions. Artificial Intelligence Review 1–51

22. Nti KO, Adekoya A, Weyori B (2019) Random forest based feature selection of macroeconomic variables for stock market prediction. Am J Appl Sci 16(7):200–212

23. Paiva FD, Cardoso RTN, Hanaoka GP, Duarte WM (2019) Decision-making for financial trading: A fusion approach of machine learning and portfolio selection. Expert Syst with Appl 115:635–655

24. Preis T, Moat HS, Stanley HE (2013) Quantifying trading behavior in financial markets using google trends. Sci Rep 3:1684

25. Russell SJ, Norvig P (2016) Artificial intelligence: a modern approach. Pearson Education Limited, Malaysia

26. Sharpe WF (1966) Mutual fund performance. J Bus 39(1):119–138

27. Silver D, Lever G, Heess N, Degris T, Wierstra D, Riedmiller M (2014) Deterministic policy gradient algorithms

28. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction. MIT press, USA

29. Thakkar A, Chaudhari K (2021) A comprehensive survey on deep neural networks for stock market: The need, challenges, and future directions. Expert Systems with Applications. p 114800

30. Wang J, Zhou M, Guo X, Qi L (2018) Multiperiod asset allocation considering dynamic loss aversion behavior of investors. IEEE Trans Comput Soc Syst 6(1):73–81

31. Watkins CJ, Dayan P (1992) Machine learning. Q Learn 8(3–4):279–292