ROBERTO FRAY DA SILVA

Automated stock trading system using deep reinforcement learning and price and sentiment prediction modules

São Paulo
2021

ROBERTO FRAY DA SILVA

Automated stock trading system using deep reinforcement learning and price and sentiment prediction modules

Doctoral thesis presented to Escola Politécnica da Universidade de São Paulo to obtain the degree of Doctor of Science.

São Paulo
2021

ROBERTO FRAY DA SILVA

Automated stock trading system using deep reinforcement learning and price and sentiment prediction modules

Corrected Version

Doctoral thesis presented to Escola Politécnica da Universidade de São Paulo to obtain the degree of Doctor of Science.

Concentration area:

Computer Engineering

Advisor:

Dr. Carlos Eduardo Cugnasca

São Paulo
2021

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 23 de Julho de 2021

Assinatura do autor: _____

Assinatura do orientador: _____

Catalogação-na-publicação

Silva, Roberto Fray
Automated stock trading system using deep reinforcement learning and price and sentiment prediction modules / R. F. Silva -- versão corr. -- São Paulo, 2021.
180 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Deep Learning 2.Deep Reinforcement Learning 3.Price Prediction 4.Sentiment Analysis 5.Stock Trading I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t.

ACKNOWLEDGMENTS

Many people contributed to the development of this research, and one page is not enough to thank them all.

First, I thank my advisor Prof. Carlos Eduardo Cugnasca, for all his support, encouragement, and help during this research and other projects. I also thank Prof. Hugo Tsugunobu Yoshida Yoshizaki for the opportunity to participate in different academic projects and for all the valuable pieces of advice. You helped me become a better researcher and a better person.

This research would not be possible without all the support given by Itaú Unibanco S.A. through the Itaú Scholarship Program (Programa de Bolsa Itaú - PBI) in the Data Science Center (C^2D) of the Escola Politécnica da Universidade de São Paulo. I thank all researchers, advisors, and the committees that are part of this laboratory. Being part of this laboratory was unique, and it contributed a lot to all the ideas that were implemented in this research and on the research articles written.

I thank all my coworkers and friends at the two laboratories I have spent most of the time during the PhD: C^2D and LAA. You helped me a lot in becoming a better researcher and better understanding concepts related to data science, artificial intelligence, machine learning, computer science, and research in general.

Special thanks at C^2D are due to Angel de Paula, Bruno Nishimoto, and Francisco Paiva, for discussing important topics and concepts and helping to implement the sentiment analysis code. In the case of LAA, special thanks are due to Bruna Barreira, Fernando Xavier, Gustavo Mostaço, and Tiago Marto, for all the help in reviewing the manuscript, discussing concepts, and helping to implement the code for price prediction. I also thank Fernando Hattori and Carlos Agarie from Mercúrio Digitalizações for all the help, patience, and hard work. All of you worked with me to improve myself, this research, and the projects I have participated in. You also helped me to see that there is always a solution no matter the problem.

Lastly, I thank my parents Carlos and Vera, and my brother Rodrigo for all the support, understanding, patience, and kindness. You provided me the values and the support to get to this moment, especially in difficult situations. You also taught me how to face the most challenging situations, behave, and act. I will always be grateful.

I also thank all my friends for all the patience and understanding on different situations and projects, both personal and professional. You helped me on keeping my focus, motivation, and purpose. You know who you are and that this message is for you.

Thank you all for all your support!

"It is not that we have a short space of time, but that we waste much of it. Life is long enough, and it has been given in sufficiently generous measure to allow the accomplishment of the very greatest things if the whole of it is well invested."

On the shortness of life
Lucius Annaeus Seneca

RESUMO

Sistema automático para negociação utilizando aprendizagem por reforço profundo e módulos de previsão de preços e de sentimentos

Os modelos de inteligência artificial são considerados o estado da arte em diversos domínios. Os modelos de aprendizagem por reforço profundo, uma das principais categorias de modelos de inteligência artificial, apresentam um grande potencial de aplicação em domínios que apresentam alta complexidade, não linearidade e existência de autocorrelação e de componentes sazonais, cíclicos e de ruído. Um domínio de grande relevância que apresenta estas características é o de negociação no mercado de ações. Trabalhos recentes foram realizados neste domínio utilizando aprendizagem por reforço profundo, porém sem uma integração com outros componentes relevantes como previsão de séries históricas de preços e análise de sentimentos de mercado. Uma outra lacuna importante é a falta de comparação entre modelos distintos de aprendizagem por reforço profundo em diferentes cenários de negociação de ações. O mercado de ações brasileiro é um dos 20 maiores do mundo, além de ser um importante mercado em desenvolvimento. Um problema crítico para todos os investidores nesse mercado é como melhorar as estratégias e sistemas utilizados para aumentar os retornos, considerando os riscos associados a estes. O objetivo deste trabalho foi investigar e propor um sistema para a negociação automática de ativos considerando múltiplas variáveis, previsões de séries históricas, análise de sentimentos e modelos de aprendizagem por reforço profundo. A metodologia utilizada foi a simulação do funcionamento do mercado, considerando um ativo, e a avaliação de dois cenários relevantes. Foram implementadas e avaliadas oito versões do sistema proposto, considerando seis métricas relevantes para o domínio e a estratégia de *buy-and-hold*, o principal modelo de comparação na literatura. Para o primeiro cenário, que simulou um ciclo com aumento e queda de preços, a configuração do sistema que apresentou melhores resultados utilizou o componente de previsão de preços obtido por uma rede neural recorrente com um tamanho máximo de ordem de 200 ações. Este superou o modelo de comparação. Para o segundo cenário, o qual simulou uma queda acentuada nos preços, todas as versões do sistema apresentaram melhores resultados que o modelo de comparação. A configuração utilizando uma rede neural recorrente para o componente de previsão de preços com um tamanho máximo de ordem de 10 ações demonstrou os melhores resultados. A principal contribuição desta pesquisa para a área de aprendizagem por reforço profundo foi propor um sistema que utiliza variáveis adicionais relacionadas à análise de séries temporais e análise de sentimentos, extraídas por modelos de aprendizagem profunda. A principal contribuição desta pesquisa para a negociação de ações foi propor a utilização de aprendizagem por reforço profundo considerando como entradas os preços de mercado, o volume transacionado, indicadores técnicos de mercado e as previsões de preços e de sentimentos de mercado obtidos através de modelos de aprendizagem profunda. O sistema proposto pode ser utilizado em diferentes mercados e ativos e pode ser adaptado para outros domínios.

Palavras-chave: Aprendizagem profunda. Aprendizagem por Reforço Profunda. Previsão de Preços. Análise de Sentimentos. Negociação de Ações.

ABSTRACT

The artificial intelligence models are considered state of the art in several domains. The deep reinforcement learning models, one of the main categories of artificial intelligence's models, have a high potential for being applied on domains with high complexity, nonlinearities, and the existence of autocorrelation, seasonal and cyclical components, and noise. One highly relevant domain that presents these characteristics is stock market trading. Recent works were conducted in this domain using deep reinforcement learning. Nevertheless, these did not consider integrating other relevant components such as price time series prediction and market sentiment analysis. Another critical gap is the lack of comparison of different deep reinforcement learning models in different stock trading scenarios. Besides being an important developing market, the Brazilian stock market is one of the 20 biggest markets in the world. A critical problem for all the investors in this stock market is how to improve the strategies and systems used for improving returns, considering their associated risks. This research aims to investigate and propose a system for automatic asset trading considering multiple features, time series prediction, sentiment analysis, and deep reinforcement learning models. The methodology used was a simulation of the market environment simulation, considering one asset and the evaluation of two relevant scenarios. Eight versions of the proposed system were implemented and evaluated, considering six relevant domain metrics and the buy-and-hold strategy, the main baseline model in the literature. For the first scenario, which simulated a cycle with upward and downward trends, the system's configuration that presented the best results used the price prediction component obtained from a recurrent neural network with a maximum order size of 200 stocks. It obtained better results than the baseline model. For the second scenario, which simulated a deep downward trend, all the system configurations presented better results than the baseline model. The configuration using a recurrent neural network for price prediction and a maximum order size of 10 stocks presented the best results. The main contribution of this research for the deep reinforcement learning area was the proposal of a system that uses additional time series analysis and sentiment analysis features extracted with deep learning models. The main contribution of this research for stock market trading was to propose the use of deep reinforcement learning considering as features: market prices, volume traded, technical indicators, and price and market sentiment predictions obtained using deep learning models. The proposed system can be used in different markets and assets and adapted to other sub-domains.

Keywords: Deep Learning. Deep Reinforcement Learning. Price Prediction. Sentiment analysis. Stock Trading.

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

A2C	Advantage actor-critic
ACF	Autocorrelation function
ADI	Accumulation distribution indicator
ANN	Artificial neural networks
AR	Annual returns
ARIMA	Autoregressive integrated moving average
AUC	Area under the receiver operating curve
AV	Annual volatility
B3	Brasil, Bolsa, Balcão (formerly BM&FBOVESPA)
BB	Bollinger bands
BH	Buy and hold
BiLSTM	Bidirectional long short-term memory networks
CNN	Convolutional neural networks
CR	Cumulative returns
CV	Coefficient of variation
D2-A	Labeled dataset used on Module M2 for training
D2-B	Unlabeled dataset used on Module M2 for prediction
DACT	Deep actor-critic trader
DAN2	Dynamic artificial neural networks
DDPG	Deep deterministic policy gradient
DDRL	Deep direct reinforcement learning
DJIA	Dow Jones industrial index
DL	Deep learning
DPG	Deterministic policy gradient
DQN	Deep Q network

DRL	Deep reinforcement learning
EMH	Efficient market hypothesis
ETF	Exchange-traded funds
GARCH	Autoregressive conditional heteroskedasticity
GDP	Gross domestic product
GloVe	Global Vectors for Word Representation
GPOMS	Google-profile of mood states
GRU	Gated recurrent unit
HAN	Hybrid attention network
LSTM	Long short-term memory networks
M1	Price prediction module
M2	Market sentiment prediction module
MACD	Moving average convergence divergence
MAD	Mean absolute deviation
MAE	Mean absolute error
MAPE	Mean absolute percentage error
MD	Maximum drawdown
MDP	Markov decision process
ML	Machine learning
MLP	Multilayer perceptron
MOS	Maximum order size
MSE	Mean squared error
MT	Automated trading module
MT1	Trading model considering price prediction signals
MT1A	Trading model considering price prediction signals from the E1 model (ensemble of the SVR multivariate and the SARIMAX models)
MT1B	Trading model considering price prediction signals from the SARIMAX model
MT1C	Trading model considering price prediction signals from the LSTM multivariate model
MT2	Trading model considering market sentiment signals
MT3	Trading model considering only OHLCV

MT3ta	Trading model considering OHLCV and technical indicators
MT4	Trading model considering price prediction and market sentiment signals
MT4ta	Trading model considering price prediction and market sentiment signals and technical indicators
NLP	Natural language processing
OHLCV	Open, high, low, and close prices and total volume
PACF	Partial autocorrelation function
PPO	Proximal policy optimization
R2	Coefficient of determination
REIT	Real estate investment trusts
RL	Reinforcement learning
RMSE	Root mean squared error
RNN	Recurrent neural network
RRL	Recurrent reinforcement learning
RSI	Relative strength index
SAC	Soft actor-critic
SARIMA	Seasonal autoregressive integrated moving average
SARIMAX	Seasonal autoregressive integrated moving average with external factors
SR	Sharpe ratio
ST	Stability
SVM	Support vector machine
SVR	Support vector regression
TD3	Twin delayed DDPG
TF-IDF	Term frequency-inverse document frequency
TI	Technical indicators
TRPO	Trust-region policy optimization
WR	Williams %R

LIST OF SYMBOLS

S	State space
t	Timestep
p	Autoregressive component of the ARIMA, SARIMA, or SARIMAX models
d	Differentiation component of the ARIMA, SARIMA, or SARIMAX models
q	Moving average component of the ARIMA, SARIMA, or SARIMAX models
S_t	State at time t
S_{t+1}	State at time $t + 1$
S'	State at time $t + 1$
A	Action space
A_t	Action at time t
W	Observation space
w_t	Observation at time t
w_{t+1}	Observation at time $t + 1$
r	Reward function
r_t	Reward at time t
r_{t+1}	Reward at time $t + 1$
T	Number of interactions or timesteps
P	Transition probabilities between the states
π	Policy adopted by the agent
π^*	Optimal policy
γ	Discount factor
No	Number of shares owned
C	Closing price
B	Balance (amount of money available that was not spent buying stocks)

CONTENTS

1 INTRODUCTION

This chapter explores the motivations for this work and describes its research questions, objective, and background. It is divided into the following sections: 1.1 contains relevant background information; 1.2 describes the research questions and the objective of this work; 1.3 describes the main motivations for this study; 1.4 describes the scope of this research; 1.5 describes the main assumptions used for proposing, designing, and testing the automated trading system; and 1.6 describes the structure of this document.

1.1 Background

According to the World Bank (WORLD BANK, 2021), Brazil's gross domestic product (GDP) was among the world's ten highest in 2019. Its stock market is still considered a developing market, even though several important aspects (such as market regulation and supervision to detect and avoid frauds) have been implemented in the last decades.

There has been a recent increase in the number of investors in this stock market, named B3 (Brasil, Bolsa, Balcão, formerly BM&FBOVESPA), reaching around 3.5 million investors in 2021 (B3, 2021a). In March 2021, around 9% of the buying and selling was done by individual investors, while around 11.5% was done by institutions (B3, 2021a), showing that these are important players in the Brazilian stock market.

The main stock market index, Ibovespa, was created in 1968 represents a portfolio of the most important companies traded at this stock market in terms of market capitalization and volume traded (B3, 2021b). Even though these change over time, the index tends to be formed mainly by banks, commodities, utilities, oil and gas, among others.

The investment on exchange-traded funds (ETF) is an important strategy for individual investors, as it: (i) is passive, so the investor does not need to constantly rebalance the portfolio; (ii) allows for diversification; (iii) is cheaper than investing in mutual funds; and (iv) has high liquidity (JOHNMAN; VANSTONE; GEPP, 2018).

To develop a trading strategy, an investor must choose among several methodologies, techniques, and tools. These can be grouped into three main investment philosophies: (i) fundamental analysis, which is related to using macro and microeconomic data and indicators to identify market opportunities (HU et al., 2018; NASSIRTOUSSI et al., 2014); (ii) technical analysis, which is related to using price and volume trends, as well as several other tools related to trading momentum, volatility, and returns to evaluate which stocks to buy and when to buy them (HU et al., 2018; NASSIRTOUSSI et al., 2014; OLIVEIRA; NOBRE; ZÁRATE, 2013; PERSIO; HONCHAR, 2016); and (iii) algorithmic trading, which can be defined as trading by using computer programs (NASSIRTOUSSI et al., 2014; WU et al., 2019; TRELEAVEN; GALAS; LALCHAND, 2013).

In recent years, algorithmic trading has become synonymous with applying machine learning (ML) and deep learning (DL) techniques to predict prices and trends and use them for decision-making. It is also possible to use reinforcement learning (RL) and deep reinforcement learning (DRL) models to automate trading. Two in-depth reviews on RL trading by Fischer (2018) and Meng and Khushi (2019) have shown that few works in the literature explore DRL agents' use for trading. Most of these works focus on developed markets, which have different characteristics and dynamics than developing stock markets.

A very relevant problem for all investors in the stock market is how to improve their trading strategies and systems? This could be accomplished by several different approaches, such as: (i) considering new features, which may improve prediction quality or trading results, such as in the works by Kara, Boyacioglu and Baykan (2011), Weng et al. (2018), and Wu et al. (2019); (ii) improving existing models or adopting new trading models, such as DRL agents, as in the works by Liu et al. (2020) and Conegundes and Pereira (2020); (iii) improving the quality of the data used as inputs (which may involve new preprocessing techniques or the generation of new features), such as in the works by Mansar et al. (2017) and Ferreira et al. (2019); (iv) improving the decision-making policy based on the models' and systems' results, such as in the works evaluated by Meng and Khushi (2019); among others. In this work, all of those items are considered in the proposed trading system's design and implementation.

Price prediction methods are essential for investment allocation strategies. Those models are normally divided into: (i) price prediction models (which are regression models that aim to predict a price number); and (ii) price trend prediction models (which are classification models that aim to predict a trend, usually up or down) (KARA; BOYACIOGLU; BAYKAN, 2011; BALLINGS et al., 2015; RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020). Currently, DL models are used for stock price predic-

tion with a better performance than other ML Models (BALLINGS et al., 2015; RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020). Some important works that use ML models for predicting prices on the Brazilian stock market are: Kristjanpoller, Fadic and Minutolo (2014), Freitas, Souza and Almeida (2009), Oliveira, Nobre and Zárate (2013), Nelson, Pereira and Oliveira (2017), and Pauli, Kleina and Bonat (2020).

A critical aspect to evaluate is related to the efficiency of the market. In a highly efficient market (as explored on the Efficient Market Hypothesis or EMH), it is not possible to obtain excessive returns in a sustainable form (NASSIRTOUSSI et al., 2014; FAMA, 1965; MALKIEL; FAMA, 1970; BOLLEN; MAO; ZENG, 2011; NELSON; PEREIRA; OLIVEIRA, 2017). In summary, that means that if an asset or strategy provides excessive returns, other agents in the market will target that asset or adopt that strategy, and the excessive returns will disappear (NASSIRTOUSSI et al., 2014; FAMA, 1965; MALKIEL; FAMA, 1970; BOLLEN; MAO; ZENG, 2011; NELSON; PEREIRA; OLIVEIRA, 2017).

Nevertheless, several works have pointed out that: (i) it is possible the EMH can be applied only partially for developing markets (FISCHER; KRAUSS, 2018; NASSIRTOUSSI et al., 2014; MEHTAB; SEN; DASGUPTA, 2020), creating opportunities for excessive returns; (ii) the use of additional features to the OHCLV data, such as market sentiment, could improve the quality of the predictions, by incorporating important information on the predictions and strategies before they are fully incorporated on the asset's price (leading to higher returns) (NASSIRTOUSSI et al., 2014; BOLLEN; MAO; ZENG, 2011; SOHANGIR et al., 2018); (iii) not always individual agents or the market as a whole behave in a rational manner (NASSIRTOUSSI et al., 2014; BONDT; THALER, 1985; FISCHER; KRAUSS, 2018); and (iv) using complex non-linear models such as deep neural networks and their variations (multilayer perceptron or MLP, long short-term memory networks or LSTM, convolutional neural networks or CNN, among others) could improve pattern recognition capabilities, improving trading results in comparison to the use of traditional econometrics and ML models (BALLINGS et al., 2015; RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020).

The DL models also have a considerable advantage over other models in terms of prediction tasks: they can discover latent features on the data, eliminating the need to manually handcraft features (RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020; JORDAN; MITCHELL, 2015; LECUN; BENGIO; HINTON, 2015), such as the case with generating technical indicators (TI) for conducting technical analysis.

Several works observed that a relatively simple strategy, denominated buy and hold

(BH) strategy, can lead to better results than using traditional trading models (FISCHER, 2018; MENG; KHUSHI, 2019; DANTAS; SILVA, 2018). The BH strategy can be described as buying stocks at the beginning of the period (timestep t0) and selling at its end (last timestep). Although this is rarely used in this simplistic way in real-life trading, several works use it as a baseline due to its good results on several different scenarios over the long term, such as Dantas and Silva (2018), Wang et al. (2017), Conegundes and Pereira (2020), Li, Rao and Shi (2018), Liu et al. (2020), and Yang et al. (2020).

However, many works, such as Wang et al. (2017), Conegundes and Pereira (2020), Liu et al. (2020), and Yang et al. (2020), observed that their proposed models and strategies outperformed the BH strategy on different stock markets. Usually, returns (annual or cumulative) or the Sharpe ratio (an important metric that considers both risks and returns of a strategy) are evaluated on those works. Rarely, several metrics are used for trading evaluation, as is the case with the works by Li, Rao and Shi (2018), Liu et al. (2020), and Yang et al. (2020). Notwithstanding, few works analyze multiple trading scenarios.

A series of works in the literature, such as the ones by Liu et al. (2020) and Yang et al. (2020), have observed that the use of RL and DRL models may present better results than the BH strategy, as well as several other strategies, such as mean-variance and momentum trading. As an example, in the work of Liu et al. (2020), the authors observed an annualized return of 21.40% (versus 8.38% for min-variance and 10.61% for BH), a Sharpe ratio of 1.38 (versus 0.44 for min-variance and 0.48 for BH), and a maximum drawdown of 11.52% (versus 34.34% for min-variance and 37.01% for BH).

Lastly, a considerable amount of works in the literature use ML metrics for evaluating the implemented models, such as regression and classification metrics. Some examples of those works are: Mansar et al. (2017), Neuenschwander et al. (2014), Weng et al. (2018), Jin, Yang and Liu (2020), Medeiros and Borges (2019), and Souma, Vodenska and Aoyama (2019). Nevertheless, as observed by Sezer, Gudelek and Ozbayoglu (2020), Fischer (2018), and Meng and Khushi (2019), it is essential to evaluate financial metrics to clearly understand the possible impact of using those models. The following section contains the main and the secondary research questions and the work's main objective.

1.2 Research questions and objective

This research's main objective is to investigate and propose a system for automatic asset trading considering multiple features, time series prediction, sentiment analysis, and

deep reinforcement learning models. It will be tested with the BOVA11 ETF, which is traded at B3 to match the Ibovespa index. This ETF was chosen because it represents the total market. This avoids potential price distortions related to a specific sector in the period evaluated, resulting in more generalizable results. Additionally, market indices are the most used assets in the literature to evaluate stock trading models.

The trading system will then be evaluated in two trading scenarios, and the most important module configurations will be evaluated in-depth. Lastly, six financial metrics will be evaluated, analyzing aspects related to returns, risks, volatility, and losses compared to a traditional baseline, the BH strategy.

To better address the problem described in the previous section, considering the domain background presented in section 1.1 and the need to explore different aspects of the trading system proposed in this work, several research questions were proposed.

The main research question of this work was: **"Does the proposed system present better trading results than the BH strategy, considering the evaluated risk and returns metrics and two different trading scenarios?"**. This question is vital because it addresses the main concerns of all investors: maximizing profit over time while also considering the strategy's potential risks. Although the BH strategy is simple, it has been shown to provide difficult-to-beat results in the longer term, and has become a standard baseline for many works, such in the ones by Wang et al. (2017), Conegundes and Pereira (2020), Li, Rao and Shi (2018), Liu et al. (2020), and Yang et al. (2020).

Nevertheless, very few works in the literature explore the use of DRL agents versus the BH strategy for the Brazilian market, with the work by Conegundes and Pereira (2020) as the best example. Unlike the work of Conegundes and Pereira (2020), in the present dissertation, several models are compared using different inputs and two trading scenarios. This is essential to understand better the behavior of the DRL agents on stock market trading. The extensive reviews by Fischer (2018) and Meng and Khushi (2019) observed very few papers on the Brazilian stock market.

Eight secondary questions were proposed to evaluate different aspects of the trading system and its modules. Their results are helpful for both practitioners and researchers on the use of ML, DL, and DRL models on the financial domain. These were:

- **SRQ1: "Does the use of DRL with sentiment analysis improve stock trading in terms of profits in relation to the use of the BH strategy?"**. This is an essential question to evaluate the use of different features that are not

generally considered in trading using DRL agents.

- **SRQ2: "Which model results in the best forecast of market indices prices? Econometrics, ML, or DL models?"**.This is a crucial question for evaluating different model categories The in-depth hyperparameters analysis is also fundamental to better understand each model's behaviors and better guide practitioners in model and hyperparameter value choices.

- **SRQ3: "Does the use of TIs as features improve the forecasts of market indices prices?"**. This is a vital question related to an ongoing debate for decades related to the use of TIs (derived from the technical analysis literature). In this work, the use of TI as an input for the price prediction module.

- **SRQ4: "Which model (considering MLP and CNN) best predicts market sentiment?"**. This is an essential question for evaluating the results of two state-of-the-art DL models for market sentiment prediction in Portuguese.

- **SRQ5: "Does the use of a dictionary (considering Sentilex, Oplexico, and WordnetAffectBR) improve the prediction of market sentiment for financial news headlines in Portuguese?"**. This is an important question related to an ongoing debate for many years related to the importance of using sentiment dictionaries or lexicons for improving market sentiment prediction. However, few works consider this question applied for the Portuguese language, as most of the literature, tools, and dictionaries are related to the English language.

- **SRQ6: "Does the use of the price prediction module improve stock trading results?"**.

- **SRQ7: "Does the use of sentiment analysis of news headlines improve stock trading results compared to the BH strategy?"**.

- **SRQ8: "Does the use of TIs improve the stock trading results of DRL models?"**.

The following section describes the primary motivations for this study based on a thorough literature review.

1.3 Motivations for the study

As was explored in section 1.1, there was a significant increase in the number of investors in the Brazilian stock market (B3, 2021a). It is also essential to observe the increased interest in algorithmic trading worldwide (WU et al., 2019; TRELEAVEN; GALAS; LALCHAND, 2013). Although this group of methods was already used in the American stock market since the 1980s (TRELEAVEN; GALAS; LALCHAND, 2013), it is increasingly adopted in the Brazilian stock market. This is an important motivation for exploring several price prediction and trading models, especially the most recent ones, such as DRL agents (FISCHER, 2018; MENG; KHUSHI, 2019; SOHANGIR et al., 2018; RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020).

Another critical motivation to explore new trading systems, models, and strategies is the considerable impact of financial crisis and market anomalies, in which several investors present irrational behaviors (NASSIRTOUSSI et al., 2014; FISCHER; KRAUSS, 2018; BONDT; THALER, 1985; TRELEAVEN; GALAS; LALCHAND, 2013; FISCHER, 2018; MENG; KHUSHI, 2019). Systems that extract patterns from the market could guide investors during highly volatile and uncertain scenarios. Therefore, a data-driven trading system could assist on decision-making during highly volatile scenarios.

Another motivation that drives most works on developing trading systems and better price prediction models is that improvements on current trading systems would increase investors' revenues, generating lots of incentives for new research on this field (TRELEAVEN; GALAS; LALCHAND, 2013; FISCHER, 2018; MENG; KHUSHI, 2019; CONEGUNDES; PEREIRA, 2020; LIU et al., 2020; YANG et al., 2020). In asset price prediction, new DL models for time series analysis have been used for several markets and assets (RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020). Most works observe significant results by using the LSTM model (RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020). This is a state-of-the-art DL model that is mainly used for autocorrelated data on several domains.

The LSTM model was proposed by Hochreiter and Schmidhuber (1997), and it can be defined as a feedforward neural network that uses its output as a recurrent input to learn from data with past values. This allows the autocorrelation of the data to be maintained, unlike models like the Support Vector Regression (SVR). Although the idea of recurrence on deep neural networks had been studied before developing the LSTM, an issue denominated vanishing problem was an essential constraint for maintaining autocorrelation properties on deep neural networks. The LSTM was the first deep neural

network to solve this issue (SEZER; GUDELEK; OZBAYOGLU, 2020; HOCHREITER; SCHMIDHUBER, 1997; KARPATHY; JOHNSON; FEI-FEI, 2015; RYLL; SEIDENS, 2019).

Notwithstanding, few works consider the use of LSTM to improve price predictions on developing markets, as was observed by Mehtab, Sen and Dasgupta (2020), and Nelson, Pereira and Oliveira (2017). They all have observed improvements in relation to the baselines, but most have not considered important econometrics models, such as the autoregressive integrated moving average (ARIMA) or seasonal autoregressive integrated moving average (SARIMA), as baselines. One common baseline used is the SVR model, a model that does not consider autocorrelation in the data. Also, few works consider an in-depth analysis of hyperparameter values and cross-validation methods for predicting stock prices, as in the works by Ferreira et al. (2019), Persio and Honchar (2016), Eapen, Verma and Bein (2019), and Ballings et al. (2015). Comparing several relevant models for price prediction on the Brazilian stock market is a strong motivation for this work.

Two in-depth literature reviews conducted by Nassirtoussi et al. (2014) and Sohangir et al. (2018) have shown that sentiment analysis provides important features for improving price prediction, considering different markets, assets, and time windows. Few works explore the use of market sentiment on price prediction on the Brazilian market. The works by Neuenschwander et al. (2014) and Medeiros and Borges (2019) have explored these impacts and have observed an increase in prediction performance, considering different metrics. Nevertheless, a more in-depth analysis of the use of sentiment analysis for price prediction in Brazil, considering the impact of the different hyperparameters (and, especially, sentiment dictionaries or lexicons), is highly desirable. This work explored the impact that the sentiments may have on daily trading in two different scenarios.

According to the in-depth literature reviews on the use of sentiment analysis by Nassirtoussi et al. (2014) and Medhat, Hassan and Korashy (2014), DL models, such as CNN using word embeddings, can provide the best results in terms of sentiment prediction. They also observed that daily frequency is the most used for predicting both prices and sentiments on different works in the literature.

There is an increasing trend in using market sentiment for price prediction (WENG et al., 2018; GHOSAL et al., 2017). These typically consider both price or price trend prediction models and sentiment analysis models to make a final prediction. This can then be used for trading. Few works in the literature explore those models on the Brazilian stock market. This is especially true when considering the use of a hybrid approach for sentiment

analysis as done in this research, in which both ML techniques and sentiment dictionaries or lexicons are used together (MEDHAT; HASSAN; KORASHY, 2014; NASSIRTOUSSI et al., 2014; RUDER; GHAFFARI; BRESLIN, 2016).

According to Ferreira et al. (2019) and Mansar et al. (2017), the increasing interest in market sentiment for price prediction and trading purposes can be illustrated by the SemEval 2017 Task 5 challenge, an important financial sentiment analysis regression task. This task developed a high-quality labeled dataset for sentiment analysis of news titles in English, focusing on market sentiment. It is critical to observe that DL models were the ones that presented the best results on that challenge, mainly the CNN (MANSAR et al., 2017). However, no such open dataset exists in Portuguese. The present research's sentiment analysis module is inspired by the SemEval 2017 Task 5 challenge approach.

Important research was conducted by Johnman, Vanstone and Gepp (2018), using sentiment analysis based on news from 2000 to 2016 with a lexicon-based model to predict FTSE-100 returns and volatility. The lexicon-based model used presented a better return than the BH strategy (cumulative return of 1.347 for their model versus -0.0929 for BH). Nevertheless, the model used was considerably simple (based mainly on a predefined lexicon) and did not learn with the data. Therefore, this could be improved by using a DL model, such as in this research. A DL model could learn over time, besides using different dictionaries and being a part of a model ensemble.

As described in section 1.1, new AI models for automated decision-making (such as RL and DRL), which presented excellent results in the gaming domain (SILVER et al., 2016; MNIH et al., 2015; LI, 2018; FRANÇOIS-LAVET et al., 2018; VÁZQUEZ-CANTELI; NAGY, 2019), are increasingly being used for financial trading (FISCHER, 2018; MENG; KHUSHI, 2019). Fischer (2018) and Meng and Khushi (2019) conducted in-depth analyses on the use of RL and DRL for stock trading, observing that there is an increased interest in using those models for trading systems and strategies.

Liu et al. (2020) and Yang et al. (2020) implemented several DRL models for trading stocks in the American stock market. They have observed a considerable improvement in the annualized returns, Sharpe ratio, and maximum drawdown using their DRL trading model related to the minimum-variance and BH strategies.

It is essential to observe that none of the models explored by Pendharkar and Cusatis (2018) use deep neural networks to learn better the market dynamics and possible non-linearities on the data. The use of deep neural networks as the "brain" of the agent (as in the case of the DRL models) could improve the model's pattern recognition on a very

uncertain domain with many different states, as is the case of the stock markets.

However, few works consider RL and DRL's use for stock trading in the Brazilian market. The most relevant works are the ones by Dantas and Silva (2018) and Conegundes and Pereira (2020). Nevertheless, those works did not explore the use of additional sentiment features, TI's impact, or additional price prediction features. Therefore, exploring the use of state-of-the-art DRL agents for financial trading in the Brazilian stock market is one of this research's main motivations.

It is also important to note that the present thesis is one of the few works to explore in-depth: price prediction and sentiment features on state-of-the-art DRL agents for stock market trading. Also, it is one of the first works to apply an automated trading system considering those models and features on the Brazilian stock market.

Lastly, one of the objectives of a trading system is that it should reflect real-life scenarios and address essential aspects, such as: considering multiple relevant financial metrics, being easily replicable, consider the impact of TI as additional features, and evaluating multiple trading scenarios (FISCHER, 2018; MENG; KHUSHI, 2019; LIU et al., 2020; CONEGUNDES; PEREIRA, 2020; YANG et al., 2020; SEZER; GUDELEK; OZBAYOGLU, 2020). Few of the proposed systems and models in the literature attend to those critical aspects. The following section describes the scope of this thesis.

1.4 Scope of the thesis

This section describes the main scope of this research. This definition is crucial for better understanding the assumptions made and how the trading system can be applied for real-life financial trading. The scope also guided the design and implementation of the system and its components. This work is based on eleven main points:

1. It may be possible to obtain excess returns on the stock market. Although several authors consider this a direct opposition to the EMH, many works have shown that it is possible to obtain excess returns in the short term (NASSIRTOUSSI et al., 2014; BOLLEN; MAO; ZENG, 2011; SEZER; GUDELEK; OZBAYOGLU, 2020; FISCHER, 2018; MENG; KHUSHI, 2019). Additionally, the Brazilian stock market is not as developed as the American or Western European markets. Therefore, this may create further opportunities for profit (NASSIRTOUSSI et al., 2014);

2. The maximum order size or maximum daily order (the maximum number of stocks

a system can trade within a day) may influence the model's results, especially on highly volatile scenarios. Two maximum order sizes were evaluated: 10 (representing a severe constraint) and 200 (a value in line with DRL models in the literature);

3. A trading system may present different results in different scenarios. Therefore, two significant scenarios were considered: a longer-term scenario (2019 and 2020) and a shorter-term scenario with high volatility (2020);

4. Only one asset was considered, the BOVA11 ETF. It is a fund that makes it easier to trade based on the Ibovespa market index. All the design, implementations, and analyses consider only one asset. Nevertheless, this could be expanded, with a few modifications on the DRL agent's Markov Decision Process (MDP) and the trading environment, to incorporate multiple assets;

5. The proposed system contained three modules based on the most relevant features and results observed throughout the literature. These modules were: price prediction (M1), market sentiment prediction (M2), and automated trading (MT);

6. Only the three main actions were considered for the DRL agent: buy, sell, and hold. These are also the most used actions in the trading literature, as observed in the works by Liu et al. (2020), Yang et al. (2020), Conegundes and Pereira (2020), Li, Rao and Shi (2018), and Lei et al. (2020). Additional actions such as shorting and using leverage were not considered, as they build on top of those basic actions. However, they could be added as modifications on the DRL agent's MDP;

7. All transactions have costs, and these must be taken into consideration by the trading system to be more realistic. This is pointed out as a significant problem in several works in the literature and may influence significantly on the system's returns (FISCHER, 2018; MENG; KHUSHI, 2019; LIU et al., 2020);

8. Only daily trading was considered, as this is the most common form of trading for most investors. Its dynamics are similar if long-term trading is considered (but more frequent). Additionally, it is considerably easier for investors to gather daily data, as they are freely available through several websites. Intraday data is more restricted and may depend on using services that may not be available to all investors (especially small-sized investors). Several works point out that market dynamics are different from intraday trading, especially for trading by the minute or second (CHONG; HAN; PARK, 2017; MEHTAB; SEN; DASGUPTA, 2020; NELSON;

PEREIRA; OLIVEIRA, 2017; SOUMA; VODENSKA; AOYAMA, 2019). Nevertheless, the proposed system can be adapted for intraday trading;

9. It is important to consider multiple trading metrics related to returns, risks, volatility, and losses. Most RL and DRL trading literature works consider only returns or risk-related metrics (DANTAS; SILVA, 2018; WU et al., 2019; LI; RAO; SHI, 2018; LEI et al., 2020). Notwithstanding, especially when considering very negative scenarios, it is crucial to consider metrics related to volatility and losses;

10. As the market dynamics are complex, unknown, and may constantly be changing, the use of DRL agents that are model-free for the trading module is the best option, as explored by Fischer (2018) and Meng and Khushi (2019);

11. This work did not focus on production server implementation.

The following section describes the main assumptions considered for the proposed system (and its components) design and implementation.

1.5 Main assumptions

To design and develop the trading system and its components and test them, seven main assumptions were considered. It is important to note that they are directly connected to the last section's research scope. The main assumptions were:

1. No market agent can influence, by itself, the asset's price. This is true for market indices and large companies, so it is realistic in the scenarios evaluated in this work. Nevertheless, if a practitioner wants to use the proposed system for an asset that has a small size or that is not liquid, this assumption must be more carefully analyzed;

2. Sentiment analysis can be used to identify and predict market sentiment, which can then be used to improve trading or price prediction (NASSIRTOUSSI et al., 2014; SOHANGIR et al., 2018; BOLLEN; MAO; ZENG, 2011; NEUENSCHWANDER et al., 2014; SOUMA; VODENSKA; AOYAMA, 2019). Several data sources can be used. In this research, financial news titles or headlines were used, as they present a summary of the main idea of each specific news;

3. General sentiment dictionaries can be used for improving market sentiment prediction. This has been observed in several works, such as Mansar et al. (2017)

and Ferreira et al. (2019). This is an important assumption because there are no openly available financial sentiment dictionaries in Portuguese, unlike the English language. The latter has financial sentiment dictionaries that are openly available and well-accepted, such as Loughran and Mcdonald (2011), which have been proven to improve the quality of the sentiment prediction (FERREIRA et al., 2019; LOUGHRAN; MCDONALD, 2011; MANSAR et al., 2017);

4. The trading system cannot borrow additional funds or use leverage. This is the case of most works in the literature (FISCHER, 2018; MENG; KHUSHI, 2019). It is possible to implement those functionalities in future works;

5. The trading system's initial funds on both trading scenarios is R$ 100,000. This value is based on Liu et al. (2020) but should have no significant impact on the final results. This value must be large enough, so it is not the main initial restraint for trading. Nevertheless, it is possible to change this value freely on the MT module;

6. It is essential to consider the trading cost for each trade. A trading cost of 0.01% was considered for every trade in this work, as used by Liu et al. (2020). This makes the model more realistic, as the trading costs are an essential part of the final results (especially the returns). It is also possible to change this value within the trading system, using both a flat fee per trade or a percentage cost for every trade;

7. In the ML sense, the trading scenarios were considered test scenarios. Therefore, the models only learn before those periods. During the trading periods, the models were not learning new policies or patterns, only applying the policies and patterns recognized during their training subsets. This is essential to conduct a more fair comparison and is the standard approach followed in the literature (FISCHER, 2018; MENG; KHUSHI, 2019). However, in real-life scenarios, the model could be retrained daily to improve its pattern recognition and trend identification capacities.

The following section describes this document's structure and an overview of each of its chapters.

1.6 Document structure and chapter overview

This work is divided into six chapters. Chapter 1 contains the background of this work, its main objective and research questions, and the main assumptions of the proposed system. Chapter 2 contains a literature review of the main areas and domains that

are relevant to this work: price prediction, sentiment analysis, stock trading, and the econometrics, ML, DL, and DRL models that will be used.

Chapter 3 contains the description of the methodology used in this research, with a thorough description of the main activities conducted. Chapter 4 contains the experiments' main results with the proposed system, considering both its components and the two trading scenarios.

Chapter 5 contains several meaningful discussions related to the system's results and its components and architecture. It clearly describes this work's four contributions, its main limitations, and suggestions for three different applications. This chapter concludes with recommendations for future work, both considering different models and markets and different configurations for each module. Chapter 6 concludes the research, summarizing the main objective and research questions, the proposed trading system and its modules, and its main contributions.

2 LITERATURE REVIEW

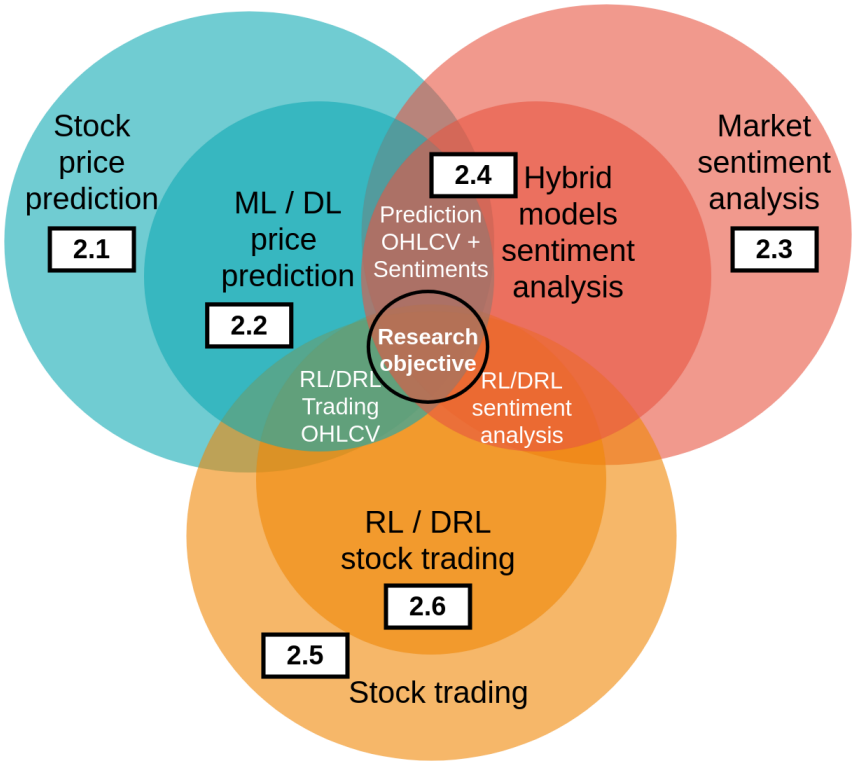
This section describes in-depth the most relevant and state-of-the-art research in the main themes related to this thesis. Figure 1 illustrates those themes and their respective sections. This forms the basis for the research, the gap identification, and the proposing of the automated trading system. It is essential to observe that the AI techniques and models, especially the DRL agents, are considered state-of-the-art for stock market trading because of their capacity to learn in complex, non-linear environments (CONEGUNDES; PEREIRA, 2020; FISCHER, 2018; YANG et al., 2020; MENG; KHUSHI, 2019).

This chapter is organized in the following sections: 2.1 describes the stock price prediction task, its main difficulties, and the traditional econometrics models explored in this work (ARIMA, SARIMA, and seasonal autoregressive integrated moving average with external factors or SARIMAX); 2.2 describes the use of ML and DL for stock price prediction, describing the ML (SVR and AdaBoost) and DL (LSTM) models explored in this work; 2.3 explores the role sentiment analysis on the stock markets, considering aspects related to model categories, data sources, and main concerns; 2.4 describes the use of hybrid models (using lexicons and ML or DL models) for analyzing and predicting the stock market sentiment; 2.5 summarizes the main aspects related to stock trading and algorithmic trading and the importance of using domain metrics for model evaluation; 2.6 explores the use of RL and DRL, state-of-the-art models, for automated stock trading; and 2.7 concludes the chapter, summarizing its main points and fundamental concepts.

2.1 Stock price prediction

Nassirtoussi et al. (2014) state that one of the most critical aspects of market economies is the financial market. The ability to better predict future prices and price trends in a market economy has two main advantages: (i) avoiding financial losses; and (ii) making financial gains (NASSIRTOUSSI et al., 2014). Improving asset price predictions could improve current market trading strategies (KARA; BOYACIOGLU; BAYKAN, 2011).

Figure 1 – Main themes of the literature review chapter and their respective sections



Source: elaborated by the author.

Nevertheless, asset prices and price trends are very challenging to predict (NASSIRTOUSSI et al., 2014). Some of the main reasons that explain this fact are short and long-term trends, seasonal patterns, cyclical fluctuations, and noisy data (YADAV; JHA; SHARAN, 2020). There are also crucial aspects related to: (i) political and economic contexts; (ii) different market factors; (iii) investors' behavior on different scenarios; and (iv) development and use of new technologies, among others (JIN; YANG; LIU, 2020). Also, the high volatility and non-stationary aspect of the stock markets increase the complexity of prediction tasks (HU et al., 2018).

The EMH was a theory proposed in 1965 to explain the behavior of price movements in the stock market and why it is so difficult to predict it (FAMA, 1965). According to Nassirtoussi et al. (2014), analyzing the work by Fama (1965), the EMH states that the price movements in a stock market follow a random walk model and, therefore, are unpredictable. This infers that it is not possible to achieve sustainable excess returns at the decision-making moment considering the potential impacts of risks. However, in subsequent works, Malkiel and Fama (1970) concluded that this only fully applies to specific situations: in strongly efficient markets without information asymmetry. In other words, the better the information access in the market, the more it may resemble a random walk.

However, several works have observed that this is not the case of markets in developing countries, as these are not considered strongly efficient markets (NASSIRTOUSSI et al., 2014).

According to Bollen, Mao and Zeng (2011) and Nelson, Pereira and Oliveira (2017), one crucial aspect of the EMH is that, according to its original formulation, stock price changes are driven by: (i) new information rather than old information; and (ii) and current prices, rather than historical prices. According to this theory, the current price already incorporates all available information in the market. Therefore, only new information could impact its price. According to Freitas, Souza and Almeida (2009) and Fischer and Krauss (2018), the EMH has been under discussion and empirical testing since it was proposed because several observed effects tend to contradict this hypothesis (such as irregularities, calendar effects, among others). These usually are called market anomalies (FISCHER; KRAUSS, 2018).

According to Nelson, Pereira and Oliveira (2017), the random-walk hypothesis states that an asset's price changes are independent of its history. In this sense, the best strategy would be to predict that tomorrow's price would be today's price plus random noise. This is sometimes used as a baseline for comparing prediction models. According to Ballings et al. (2015), the EMH was based on linear statistical algorithms. Therefore, using algorithms that can capture complex non-linear dynamics on the data could, in theory, go against the EMH and provide excessive returns due to partial assets' predictability. This has been attempted (and, to a certain extent, observed) by several works in the literature (BALLINGS et al., 2015). Additionally, many works such as the one by Bollen, Mao and Zeng (2011) try to evaluate the impact that news and posts on social media could have on price and if this would allow for better market prediction (as there is a period between the news or post-release and its incorporation on the asset' price).

However, Bollen, Mao and Zeng (2011) observe that: (i) several studies observe that prices (especially on developing markets) do not follow a random walk and that some degree of asset price prediction is possible; and (ii) many other studies observed that the use of news and social media messages could improve price prediction, even if it is only in a short time window (minutes to hours after the news release). Different methods and techniques have been used to predict and trade stock market assets, such as optimization, signal processing, time series analysis, and machine learning (CONEGUNDES; PEREIRA, 2020).

Among the researchers and practitioners that believe that stocks prices and trends can

be predicted, at least partially, there are four main sets of models, tools, and techniques that can be used: (i) fundamental analysis; (ii) technical analysis; and (iii) ML-based models (HU et al., 2018; NASSIRTOUSSI et al., 2014). Additionally, traditional statistical analysis models (also called econometrics models) are used to predict stock prices in several works in the literature. This can be considered the fourth category of models, as it presents different aspects compared to the other three approaches.

Fundamental analysis is related to the use of macro and microeconomic information, as well as specific techniques and formulas, to determine the intrinsic value of an asset (NASSIRTOUSSI et al., 2014; HU et al., 2018). This allows the investor to compare an asset's market value with its intrinsic value and then decide if it is rational to buy or sell that asset at that moment. However, it is vital to observe that there is no widely accepted definition of how to calculate an asset's intrinsic value, with many proposals by different researchers and practitioners.

The data used in fundamental analysis is considerably challenging to gather and evaluate because it usually is unstructured and may present varying time windows (NASSIRTOUSSI et al., 2014). For example, company reports present relevant information related to a company but may be released only once a year. Additionally, company reports contain unstructured text, and mining relevant information from those reports is complex. According to Nassirtoussi et al. (2014), fundamental data present in unstructured text is one of the most difficult ones to work with. Some examples of data sources for fundamental analysis are: company reports, sector reports, official documents, news, news headlines (or news titles), social media posts, blogs, messages on forums, among others.

The second approach, technical analysis, is related to the idea that market movements happen in cycles and that trends tend to repeat themselves (NASSIRTOUSSI et al., 2014). Generally, it uses chart analysis or mathematical models to identify patterns and predict future trends (OLIVEIRA; NOBRE; ZÁRATE, 2013; PERSIO; HONCHAR, 2016). Then, the decision-maker applies a set of rules to decide if a specific asset should be bought or sold. As pointed out by Hu et al. (2018), this approach's critical limitation is that it cannot help identify patterns that influence the market dynamics outside of the open, high, low, and close prices, and volume (OHLCV) data. For example, it does not evaluate the impact that market news could have on an asset's price, evaluating only data that is already incorporated on the price.

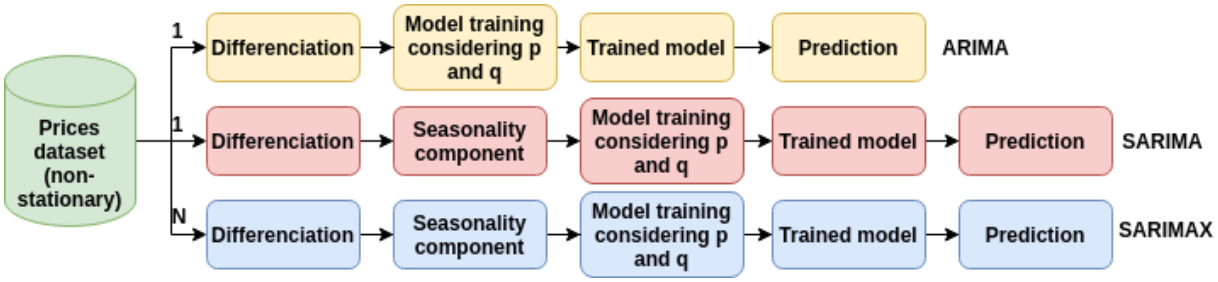
According to Neuenschwander et al. (2014), the proponents of technical analysis believe in repeated patterns over time. Some of the main assumptions of technical analysis

tools pointed out by Nelson, Pereira and Oliveira (2017), Persio and Honchar (2016), Oliveira, Nobre and Zárate (2013), and Weng et al. (2018) are: (i) prices are defined only by the relation between supply and demand; (ii) long and short term trends influence asset prices; (iii) changes in the structure of supply or demand can cause the trends to change direction (also denominated reversing the trend); (iv) it is possible to identify those changes using indicators (the TIs); and (v) there are a set of patterns that repeat themselves over time.

Dantas and Silva (2018) state that TIs are metrics based on OHLCV data to obtain additional information from the raw data. They can be considered as handcrafted features designed by experts. Nevertheless, there are divergent opinions on the literature related to their effectiveness on trading systems. Two very important TIs are the Bollinger Bands (BB) and the moving average convergence divergence (MACD). The Bollinger Bands are used to delimit a region of two standard deviations (upper and lower bands) surrounding the 20 days simple moving average of the price (middle band). As the middle band crosses the upper or lower bands, rules are used to buy or sell the specific asset (DANTAS; SILVA, 2018). The MACD is an indicator derived from the 9, 12, and 26 days exponential moving average of the prices. A set of rules is used to buy or sell the specific asset based on the moving averages' behaviors (DANTAS; SILVA, 2018). Both TIs are considered in this work, as well as other relevant ones used by works such as Kara, Boyacioglu and Baykan (2011), Weng et al. (2018), and Wu et al. (2019). To better understand their roles, these are divided into four classes in this thesis: volume, volatility, trends identification, and momentum.

The third approach, the use of ML models, can use data sources and indicators from fundamental and technical analyses (NASSIRTOUSSI et al., 2014). Its main advantage is learning from the data, identifying new patterns as more data is added (JORDAN; MITCHELL, 2015). ML models can also deal with non-linear datasets with a large number of dimensions and a low signal-to-noise ratio (JORDAN; MITCHELL, 2015; NELSON; PEREIRA; OLIVEIRA, 2017; LECUN; BENGIO; HINTON, 2015). ML models are used to extract sentiment data from company reports, news, among other data sources in the case of fundamental analysis. In technical analysis, which is the most common in the literature, ML models are used to extract patterns from OHCLV data and TIs. The ML and DL models are explored in-depth in Section 2.2. In this thesis, aspects of fundamental analysis (market sentiment) are explored on the M2 module, and aspects of price trends and TIs are explored on modules M1 (to predict future prices) and MT (to improve trading decision-making).

Figure 2 – Main components of the ARIMA, SARIMA, and SARIMAX models



Source: elaborated by the author, based on Box et al. (2015).

Lastly, several works consider econometrics models to predict asset prices and trends. These are also denominated conventional time series analysis models and include the ARIMA and its components as models (AR, MA, and ARMA) (JIN; YANG; LIU, 2020), as well as its seasonal (SARIMA and SARIMAX) and multivariate counterparts (ARIMAX and SARIMAX). These models were specifically designed to address prediction tasks on stationary time series. As observed by Jin, Yang and Liu (2020), those models tend to consider only the time series, ignoring other potential influencing factors on the prices.

According to Rundo et al. (2019), the ARIMA model is a generalization of the ARMA model and its variations (AR and MA models) for application on non-stationary series due to its differencing component (which transforms a non-stationary series into a stationary one). The stationarity property can be summarized as the time series having constant statistical properties (mean and standard deviation) over the whole dataset. Therefore, a stationary series cannot present trends (which are common in financial data).

In the M1 module of this thesis, three econometrics models were implemented: ARIMA, SARIMA, and SARIMAX. These were used as baseline models for comparing with the ML and DL prices, as they are used in real-life situations. As illustrated in Figure 2, the main differences between them are (BOX et al., 2015): (i) ARIMA and SARIMA are univariate models; and (ii) SARIMA and SARIMAX consider seasonal components. All models have a differentiation component, which aims to transform the series from non-stationary into stationary. The SARIMA and SARIMAX extract a seasonal component. Then, all models are trained, evaluating different values for p (autoregressive component) q (moving average component), and the seasonal components (BOX et al., 2015). The model prediction is then compared with the real value, and the model error is calculated. The work by Box et al. (2015) contains a further description of the ARIMA, SARIMA, and SARIMAX models' workings.

Junior, Salomon and Pamplona (2014) evaluated using the ARIMA model to forecast the monthly Ibovespa index from 1995 to 2013. The main baseline models used were AR1 (autoregressive model with the autoregressive component equal to 1), single exponential smoothing, and double exponential smoothing, and the evaluation metric was the MAPE. The authors concluded that the ARIMA model results in the best MAPE (0.064% versus 0.052% for AR1, 0.086% for single, and 0.118% for double exponential smoothing). However, the authors did not compare this model with ML and DL models, which could provide better results according to the literature. Additionally, the authors did not explore the use of ARIMA variations that consider seasonal components (SARIMA, SARIMAX) or external factors (ARIMAX, SARIMAX).

Siami-Namini, Tavakoli and Namin (2019) compared the ARIMA, univariate LSTM, and univariate Bidirectional LSTM (BiLSTM) models for predicting closing prices of several indices and stock markets with varied frequencies (daily, weekly, and monthly) from 1985 to 2018. The authors observed that both the LSTM and BiLSTM provided considerably better root mean squared error (RMSE) than the ARIMA model. However, although the BiLSTM presented the best overall results, its convergence was considerably slower than the LSTM. Therefore, it is possible to conclude that the BiLSTM would demand more training data and more computational resources than the LSTM.

Chong, Han and Park (2017) compared the use of DL models, ARIMA, and ensemble models for the prediction of stock intraday returns, concluding that: (i) the DL models perform better than the ARIMA models; and (ii) that an ensemble of ARIMA and DL model provided better results than the individual models. According to Mehtab, Sen and Dasgupta (2020), econometrics models such as the ARIMA model tend to perform poorly on volatile data with randomness and noise in it, which is the context observed on intraday prices.

The extensive literature review conducted by Ryll and Seidens (2019) contains a thorough evaluation of ML and DL models used for stock price prediction and stock trading and how they compare to the ARIMA model. The following section explores ML and DL models' use for predicting stock prices and trends and a description of the ML and DL models implemented in this work on the M1 Module.

2.2 ML and DL for stock price prediction

Jordan and Mitchell (2015) provide an important review of the field of ML, considering theoretical foundations, trends, and possible uses. According to those authors, ML is related to the development of computer programs that can improve automatically through more data (or, in other words, to improve with experience through several interactions). They also define the ML field of study as being in the intersection between computer science, statistics, AI, and data science. It is also common to describe ML and DL models as learning machines. It is also important to note that DL is a subset of ML that uses artificial neural networks (ANN) with multiple hidden layers (LECUN; BENGIO; HINTON, 2015).

According to Jordan and Mitchell (2015), a learning problem in the context of ML can be defined as improving a quality metric when executing a specific test through training experience and repeated interactions. One important characteristic of ML models is that they can be used for very different tasks with a similar architecture, unlike the considerable modifications needed to adapt rule-based systems or expert systems to other domains. The work by Jordan and Mitchell (2015) contains a thorough exploration of the ML field of study.

An online questionnaire conducted by Oliveira, Nobre and Zárate (2013) with ten trading professionals (with 50% having three years or more of experience) in Brazil to evaluate the use of different stock price and price trend prediction models pointed out that 80% of the respondents do not use computer techniques and models to trade in the stock market. Although this is a small sample, it was targeted towards experts and can help to infer that the adoption of ML models in Brazil was shallow when the questionnaire was applied.

Rundo et al. (2019) conducted an in-depth survey of ML models used in the financial domain, focusing mainly on stock prediction and portfolio allocation tasks. The authors observed that traditional models such as the ARIMA (and its variations) and the exponential smoothing model tend to perform poorly compared to DL models for the stock price and volatility prediction due to several aspects of the data: complexity, high dimensionality, and causal dynamicity. They also observed that DL models such as the LSTM tend to perform better than the support vector machine (SVM) and the MLP models in several stock trends and stock price prediction works. Among the evaluated works, the most commonly used methods were: ARIMA, MLP, SVM, and LSTM. The least used method was DRL. Rundo et al. (2019) conducted an in-depth exploration of several works

that use ML, DL, and DRL in the financial domain.

In this research, two traditional ML models were implemented: the SVR and the AdaBoost (a state-of-the-art boosting model, typically used with decision trees as its weak learners). These are explored in the following paragraphs.

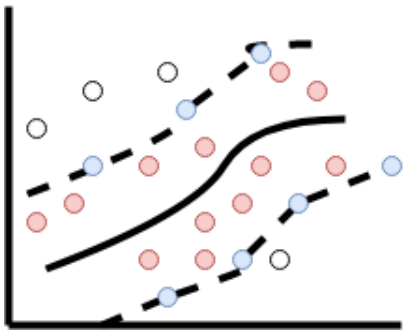
According to Persio and Honchar (2016), SVMs were the optimal choice for stock trend prediction before developing DL models. They tend to present better results than the use of ARIMA models. The SVM can be described as a non-probabilistic binary linear classifier for supervised learning, which can be used for non-linear problems through the application of the kernel trick technique (NASSIRTOUSSI et al., 2014; CHANG; LIN, 2011; DRUCKER et al., 1997; RUNDO et al., 2019). The SVM solves a quadratic programming optimization to identify a hyperplane that separates the classes, considering the maximum margin possible with the chosen hyperparameters (CHANG; LIN, 2011; DRUCKER et al., 1997; RUNDO et al., 2019). The SVR is a variation of the SVM for regression problems proposed by Boser, Guyon and Vapnik (1992) and Drucker et al. (1997).

The main objective of the SVR is to fit the observations of the dataset within the space between the boundaries defined by a parameter named epsilon. The SVR maximizes the margins that separate the different classes, and the kernel trick is used to separate non-linear data (RUNDO et al., 2019; KARA; BOYACIOGLU; BAYKAN, 2011; DRUCKER et al., 1997; CHANG; LIN, 2011). It was implemented in this thesis due to two main reasons: (i) it demands fewer data for recognizing patterns in comparison to the LSTM; and (ii) it is one of the most widely used ML models for stock prediction (RUNDO et al., 2019). Figure 3 illustrates the results of the SVR model on a general case of use. The dashed lines represent the margins (they are non-linear due to the use of the kernel trick). The white dots are clearly separated classes, while the green dots are the support vectors, and the red dots are data points inside the model boundaries.

It is important to note that the model hyperparameters are used to determine the hyperplanes and the support vectors (RUNDO et al., 2019; KARA; BOYACIOGLU; BAYKAN, 2011; DRUCKER et al., 1997; CHANG; LIN, 2011). After the model is trained and the margins are defined, it is used to make predictions based on the input data's features. The works by Rundo et al. (2019), Boser, Guyon and Vapnik (1992) and Drucker et al. (1997) contain a thorough description of the workings of the SVM and SVR models.

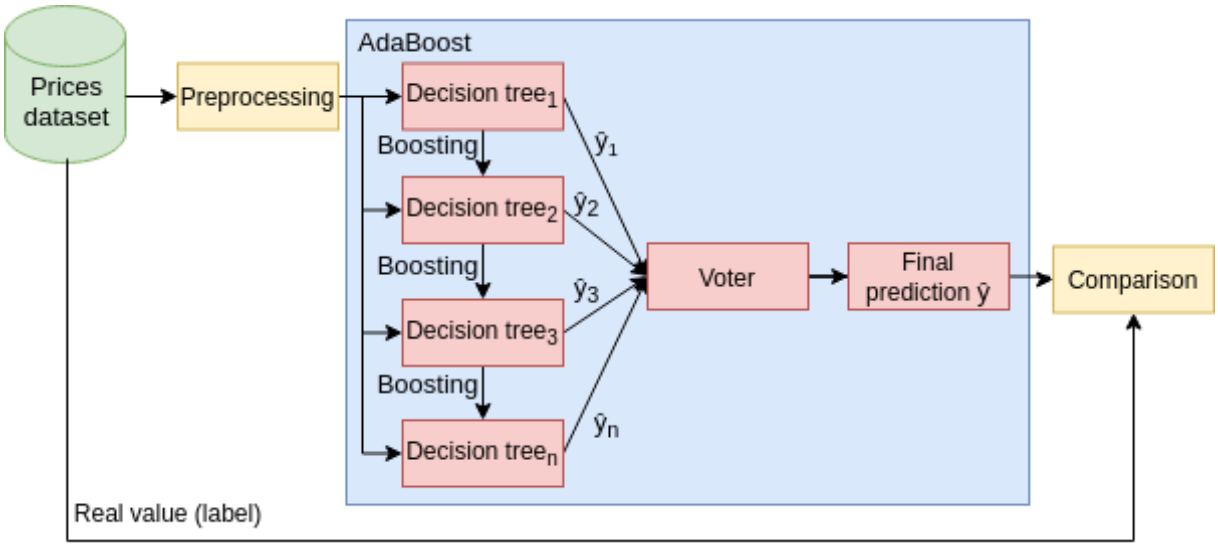
The other traditional ML model implemented was the AdaBoost model. It is an

Figure 3 – Example of application of SVR



Source: elaborated by the author, based on Boser, Guyon and Vapnik (1992), Drucker et al. (1997), Platt (1999), and Chang and Lin (2011).

Figure 4 – Main components of the AdaBoost model with decision trees for price prediction



Source: elaborated by the author, based on Ballings et al. (2015), Drucker (1997), Wang, Zhang and Verma (2015), and Freund and Schapire (1997).

ML model that uses weak learners (models that are not optimal for solving the specific problem) and the boosting method (in which the errors of one weak learner are used as an input for training the next weak learner) to generate predictions (BALLINGS et al., 2015; DRUCKER, 1997). Figure 4 illustrates this model's main components using decision trees as weak learners, as this was the model implemented in this work. First, the pre-processed data is fed to the first decision tree. Then, this model is trained, its errors are evaluated compared to the correct labels, and are used as inputs for the next decision tree (together with the initial price inputs). This process is repeated for the number of models that were defined initially, through the number of estimators hyperparameter (BALLINGS et al., 2015; DRUCKER, 1997).

After the final model is trained, the test subset is used to evaluate the AdaBoost model. Each trained weak learner (or estimator) provides a prediction, which is then weighted by a voting rule (also denominated as a voter or weighted voter), and the final prediction is provided (BALLINGS et al., 2015). This is then compared with the real values (or correct labels), and the final model error is calculated. Although this model is not normally used for price prediction tasks, it was chosen because the models used as the weak learners can be easily changed, providing an opportunity to use the boosting method on different models and architectures. The works by Ballings et al. (2015) and Freund and Schapire (1997) contain an in-depth description of the AdaBoost model's workings.

According to LeCun, Bengio and Hinton (2015), traditional ML techniques (such as SVM, decision trees, random forest, AdaBoost, among others) are limited in terms of raw data processing. It is necessary to handcraft the relevant features to be inserted in those models using expert knowledge. In DL's case, the models themselves identify and extract the data features, considering different abstraction levels (LECUN; BENGIO; HINTON, 2015).

DL is the application of ANNs with multiple hidden layers for learning different classification and regression tasks (LECUN; BENGIO; HINTON, 2015; ZHANG; WANG; LIU, 2018). In comparison to shallow neural networks (such as the perceptron or the MLP), DL architectures such as CNN and recurrent neural networks (RNN) can extract high-level features, improve pattern recognition, and deal with lower signal-to-ratio inputs and non-linearities (ZHANG; WANG; LIU, 2018). Some important vairations of the RNNs are: gated recurrent units (GRU), LSTM, and BiLSTM.

The work by LeCun, Bengio and Hinton (2015) contains an extensive review of DL's theoretical foundations. According to those authors, its main advantage is to allow the model to learn feature representations with multiple abstraction levels, which can then be used by the neural network or by other models to improve prediction and decision-making. The DL models learn abstract representations that are more likely to be invariant to local input data changes (SOHANGIR et al., 2018). For example, an object detection model using a CNN could detect an object in different positions and sizes if correctly trained. This is not observed on ML models.

Sohangir et al. (2018) point out that the resource-intensive nature of developing relevant handcrafted features through a feature engineering process is one of the main reasons to adopt DL models in relation to traditional ML models. This is even more important when the dynamics are unknown or complex, such as in the stock markets. LeCun, Bengio

and Hinton (2015) also state that DL has improved the state-of-the-art in several tasks, such as speech recognition, sentiment analysis, topic classification, object detection, drug discovery, genomics, among others.

ANNs consist of individual information processing units named neurons, organized in layers, and work sequentially to recognize patterns in the dataset and make predictions (ZHANG; WANG; LIU, 2018). It is vital to observe that these are supervised learning models and demand a considerably higher amount of information than traditional ML models, such as the SVM. A DL model is any ANN with multiple hidden layers.

For the sake of clarity, in this thesis, the term ANN was avoided, as it is too general. Term MLP was adopted for referring to ANNs with multiple layers and no specialized architecture. When referring to specific ANN architectures, their model names were used. For example, CNN refers to convolutional neural networks, which contain spatial-related components and convolutional layers. LSTM refers to RNNs that use LSTM layers (the neural network's temporal component). The term DL model was used when referring to all DL models.

All DL models contain a similar structure with three layers: (i) input layer, which receives the input vector; (ii) hidden layers, which aims at extracting features from the data and recognizing patterns and whose output is not visible; and (iii) the output layer, which presents the final output (a number in the case of regression tasks or a class for classification tasks) (ZHANG; WANG; LIU, 2018). Some DL models present a specific architecture, such as the CNN (explored in Section 2.4), which adds convolutional layers after the input layer.

As cited by Zhang, Wang and Liu (2018), the layers of the model closer to the input layer are used for learning simpler features. The layers that are closer to the output layer learn more complex features, denominated high-level features. The work by LeCun, Bengio and Hinton (2015) contains an illustration of this principle applied to several domains, including an in-depth example for facial recognition.

According to Freitas, Souza and Almeida (2009), the performance of the predictions of a DL model depends on aspects related to the network (topology and training methods, among others) and the data itself (features and noise, among others). Therefore, it is important to explore different aspects of the DL model related to its architecture and hyperparameters.

Ryll and Seidens (2019) conducted an extensive review of ML and DL models for the stock price and trend forecasting, considering more than 150 papers in the literature. The

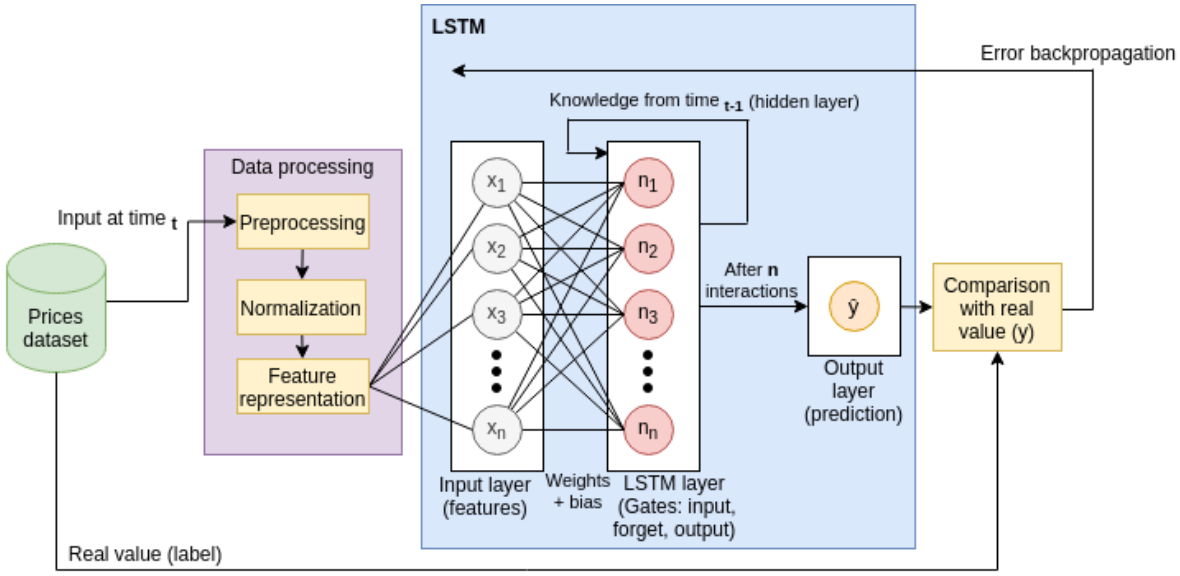
most critical points observed by the authors were: (i) ML and DL models tend to perform better than econometrics models; (ii) RNNs (especially the LSTM) tend to perform better than the MLP and the SVM; (iii) it is important to compare the results of DL models with the SVM, as several works obtained good results using it; (iv) there is a lack of studies that consider trading-related metrics, such as return metrics, which are vital to informing decision-makers better; (v) there is a difficulty in comparing different works in the literature due to the lack of standardized datasets and training and testing processes; and (vi) the BH strategy is an important baseline to evaluate trading systems. All those aspects were considered in designing the trading system and the experiments conducted in this thesis.

Lastly, it is crucial to observe that DL models can detect features and patterns in the data that might not be identified by traditional economics or econometrics models (SOUMA; VODENSKA; AOYAMA, 2019), and that DL models can provide important data for trading systems related to the market state from the noisy raw data observed (WANG et al., 2017).

The DL model implemented in this research for stock price prediction was the LSTM because it is considered state of the art in terms of time series prediction for several works on the financial domain (RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020). It can be defined as a DL model containing a temporal component, which identifies and evaluates temporal patterns in the data and uses them to predict future values (RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020). Therefore, it is a type of DL model that is well-suited for data with autocorrelation and a low signal-to-noise ratio, high complexity, and high volatility. Nevertheless, it is considerably more challenging to implement than other DL models such as the CNN or the MLP, and its high complexity results in a demand for more training data than other ML models, as described by (RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020).

The LSTM has two main forms concerning the input data: univariate and multivariate. When there is enough data available, the LSTM tends to present better results than the SVR, as was observed in the reviews by Ryll and Seidens (2019) and Sezer, Gudelek and Ozbayoglu (2020). Its main advantage in relation to the SVR and AdaBoost models for stock price prediction is its ability to capture temporal patterns, which is extremely important for data with autocorrelation. Its main advantage in relation to the ARIMA, SARIMA, and SARIMAX models is its capability to learn complex non-linear patterns and better deal with noise in the input data. Lastly, its main advantage in relation to all the cited models is that it can automatically extract features from the data. There-

Figure 5 – LSTM model and its main components for price prediction



Source: elaborated by the author, based on Chimmula and Zhang (2020), Zeroual et al. (2020), Sezer, Gudelek and Ozbayoglu (2020), Hochreiter and Schmidhuber (1997), and Karpathy, Johnson and Fei-Fei (2015).

fore, it does not demand data transformation (such as the differentiation for the ARIMA, SARIMA, and SARIMAX models) or manual feature engineering (an essential task for the SVR and AdaBoost models).

The LSTM model was proposed by Hochreiter and Schmidhuber (1997) and can be defined as a feedforward neural network that uses the output at time t as a recurrent input for subsequent time steps, allowing it to learn from current and past values data. It is used in several domains with state-of-the-art results, such as finance, speech recognition, correlation analysis, among others (RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020). Figure 5 illustrates the main components of the LSTM. These are the same for univariate and multivariate models, with the only difference being in the input data (the univariate model considers only one time series, while the multivariate can consider any number of time series).

As with all DL models, it is vital to pre-process the data and normalize it, so all features have values between 0 and 1. This accelerates the training process significantly and improves the model performance (JORDAN; MITCHELL, 2015; LECUN; BENGIO; HINTON, 2015; RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020). The feature representation step aims to transform the data to the desired window length (also denominated past history), which contains the past n days used for each prediction. This is a vital hyperparameter that has to be explored on implementing the LSTM model on

stock price prediction tasks because it can heavily influence the quality of the predictions.

The data is then sent to the model's input layer, symbolized by x in Figure 5. The number of inputs is related to the number of features in the array, with the most common being OHLCV data. Next, the LSTM layer neurons are fed with the input data. The number of neurons on the LSTM layer is a crucial hyperparameter, as it is directly connected with the complexity of the model and its ability to recognize complex patterns (LECUN; BENGIO; HINTON, 2015; JORDAN; MITCHELL, 2015; KOUTSOUKAS et al., 2017). The neurons on the LSTM layer contain three main gates that allow it to consider the impact of past data on the current prediction: (i) input, which receives the current data and the past hidden state and updates the hidden state; (ii) forget (which chooses if the new data received should overwrite past data); and (iii) output, which determines the hidden state value and sends information to the next neuron (SEZER; GUDELEK; OZBAYOGLU, 2020; HOCHREITER; SCHMIDHUBER, 1997; KARPATHY; JOHNSON; FEI-FEI, 2015).

Additionally, besides the current input at time t, the neurons also receive as input the knowledge from time t-1 (which is contained on the hidden layer). After n such interactions considering current and past data in an epoch, the model is trained. The output layer will then provide a prediction, which will be compared with the real value label, and the error will be propagated on the network, adjusting its weights (SEZER; GUDELEK; OZBAYOGLU, 2020; HOCHREITER; SCHMIDHUBER, 1997; KARPATHY; JOHNSON; FEI-FEI, 2015). As with other DL models, the backpropagation algorithm is used in the LSTM (SEZER; GUDELEK; OZBAYOGLU, 2020; HOCHREITER; SCHMIDHUBER, 1997; KARPATHY; JOHNSON; FEI-FEI, 2015). The works by Sezer, Gudelek and Ozbayoglu (2020), Hochreiter and Schmidhuber (1997) and Karpathy, Johnson and Fei-Fei (2015) contain an in-depth description of the LSTM model's workings.

The work of Persio and Honchar (2016) compared the use of MLP, CNN, LSTM, a combination of wavelets and CNN, and ensembles of those models to predict SP500 daily returns trend movements, considering two classes: upward and downward. The authors concluded that the best individual model was the CNN with wavelets (with an mean squared error or MSE of 0.249 and an accuracy of 53.60%) and the best overall model was a weighted ensemble of all the models (with an MSE of 0.226 and an accuracy of 56.90%). However, the authors did not: (i) conduct a statistical test to evaluate the significance of the results, which may be a problem because the LSTM and CNN presented very similar results for the MSE; (ii) consider the use of TIs; and (iii) compare the results with traditional econometrics models.

An important work on price prediction that considered a developing country stock market was conducted by Kara, Boyacioglu and Baykan (2011). The authors predicted daily stock market trends using an MLP and an SVM for the Istanbul Stock Exchange in this work. The asset considered was the ISE National 100 index, an important index for that market. The authors considered the use of ten TIs (simple and weighted 10-day moving average, momentum, stochastic K%, stochastic D%, Relative Strength Index (RSI), MACD, R%, A/D, and commodity channel index) as inputs for the models.

The main contributions of the work by Kara, Boyacioglu and Baykan (2011) were: (i) evaluating the use of different TIs for price prediction on developing markets; (ii) comparing a DL and a traditional ML model for stock price prediction; and (iii) conducting a statistical analysis of the results obtained. The authors observed that the MLP provided better accuracy than the SVM (75.74% versus 71.52% for the SVM). However, the authors did not explore: (i) the use of OHLCV data; (ii) additional features that may improve the prediction, such as market sentiment; (iii) a comparison with econometrics models; and (iv) the use of additional metrics that may provide further insights on the models' results, such as precision, recall, and F1-score.

Guresen, Kayakutlu and Daim (2011) evaluated the use of MLP, dynamic artificial neural networks (DAN2), and a generalized autoregressive conditional heteroskedasticity (GARCH) with MLP model to predict closing prices of the NASDAQ stock exchange index. The authors evaluated two metrics: MSE and mean absolute deviation (MAD), and concluded that the MLP model provided better returns than the DAN2 and GARCH-MLP models. However, the dataset used was considerably small, with only 36 days used as the testing subset. Also, the authors did not explore the use of additional features.

An important work on using LSTM for stock trend prediction was conducted by Fischer and Krauss (2018). In this work, the authors evaluated using several models (LSTM, MLP, logistic regression, and random forest) to predict price trends for the main stocks in the SP500 index from 1992 to 2015. Most importantly, the authors evaluated a simple rules-based trading strategy with the models' results, concluding that the LSTM provided the best results with a daily return of 0.26% after transaction costs (versus 0.04% for BH), a Sharpe ratio of 2.34 (versus 0.35 for BH), and a maximum drawdown of -18.17% (versus -54.67% for BH). However, the authors used univariate models with daily returns as their only feature. This could be improved by considering additional important features, TIs, among others. Additionally, the authors did not conduct an extensive hyperparameters analysis of the models. Lastly, the trading strategy was simplistic and could be improved by using methods such as RL.

The work by Mehtab, Sen and Dasgupta (2020) explored a comparison between univariate and multivariate CNN and LSTM models for predicting the opening price of an important company stock (Bharat Forge) in the Indian stock market from 2012 to 2015, considering intraday OHLCV data. This work is relevant for three main reasons: (i) conducting an in-depth comparison between the CNN and LSTM models for stock price prediction; (ii) exploring the use of DL models at a developing country's stock market; and (iii) conducting a five-step ahead forecasting.

Two metrics were evaluated by Mehtab, Sen and Dasgupta (2020): execution time and the ratio of RMSE to the mean open value in the test subset. Even though the CNNs presented slightly better values than the LSTMs, a statistical test is needed to evaluate if this difference is significant. Additionally, the authors did not conduct a hyperparameters analysis to fine-tune the models and did not explore additional features such as TIs or market sentiment. Lastly, the authors did not consider traditional baseline models for the comparison, such as the ARIMA models.

Eapen, Verma and Bein (2019) proposed a DL architecture that uses a CNN and a bi-directional LSTM for predicting stock market indices for the next seven days. The authors evaluated various pipelines for implementing the models, considering single and multiple pipelines, for predicting SP500 daily closing prices. The proposed model was compared with the SVR model. The multiple pipelines approach can be considered as an ensemble of the models' predictions. The authors concluded that the MSE for the LSTM was around 4.7 times lower than for the SVR. They also concluded that using a model with multiple pipelines resulted in an MSE around 9% lower than the single pipeline. In this thesis, a similar multiple pipelines approach for part of the models implemented in the M1 module was considered. However, the term "ensemble" was used because it is more commonly used in the literature.

Notwithstanding, the work by Eapen, Verma and Bein (2019) did not consider the use of additional features, such as TIs and market sentiment, as well as a comparison with econometrics models or other relevant metrics. Additionally, its description of the implementation is not extensive enough to allow for replication.

Yadav, Jha and Sharan (2020) explored using the LSTM model to predict four companies' closing prices in the Indian stock market (from 2008 to 2019), considering several hyperparameters values and numbers of hidden layers. Some important contributions from this work were: (i) evaluating the use of the LSTM on a developing stock market; (ii) conducting an in-depth analysis of the LSTM hyperparameters for stock price predic-

tion; and (iii) conducting a statistical analysis of the models' results considering several executions. One significant contribution that influenced the present thesis was that the LSTM using one LSTM layer provides the best results, with the lowest RMSE.

However, Yadav, Jha and Sharan (2020) did not consider in their analysis: (i) different batch sizes, which may impact significantly on the final results; (ii) a different number of epochs; and (iii) a traditionally used baseline for comparing the results of the different LSTMs, such as the ARIMA or SARIMAX models. Although it is possible to conclude which hyperparameter values are the most suitable among the evaluated ones, it is not possible to infer that the LSTM was better than other models for predicting the stock prices of the assets analyzed by the authors.

One of the most comprehensive works in terms of model evaluation for yearly stock trend prediction was conducted by Ballings et al. (2015). The models evaluated were: random forest, AdaBoost, kernel factory, MLP, logistic regression, SVM, and k-nearest neighbor. The authors evaluated the data from 5767 European stocks and used the area under the receiver operating curve (AUC) as the performance metric. The input data considered mostly fundamental analysis indicators and stock prices. The best models considering an average of the predictions for all stocks were: random forest, SVM, kernel factory, AdaBoost, MLP, k-nearest neighbors, and logistic regression. One of the possible explanations for the poor results of the MLP was the small training dataset used. However, the authors could have also implemented ARIMA and SARIMAX as baseline models. Additionally, the authors could have explored frequencies that are more typically used for trading, such as daily, weekly, or monthly.

Another important set of models are denominated ensemble models. These can be defined as the use of multiple models to provide a prediction (BALLINGS et al., 2015; RIBEIRO et al., 2020; DEAN et al., 2020). In theory, if the models are diverse, they can complement each other, improving the overall results compared to individual models (BALLINGS et al., 2015; RIBEIRO et al., 2020; DEAN et al., 2020). Ballings et al. (2015) emphasize the importance of considering ensembles and diverse ML models for comparing the prediction results.

There are several methods for creating ensemble models but, as this was not the main scope of this work, simple average ensemble models were implemented. In this case, the ensemble model's prediction is the simple average of each trained model's predictions. For example, the prediction of an ensemble of SVR and LSTM following this method would be the prediction of the trained SVR (after model fine-tuning) plus the prediction of the

trained LSTM, divided by two.

Kristjanpoller, Fadic and Minutolo (2014) proposed using an ensemble model that combined an MLP with a GARCH model to forecast the daily volatility of market indexes in Brazil, Chile, and Mexico, from 2000 to 2011. This is an interesting work because it uses DL for predicting volatility, while most works in the literature predict returns, prices, or price trends. The authors observed that their model provided a better MAPE than the baseline model for Brazil (60% versus 70% for the GARCH model), Chile (76% versus 91% for the GARCH model), and Mexico (76% versus 96% for the GARCH model).

Lastly, it is important to understand the state-of-the-art of stock price prediction in the Brazilian stock market, as this is a market that is much less explored than the stock markets from the USA, China, and others. Freitas, Souza and Almeida (2009) conducted one of the first works to apply an MLP for portfolio optimization in the Brazilian stock market. Their model predicted the stocks' weekly returns and then calculated a risk measure used to build a portfolio with complementary stocks. The 52 most traded stocks of the Ibovespa index were chosen as candidates for the portfolio and evaluated by the model. This work is also very relevant due to the depth of the statistical analysis that the authors conducted. The experiments conducted showed returns that were 292% above the baseline, the mean-variance model, and 78% above the BH strategy (considering the Ibovespa index).

Oliveira, Nobre and Zárate (2013) is an important early work to explore using DL models to predict stock prices in the Brazilian stock market. The authors predicted monthly closing prices for the PETR4 stock, one of the most traded assets in this market, using as inputs for an MLP: OHLCV data and several technical and fundamental analysis indicators. They have evaluated the use of different window sizes, obtaining the best results with a window size of 3, with a percentage of correct prediction direction of 93.62% on the test subset (versus 87.50% for the MLP with a time window equal to 1) and a MAPE of 5.45% (versus 6.41% for the MLP with time window equal to 1).

However, the work of Oliveira, Nobre and Zárate (2013) presents several points of improvement: (i) no traditional baseline was used (such as an ARIMA model), with the authors conducting a comparison between variations of only the MLP model; (ii) the monthly frequency is not commonly used for investment decision-making; and (iii) instead of using an MLP, the authors could have used a DL model architecture that is specific for predicting series with autocorrelation, such as the LSTM or GRU.

Nelson, Pereira and Oliveira (2017) evaluated intraday (15min) stock trend prediction

with the LSTM model, considering both OHLCV data and TI as inputs. Five important stocks of the Ibovespa index were evaluated from 2008 to 2015. The authors observed an average accuracy of 55.9% for the LSTM model, and statistical tests showed that it was significantly better than the MLP and random forest models' accuracies. One interesting contribution of this work was to use exponential smoothing to pre-process the data before inserting it into the model. Nevertheless, the evaluated period was too short (the test subset considered only the month of December 2014). A simple rule-based trading strategy was conducted at the end of the research, showing that using the LSTM outputs would lead to better results than the BH strategy for all the five stocks evaluated.

Pauli, Kleina and Bonat (2020) is one of the few works that compared different types of DL models for predicting closing prices in the Brazilian stock market. The authors evaluated the six most traded stocks during the initial stage of the Covid-19 pandemics (from March 2019 to April 2020). The models evaluated were: multiple linear regression, Elman networks, Jordan networks, radial basis function, and MLP. The evaluation of the RMSE for predicting the six stocks showed that the best models were the multilinear regression and the Elman networks. The worst models were the MLP and the use of the radial basis function. However, the authors did not consider: (i) the use of econometrics models; and (ii) the use of state-of-the-art DL models designed for dealing with temporal data, such as the LSTM. They also considered only one quality metric, when using multiple options (mean absolute error or MAE, mean absolute percentage error or MAPE, coefficient of determination or R2, among others) would allow for a better comprehension of the impacts of using the different models.

In the next section, the concept of sentiment analysis is explored and the main approaches for conducting stock market sentiment analysis, the primary data sources and models used, and the current challenges in this area.

2.3 Stock market sentiment analysis

According to Sohangir et al. (2018), market sentiment can be defined as the general attitude of the majority of investors (in terms of market impact) in relation to the market situation and the anticipation of price development for the whole market (or specific sectors or assets). The sentiment has several components, such as: assets' and sectors' situations, national and world events, history, economic reports, demand and supply, seasonal aspects, among others (SOHANGIR et al., 2018). A critical consideration for analyzing market sentiment is that, in real-life scenarios, investors' decision-making is influenced by

both the assets' price changes and the news (SOUMA; VODENSKA; AOYAMA, 2019). The sentiment prediction can then be used as a proxy for upward or downward trends on specific firms, sectors, or markets (MANSAR et al., 2017), or as a feature for ML-based models.

It is interesting to note that during the 1980s, Bondt and Thaler (1985) have already observed that market behavior can be influenced by news, which could be used to predict future movements better. Those authors' critical observation is that investors may present irrational responses to some news, overreacting in their decision-making following that news article. Some of the relevant biases that investors present are (NASSIRTOUSSI et al., 2014): overconfidence, overreaction, and information bias, among many others. These influence decision-making and are very hard to capture by traditional stock price prediction models. One crucial fact that underlines the existence of irrationality in the stock markets is speculative bubbles (which generally lead to financial crises that can be sector-specific or market-wide).

According to Mansar et al. (2017), texts such as social media messages and news can have important impacts on specific companies or the economy. This is because they contain opinions that may influence investors' decision-making. Souma, Vodenska and Aoyama (2019) describe that unstructured texts that have a qualitative nature, such as news, firms' reports, press releases, and government reports and announcements, provide critical information for decision-makers such as investors, firms, banks, traders, portfolio managers, among others. Additionally, the amount of news generated every day is much higher than the human capacity to process, evaluate, extract useful information, and use it for decision-making, generating demand for systems that can automatically extract market sentiment.

Neuenschwander et al. (2014) define the sentiment analysis task as labeling a set of texts into two classes: positive and negative. Nevertheless, other authors consider an additional neutral class. Bollen, Mao and Zeng (2011) point out that considering only two or three classes is not enough to address the complexity of sentiments and emotions and propose using more classes. Those authors considered the use of six classes with satisfactory results. In the financial context, the sentiment analysis task is related to converting unstructured text into meaningful information that stakeholders can use in decision-making (MANSAR et al., 2017).

According to Sohangir et al. (2018), sentiment analysis has been increasingly applied to infer users' sentiments, feelings, emotions, and moods from social media messages.

Their work has pointed out that DL is essential in this context because it is challenging to develop relevant features manually, and DL models can find important features themselves during the learning process, as explored in Section 2.2.

Among the many uses of sentiment analysis in the financial domain, Dridi, Atzeni and Recupero (2019) cite the following: market prediction, market sentiment prediction, box office prediction, predicting consumer's attitudes, analyzing certain venues' (such as blogs) towards specific companies, and improving decision-making for trading.

Nassirtoussi et al. (2014) cites that extracting sentiment from raw text related to financial news or posts depends on three main fields of study: linguistics, ML, and behavioral economics. In this thesis, the first two aspects are explored in depth. The third is explored indirectly by the DRL agent, as its structure allows it to recognize patterns on the price movements, inferring behavior in a data-driven form.

The work by Nassirtoussi et al. (2014) is one of the most comprehensive reviews on the use of natural language processing (NLP) and sentiment analysis for predicting market sentiment and its impact on stock price and stock price trend prediction. Even though some models were not yet proposed by the time it was conducted, such as DRL and the Global Vectors for Word Representation (GloVe) word embedding, this work is essential to understand better all the concepts related to sentiment analysis in the financial domain.

Sezer, Gudelek and Ozbayoglu (2020) conducted an extensive review of 140 papers published between 2005 and 2019 that evaluated the use of DL models for predicting stock prices and trends using different features as inputs. Some of the most relevant conclusions observed by those authors were: (i) higher prediction accuracy is not the same as a profitable model, due to the risks involved and the nature of decision-making on a trading system; (ii) RNN variations (GRU and LSTM) are the most used DL models in the most recent works; (iii) there is an increasing interest in using DRL for automated trading; (iv) there is an opportunity for using sentiment analysis of relevant texts (news and social media, among others) to provide market sentiment features to the DRL trading models; and (v) the use of ensembles considering both market sentiment and price prediction is an open issue in the literature.

Three main approaches are used for sentiment analysis (JOHNMAN; VANSTONE; GEPP, 2018; SOHANGIR et al., 2018; MEDHAT; HASSAN; KORASHY, 2014; NASSIRTOUSSI et al., 2014): (i) lexical-based, in which sentiment dictionaries (lists of words with corresponding values on important dimensions, usually sentiment value or polarity) are used to classify the input text; (ii) statistical learning or ML-based, in which ML and

DL models are used, generally with word embeddings or bag of words models, to classify the text; and (iii) hybrid, in which both lexicons and ML models are used. It is important to note that: (i) lexical-based systems are not able to learn; (ii) pure ML systems may miss on important knowledge that could be aggregated by using lexicons; and (iii) that hybrid systems with word embeddings and sentiment lexicons are the state-of-the-art on sentiment analysis on the financial domain (FERREIRA et al., 2019; MANSAR et al., 2017; NASSIRTOUSSI et al., 2014). Nevertheless, few works in the literature explore the use of different dictionaries and their impact in relation to fine-tuning the main models' hyperparameters. This is especially true for the Portuguese language. This aspect was explored in this work on the M2 module.

Johnman, Vanstone and Gepp (2018) describes the three main steps for sentiment analysis as: (i) feature extraction, in which the features (words, n-grams, part of speech tags, named entities, among others) are defined and extracted; (ii) feature representation, in which the features are converted into numbers (term frequency-inverse document frequency or TF-IDF, word embeddings, bag of words, among others); and (iii) sentiment classification, which is related to processing the features and providing the sentiment class or number. It is important to observe that the sentiment classification can be done in three different levels (JOHNMAN; VANSTONE; GEPP, 2018): (i) document-level; (ii) sentence-level; and (iii) aspect-level.

The most important data source for sentiment analysis on financial markets is specific news providers that are considered relevant and trustworthy by investors (NASSIRTOUSSI et al., 2014; JOHNMAN; VANSTONE; GEPP, 2018). Therefore, this was the data source chosen in this thesis for the M2 module. The use of financial news instead of general news is recommended due to its higher signal-to-noise ratio (for example, there are fewer irrelevant news articles in newspapers and websites that focus on the financial domain versus general-purpose newspapers and websites). Also, news headlines or news titles may provide even less noise due to the necessity to summarize the news article's most essential points in a few words. Hu et al. (2018) emphasizes that the sentiment analysis quality is directly related to the data inputs' quality. For this reason, using specific financial news may provide better results than general news articles.

According to Nassirtoussi et al. (2014), the time it takes from the release of a specific news article and its impact on the stock market prices (and the duration of this impact) may vary from seconds to months, depending on a series of complex factors. The most common time frames evaluated are: 15min, 20min, 1h, 2h, and 3h (NASSIRTOUSSI et al., 2014). In this thesis, the aim was to capture the daily impacts, as this is the most

common trading frequency in the literature. Additionally, this time frame is not typically explored in the literature for sentiment analysis, contributing to a better understanding of news impacts on daily trading.

One of the critical steps in sentiment analysis is choosing the pre-processing techniques used to prepare the data to be fed to the ML model (NASSIRTOUSSI et al., 2014). Some of the main pre-processing techniques are: feature-selection, word embeddings, stemming, conversion to lowercase, stopwords removal, and numbers removal (NASSIRTOUSSI et al., 2014; PENNINGTON; SOCHER; MANNING, 2014). All of those techniques are used in this research.

Hu et al. (2018) describes the lack of literature that explores the use of financial news articles for improving stock prediction, considering aspects related to quality, trustworthiness, and comprehensiveness. All of those aspects are considered in this research.

Weng et al. (2018) developed an expert system to predict short-term stock prices using two components: (i) a knowledge base built with five types of inputs: stock prices, TIs, sentiment scores of news articles, Google search trends, and relevant Wikipedia pages for the asset; and (ii) an ML model that will use the data and make the predictions for the next 1 to 10 days. Four models were evaluated: MLP, SVR, boosted regression tree, and random forest. Experiments with data from 2013 to 2016 for the Citigroup stock ($C) showed that the next day's prediction presented the lowest MAPE (1.50%). The best ML model on the authors' experiments with 19 stocks was the boosted regression tree.

However, several aspects could be analyzed to improve the model proposed by Weng et al. (2018): (i) considering state-of-the-art DL models for price prediction, such as the LSTM; (ii) implementing a simple trading strategy to evaluate the results in comparison to the BH strategy; and (iii) evaluating the use of word embeddings and dictionaries to improve the quality of the sentiment prediction. Additionally, it is not clear how the sentiment score was obtained, as the authors used a dataset provided by a resource that is not open and did not describe what model or technique was used to calculate the daily sentiment score.

Jin, Yang and Liu (2020) proposed a methodology for using decomposed prices (using empirical modal decomposition) and sentiments (extracted from posts from the Stocktwits microblog and comments from Yahoo Finance) as inputs for an LSTM with an attention component. The CNN was used to extract the sentiments from 96,903 posts, using a word2vec word embedding, and calculate a daily sentiment index. The daily sentiment index indicated both if the sentiment on that day was bullish (positive) or bearish (nega-

tive) and its magnitude (higher numbers indicated more positive or negative sentiment). Based on the analysis of the experiments that were conducted with the Apple (AAPL) stock, the authors observed that their model provided a considerable improvement in terms of MAPE (1.65% versus 4.58% for the vanilla LSTM), MAE (2.396 versus 7.032 for the vanilla LSTM), RMSE (3.197 versus 8.712 for the vanilla LSTM), R2 (97.74% versus 83.20% for the vanilla LSTM), and accuracy (70.56% versus 60.12% for the vanilla LSTM).

Nevertheless, several improvements could be made on the model proposed by Jin, Yang and Liu (2020), such as: (i) incorporating newer word embeddings, such as GloVe; (ii) conducting an extensive hyperparameters analysis of the CNN and LSTM models; (iii) considering TIs and other features besides the sentiment and OHLCV data; (iv) evaluating econometrics models as baselines (instead of a vanilla LSTM); and (v) evaluating the model's performance on a trading task.

Lastly, it is essential to note that very few works explore sentiment analysis in Portuguese, especially in the financial domain. According to Neuenschwander et al. (2014), the resources for NLP and sentiment analysis in Portuguese are very scarce compared to the number of resources available for the English language. Additionally, Medeiros and Borges (2019) cite that few works in the literature focus on sentiment analysis on the Brazilian stock market. One of the reasons that may explain this behavior is the reduced number of available sentiment lexicons, word embeddings, and resources in Portuguese compared to the English language.

One important related work was conducted by Medeiros and Borges (2019), focusing on the ML approach for sentiment analysis in the financial domain. The authors have implemented the SVM and random forest models and principal component analysis and t-stochastic neighbor embedding to predict sentiment classes based on 4516 tweets related to the Ibovespa market index in Portuguese. They have considered a multi-label sentiment classification task with nine classes (joy, trust, fear, surprise, sadness, disgust, anger, and anticipation), and observed that, in general, the SVM presented a better score than the random forest model. However, this work could have benefited from using specific lexicons and DL models with word embeddings, adding to the model's semantic and syntactic knowledge.

The following section contains an in-depth exploration of the use of hybrid models for analyzing and predicting the stock market sentiment, as these models are more relevant for the scope of this thesis than the other approaches described in this section (lexicon-based

and ML-based).

2.4 Hybrid models for stock market sentiment analysis

Zhang, Wang and Liu (2018) has conducted an extensive survey on DL for sentiment analysis, describing the main techniques, models, and state-of-the-art works in the literature on several domains. Among their most interesting findings, it is possible to observe that: (i) automated sentiment analysis is a necessity because of the vast volume and variety of information being generated; (ii) CNN and RNN are state-of-the-art models for this task, even when used in the sentence-level with individual words; (iii) the use of word embeddings is essential to improve the overall results, as they already contain semantic and syntactic information; (iv) most of the literature use three categories for sentiment polarity (positive, neutral, and negative); (v) lexicons can provide critical information for improving the model's results (using the hybrid approach); (vi) this task is currently one of the most important research areas in NLP; and (vii) some of the main factors that resulted in the increased interest on sentiment analysis are: new DL models and new data sources (such as social media, forums, and blogs). Based on this analysis, it is possible to observe the hybrid approach's importance for sentiment analysis.

Additionally, Neuenschwander et al. (2014) cite that the hybrid approach could result in classifiers that are less dependent on the context and better generalize their results on different tasks. Nevertheless, this is not the most used approach for sentiment analysis in the financial domain, with most recent papers focusing on the ML-based approach using DL models.

First, the lexicon component of the hybrid approach will be explored, followed by the ML-based component. Loughran and Mcdonald (2011) developed a critical financial sentiment lexicon or dictionary based on reports from the US Security and Exchange Commission, composed of six dimensions: positive, negative, litigious, uncertainty, model strong, and model weak. As shown by those authors, the six dimensions increase the quality of the market sentiment prediction in relation to general sentiment lexicons such as the Harvard IV-4. This dictionary can improve the quality of the prediction of DL models, as was done in the work by Ferreira et al. (2019). Those authors have observed that using this dictionary improved significantly the models' results in relation to using no dictionary at all. However, the Loughran-McDonald dictionary is only available in English.

No similar finance-specific dictionary exists for the Portuguese language. Therefore, the use of three relevant Portuguese general sentiment dictionaries was explored in this work: Sentilex (SILVA; CARVALHO; SARMENTO, 2012), OpLexicon (SOUZA; VIEIRA, 2012), and WordNetAffectBR (PASQUALOTTI; VIEIRA, 2008). However, the model structure that was implemented in the M2 module allows for easily changing the dictionary used.

One critical development that led to exciting results on the use of DL models for sentiment analysis was word embeddings. These can be used together with lexicons (as was done in this thesis) to improve the model's semantics knowledge. This is essential because few works use semantics features for sentiment analysis in the financial domain (DRIDI; ATZENI; RECUPERO, 2019). In this thesis, word embeddings were used as feature representations to partially address this concern.

Word embeddings can be characterized as unsupervised word representations extracted from large text corpora with multiple dimensions (MANSAR et al., 2017). Due to the high dimensionality and size of the corpora used for training, word embeddings can incorporate semantic relationships (instead of just syntactic ones obtained by using other ML-based models).

According to Sohangir et al. (2018) and Pennington, Socher and Manning (2014), word embeddings aim to create a vector representation of each word with a lower dimension space in comparison to the traditional bag of words model. According to Zhang, Wang and Liu (2018), each vector's dimension represents a feature of a specific word, and the vectors may contain important information related to both syntax and semantics. The original words are mapped to a new multidimensional space, mapping similar words to similar distances (JIN; YANG; LIU, 2020).

The GloVe word embedding (PENNINGTON; SOCHER; MANNING, 2014) is considered a state-of-the-art approach for sentiment analysis, being used on the financial domain by Mansar et al. (2017), Ferreira et al. (2019), among others. For this reason, a Portuguese version of the GloVe word embedding (HARTMANN et al., 2017) was used in the M2 module to improve the models' prior knowledge on semantics.

The following paragraphs describe the DL models that were implemented in the M2 module for market sentiment analysis as the ML component of the hybrid approach used. Two models were implemented: (i) the MLP, a traditional baseline model for sentiment analysis tasks; and (ii) the CNN, a state-of-the-art model for sentiment analysis tasks. The implementations followed the main concepts and techniques used by Ferreira et al.

(2019).

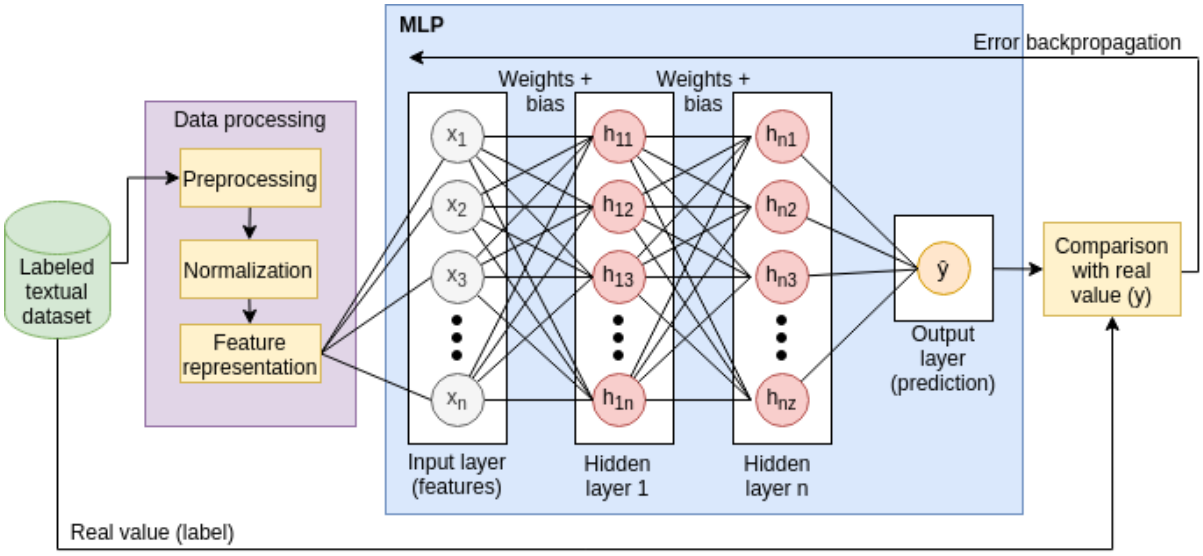
The MLP is the most common type of DL model, being considered an extension of the perceptron model (JORDAN; MITCHELL, 2015; NELSON; PEREIRA; OLIVEIRA, 2017; LECUN; BENGIO; HINTON, 2015). It was initially inspired by the human brain's functioning and how information is stored in the connections between the neurons (JORDAN; MITCHELL, 2015; NELSON; PEREIRA; OLIVEIRA, 2017; LECUN; BENGIO; HINTON, 2015). One of its main characteristics is its capability of recognizing complex non-linear patterns (JORDAN; MITCHELL, 2015; NELSON; PEREIRA; OLIVEIRA, 2017; LECUN; BENGIO; HINTON, 2015). Nevertheless, in some problems, it may: (i) present overfitting problems; (ii) be unable to detect specific patterns (such as autocorrelation); or (iii) be considerably slow when training with large datasets. Those factors have motivated the development of several modifications on the original MLP, changing its components and introducing new building blocks, to adapt ANNs to deal with temporal data (with RNNs), spatial data (with CNNs), among others.

It includes three types of layers: input, hidden, and output. The input layer receives the features of each data point according to the feature representation used. It is important to note that, as is the case of all DL models, it is essential to pre-process, normalize, and provide the model with the correct feature representation to improve its performance (JORDAN; MITCHELL, 2015; LECUN; BENGIO; HINTON, 2015). As was described in the last section, there are several feature representation options, but word embeddings were used in this research due to their promising results in the literature (MANSAR et al., 2017; FERREIRA et al., 2019).

The main components and processes necessary to use an MLP are illustrated in Figure 6. It is important to note that the "knowledge" of this DL model is captured on the weights that connect the different neurons on the different layers. The error backpropagation will change those weights based on the size of the error and the learning rate, allowing the system to learn while the model is being trained (JORDAN; MITCHELL, 2015; LECUN; BENGIO; HINTON, 2015). Among its main hyperparameters, the following can be highlighted: learning rate, number of neurons in each hidden layer, activation functions, number of hidden layers, and optimization algorithm (JORDAN; MITCHELL, 2015; LECUN; BENGIO; HINTON, 2015; KOUTSOUKAS et al., 2017; ZHANG; WALLACE, 2015). The works by Jordan and Mitchell (2015) and LeCun, Bengio and Hinton (2015) contain an in-depth description of the MLP model's workings.

Bollen, Mao and Zeng (2011) conducted one of the most cited works on sentiment

Figure 6 – Main processes for using an MLP for regression sentiment analysis



Source: elaborated by the author, based on Loukas et al. (2017) and LeCun, Bengio and Hinton (2015).

analysis for stock market sentiment prediction. They have used a self-organizing fuzzy neural network and data from tweets (9,853,498 tweets) and two mood tracking tools (OpinionFinder and Google-Profile of Mood States or GPOMS) to predict stock price trends of the Dow Jones Industrial Index (DJIA) in 2008. The proposed model achieves an accuracy of 86.7% and a MAPE of 1.83%. One of the most important contributions of this work was to consider several dimensions for calculating market sentiment: calm, alert, sure, vital, kind, and happy.

Nevertheless, several improvements could be made on the work by Bollen, Mao and Zeng (2011): (i) considering a more extensive testing period, as the original paper considered a test subset of only 18 days (described by the authors as having been chosen due to low volatility and absence of significant socio-cultural events); (ii) implementing a trading system or simple trading rules to verify the impacts of the proposed model versus the BH strategy; (iii) considering newer methods that were not available at the time of this research, such as word embeddings; (iv) considering the use of a hybrid approach, with a sentiment lexicon to improve the accuracy of the predicted sentiments; and (v) considering other important models for predicting stock prices, such as ARIMA, SARIMAX, and the LSTM. Additionally, one aspect that makes it impossible to replicate this work is that the GPOMS tool is no longer available.

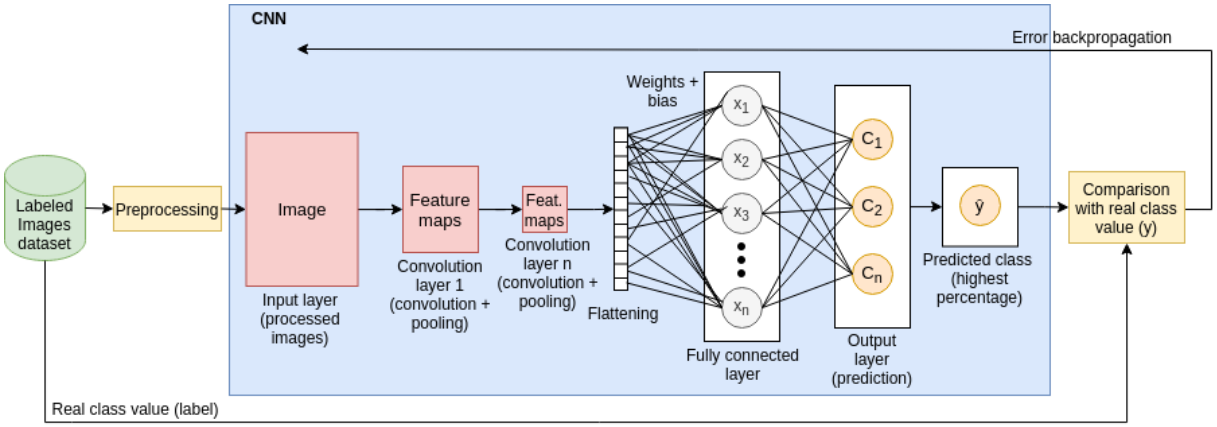
Dridi, Atzeni and Recupero (2019) have proposed a model that uses lexical, semantic, and hybrid features to predict the market sentiment using a range between -1 (bearish)

and +1 (bullish), based on financial microblogs and news headlines from the SemEval 2017 task 5. They have evaluated several models, such as random forest, linear regression, lasso, ridge regression, SVR, and clustering methods, using ten-fold cross-validation. They have observed that the SVR model with lexical and semantic features obtained the best results in terms of cosine similarity for both the news headlines (0.655 versus 0.563 for the random forest model with only lexical features) and the microblogs posts (0.726 versus 0.680 for the random forest model with only lexical features) datasets. Although their results were significant, the authors did not explore word embeddings and DL models, which could considerably improve their results.

According to Zhang, Wang and Liu (2018), Sohangir et al. (2018), Zhang and Wallace (2015), and LeCun, Bengio and Hinton (2015), CNNs are a special type of DL model developed for computer vision tasks, which involve images and videos. Its main difference with the MLP is the existence of multiple convolutional layers, which contain convolutions, subsampling, and pooling operations to extract local features independent of their positions. For that reason, it is used for all kinds of tasks that present spatial correlation (ZHANG; WALLACE, 2015; ZHANG; WANG; LIU, 2018). Its design was inspired by mammals' visual system (MNIH et al., 2015) and it was proposed by Lecun and Bengio LeCun et al. (1998). It is considered a state-of-the-art architecture for several domains, including object detection, facial recognition, among other computer vision tasks (SOHANGIR et al., 2018; MNIH et al., 2015; LECUN; BENGIO; HINTON, 2015; ZHANG; WALLACE, 2015). Figure 7 illustrates its main components. CNNs have several uses for NLP, such as: sentiment analysis, tagging, named entity recognition, entity search, sentence modeling, summarization, topic modeling, among others (SOHANGIR et al., 2018; ZHANG; WALLACE, 2015).

One crucial aspect to consider is that the CNN contains three operations that are different from the MLP: (i) the use of convolutions, which aim at hierarchically detecting features, generating feature maps; (ii) the use of pooling (typically, max-pooling), which compresses the data and allow for spatial invariance (the ability of the CNN to detect objects of different sizes and in different positions); and (iii) the use of flattening, which prepares the data to enter on the model's fully connected layer (LECUN et al., 1998; LECUN; BENGIO; HINTON, 2015; ZHANG; WALLACE, 2015). These operations allow for: pattern detection on different parts of an image and hierarchical-feature detection (LECUN et al., 1998; LECUN; BENGIO; HINTON, 2015; ZHANG; WALLACE, 2015; MNIH et al., 2015). Additionally, it is possible to implement CNNs for classification (the most common use, in which the model predicts the probability of the detect object

Figure 7 – Main processes for using a CNN for image classification



Source: elaborated by the author, based on LeCun et al. (1998), LeCun, Bengio and Hinton (2015), and Zhang and Wallace (2015).

belonging to any of the possible classes) and regression (in which the model predicts a number based on the features of the data point) tasks. It was implemented for a sentiment analysis regression task in this work, following the works by Mansar et al. (2017) and Ferreira et al. (2019). The use of regression instead of classification allows for a more fine-grained identification of the sentiment contained in a specific sentence (FERREIRA et al., 2019).

Four critical aspects that should be observed regarding the CNN are: (i) there are several variations of CNN implementations, according to different objectives; (ii) it is possible to have multiple convolutions and pooling layers; (iii) the pre-processing stage is one of the most critical steps for the CNN, especially in NLP uses such as sentiment analysis; and (iv) the CNN can be used for n-dimensional data, by using n-dimensional convolutions (SOHANGIR et al., 2018; LECUN; BENGIO; HINTON, 2015; ZHANG; WALLACE, 2015). In the case of sentiment analysis, 1d convolutions were used in this work, following the works by Zhang and Wallace (2015), Mansar et al. (2017), and Ferreira et al. (2019). The works by Zhang and Wallace (2015) and LeCun et al. (1998) provide a further description of the workings of the CNN model.

Sohangir et al. (2018) implemented and evaluated different configurations of LSTM and CNN to analyze and predict sentiments based on messages from the StockTwits social media. They have used logistic regression and doc2vec as baseline models and have observed that CNN presented the best results in terms of accuracy, precision, recall, F1, and AUC. It is important to note that they compared the predicted sentiment and its evolution compared to stock trends, but the model did not consider the price itself as an input.

Mansar et al. (2017) proposed the model that won the SemEval task 5 subtask 2 challenge, which used financial news headlines to analyze the sentiments toward specific firms, with a cosine similarity of 0.745. The authors proposed a hybrid approach. It was based on using the DepecheMood sentiment lexicon (which contains nine dimensions), the GloVe word embedding, the VADER sentiment analysis tool, and a CNN with one convolutional layer to extract sentiment from the news headlines. They have also observed better generalization and reduced problems with overfitting by using the GloVe word embedding (in relation to using no word embedding at all).

Souma, Vodenska and Aoyama (2019) evaluated the use of several configurations of LSTMs for news sentiment analysis on the Dow Jones Industrial Average (DJIA30) on a different setting: using intraday stock price returns one minute after each news article to label the news as positive or negative. They have used the GloVe word embedding trained with the Wikipedia 2014 and Gigaword 5 corpora, and their dataset was extracted from the Thompson Reuters News Archive from 2003 to 2013. This is a very interesting work because the stock market results provided the labels of the news dataset. Nevertheless, there are several drawbacks: (i) the prediction accuracy on the test set was around 50%; (ii) it is difficult to analyze if the movement that was observed was related to a specific news release or not; (iii) the authors did not explore the impact of using dictionaries to improve their predictions; (iv) an extensive hyperparameters analysis was not conducted; and (v) the authors did not implement a trading system to evaluate the results of the use of their model. In this thesis, all these aspects were considered.

The work by Ferreira et al. (2019), which is directly connected with the M2 module of this thesis, evaluated the use of SVR and CNN for sentiment analysis on the financial domain, using the SemEval 2017 task 5 subtask 2 dataset. This dataset was composed of 1,142 labeled financial news headlines with sentiments on a continuous scale from -1 (very negative) to +1 (very positive). Different feature representations and the impact of the main hyperparameters for both models were analyzed in depth. A stratified 5 fold split was implemented to maintain the data balanced across the splits, and CNN was implemented using the GloVe word embedding and three options of dictionaries (no dictionary, the lexicon from the VADER tool, and the Loughran-McDonald domain-specific dictionary). The main results of the experiments conducted were: (i) the CNN with the GloVe word embedding and no dictionary provided the best MSE (0.094 versus 0.108 for the SVR with TF-IDF); (ii) the use of more dimensions on the word embedding resulted in better performance; (iii) the use of dictionaries did not bring a significant performance improvement; and (iv) the best CNN model led to slightly better results than the best

SVR model.

However, Ferreira et al. (2019) focused on the English language and did not consider the model's accuracy in predicting market prices or price trends. Additionally, it is not possible to conclude that using the best model proposed in that work will lead to better trading results, as no statistical analysis was conducted.

As was already explored during this section and the last one, few works in the literature explore the use of sentiment analysis to predict the stock market sentiment in Portuguese. One of the first important sentiment analysis works for the Brazilian stock market considering data in Portuguese was the one by Neuenschwander et al. (2014). In this work, the authors explore the use of data extracted from two labeled sources: 922 tweets and 373 news articles collected in 2013. After pre-processing the data (accent and punctuation removal, lemmatization, stemming, lower casing, and stopwords removal), the data was used on three classifiers on the following approaches: (i) a lexicon-based model using the SOCAL method and the SentiLexPT dictionary; and (ii) two ML models: naive-Bayes and naive-Bayes multinomial. The naive-Bayes multinomial model presented the best results for sentiment analysis on the news articles dataset, with an F1 of 0.778 (versus 0.527 for the best SOCAL configuration). Therefore, they have concluded that the ML approaches may provide better results for sentiment analysis related to financial news in Portuguese (NEUENSCHWANDER et al., 2014).

Nevertheless, the authors did not consider the use of: (i) state-of-the-art DL models; (ii) the use of word embeddings; (iii) the use of other dictionaries in Portuguese; and (iv) the use of the hybrid approach, which could improve the results obtained by the individual approaches. Additionally, as the data was collected in a short period (less than six months), there is a considerable probability that it was related to the same market trend, making it difficult to generalize the results.

Gildo, Júnior and Marinho (2018) explored the use of sentiment analysis of 471,430 economic news articles in Portuguese (from 2000 to 2015 from several important newspapers) and ML models for predicting price trends for several sectors on the Brazilian stock market. The models used for prediction were: decision trees, random forest, extra trees, adaptive boosting, and gradient boosting. They have observed that the news impacted differently on the different sectors, concluding that some sectors are more susceptible to news than others and that the gradient boosting provided the best results, with an accuracy higher than 70% for most scenarios. However, it is vital to note that no state-of-the-art models (such as CNN) were used in the work by Gildo, Júnior and

Marinho (2018). Also, no word embeddings were used.

It is crucial to observe that Gildo, Júnior and Marinho (2018) used a different method than the one adopted in this research. Instead of using a word embedding in Portuguese as in this thesis, Gildo, Júnior and Marinho (2018) translated the news to English and used the VADER sentiment classification tool, a lexicon-based approach (HUTTO; GILBERT, 2014). The authors have also aggregated the news sentiment scores for a given period using a simple average of all news sentiments in that period, as is done in this thesis.

The following section describes the main aspects of stock trading and algorithmic trading and the importance of considering the trading metrics to evaluate trading strategies (instead of considering traditional ML metrics).

2.5 Stock trading

According to Freitas, Souza and Almeida (2009), investors' most important concern is related to the expected future returns and the associated risks of their trading strategies, not to the accuracy of the predicted prices or trends. Therefore, trading metrics and aspects must also be considered. Some of the most important metrics for trading stocks are: annualized and cumulative returns, the Sharpe ratio, annual volatility, and maximum drawdown. These are described in the following paragraphs.

According to Lei et al. (2020), annualized returns are the return of an investment over one year. Cumulative return is the return of the investment over the studied period. In both cases, higher values indicate better strategies. Nevertheless, it is also essential to consider risks and losses related metrics to evaluate a trading strategy. Those authors define the Sharpe ratio as a standardized indicator that considers both risks and returns of a specific strategy. It is one of the most important metrics to evaluate trading strategies, and it is calculated by subtracting the annualized risk-free rate from the annual returns and then dividing the total by the annual volatility (YANG et al., 2020). Greater values of the Sharpe ratio indicate better overall decisions in terms of risks and returns.

According to Wang et al. (2019), the annual volatility is used to measure a trading strategy's average risk during one year. It can be defined as the standard deviation of the portfolio returns over one year (YANG et al., 2020). Together with the stability, which is calculated as the R-squared of a linear fit of the cumulative log returns[1], these are important measures of risk. Conegundes and Pereira (2020) define the drawdown as

[1] https://github.com/quantopian/pyfolio

the total percentage loss of the trading system's capital before it starts winning again, considering the daily closing prices. Therefore, it can be thought of as the bottom of the valley in terms of losses before the portfolio starts to observe a positive return. The maximum drawdown is the most negative return experience in the period (or the bottom of the lowest valley), and it is a proxy used to represent the risk of different trading strategies and assets (CONEGUNDES; PEREIRA, 2020).

Martinez et al. (2009) state that choosing what strategies to use to make investment decisions based on stock price predictions is vital to investors. Besides having good predictions, investors also need a good decision-making model that can result in profits (ideally, with low risk), considering important metrics such as those introduced in the last paragraphs.

Algorithmic trading is one option for addressing this concern. Algorithmic trading can be defined as the use of software for trading (generally referred to as trading robots or robotic trading agents), typically in an automated fashion (NASSIRTOUSSI et al., 2014; WU et al., 2019; EAPEN; VERMA; BEIN, 2019; TRELEAVEN; GALAS; LALCHAND, 2013). This usually depends on: (i) fast decision-making; (ii) daily or intraday trading; and (iii) use of price prediction models (NASSIRTOUSSI et al., 2014; WU et al., 2019; EAPEN; VERMA; BEIN, 2019; TRELEAVEN; GALAS; LALCHAND, 2013). According to Treleaven, Galas and Lalchand (2013), algorithmic trading was responsible for around 73% of the volume traded in the USA in 2011.

Some of the main benefits of using automated trading systems are (EAPEN; VERMA; BEIN, 2019; TRELEAVEN; GALAS; LALCHAND, 2013): (i) better response times to market changes, being able to respond faster; (ii) more accurate trading operations, being able to obtain more returns; and (iii) reduced risk of loss due to repeated or mistaken operations, which may happen in systems that are based on human decision-making.

According to Wu et al. (2019), one of the main characteristics of algorithmic trading is that it employs mathematical and learning models to analyze and automate stock trading without the need to incorporate rules or knowledge from financial theory. Treleaven, Galas and Lalchand (2013) conducted an in-depth exploration of the field of algorithmic trading, presenting its central concepts, theoretical foundations, and models used. According to those authors, algorithmic trading can be defined as any form of trading that uses algorithms to automate all or part of the trading cycle. Therefore, it is possible to observe that this is a very general term, including all types of automation on trading systems. In this thesis, it is used to refer specifically to the use of AI models (ML, DL, RL, or DRL)

to automate feature extraction, price prediction, or trading execution tasks.

Treleaven, Galas and Lalchand (2013) classify the main tasks of algorithmic trading as: (i) pre-trade analysis, including exploratory and statistical data analysis; (ii) signal generation, including feature extraction or generation of handcrafted features; (iii) trade execution; (iv) post-trade analysis, including evaluation of financial metrics; (v) risk management; and (vi) asset allocation. This work focused on items i (on the exploratory data analysis), ii (by generating features with the M1 and M2 modules), iii (by automating trade execution using the MT module), and iv (by using several relevant financial metrics to evaluate the trade results). As the DRL agent's reward function on the MT module is directly related to the returns of the actions executed, it addresses the risk management task. Lastly, the asset allocation was not be considered in this work, as the scope was to develop and evaluate the trading system for an individual asset.

One commonly used baseline for algorithmic trading models and strategies is the BH strategy. It can be described as buying stocks at the beginning of the period and selling them at its end, without intermediate transactions. According to Dantas and Silva (2018), the BH strategy is a challenging baseline strategy to overcome, with several works in the literature (such as the one by Dantas and Silva (2018)) not obtaining better returns in the studied periods. Thus, the BH strategy was considered the baseline model for the trading system proposed in this thesis.

Wang et al. (2017) characterize the approaches for algorithmic trading in two groups: (i) knowledge-based methods, in which expert knowledge is used as features or to make decisions; and (ii) ML-based, which are data-driven and make automated decisions. This work focused on the second approach. Nevertheless, the analysis of additional data other than OHLCV data was conducted, as this was observed by Li, Rao and Shi (2018) and Dantas and Silva (2018) to be very important in terms of impacts of the models' results.

Li, Rao and Shi (2018) proposed a trading strategy that considered both stock prices and sentiment analysis of news using an SVM for prediction and specific trading rules. They have observed that the use of sentiment extract from firm-specific news articles may improve the trading strategy, that general market sentiment can impact decision-making, and that the size of the impact on the price may vary depending on the company's characteristics and the content of the news article. They have evaluated both the directional accuracy of the predictions and the RMSE. They have observed that some sectors present better accuracies (more than 60%), such as: information technology, social service, and wholesale and retail trade. However, it is important to note that: (i) state-of-the-art

models should be evaluated, as several works have proven that they perform better than the SVM for sentiment analysis and price prediction tasks; and (ii) the trading system proposed by Li, Rao and Shi (2018) is not able to learn with time, as is the case of the system proposed in this thesis.

Hu et al. (2018) proposed a hybrid attention network (HAN) to predict stock price trends considering prices and texts from news articles as inputs. Their experiments considered 425,250 news articles from 2014 to 2017 and 2,527 stocks, creating a market sentiment time series for each stock. Then, the author introduced a rule-based system with the following rule: (i) at the end of each trading day, obtain the probability of upward and downward trends on the next trading day for each stock; (ii) calculate the final probability as the upward trend probability minus the downward trend probability for each stock; (iii) rank the stocks from the highest to the lowest final probabilities; and (iv) invest evenly on the top K stocks on the market opening the next day. The K variable was considered a hyperparameter that was evaluated with different values.

Hu et al. (2018) has observed both the best accuracy and the best profits (0.611 annualized return for K=40 stocks, while the BH strategy result was 0.04) for the HAN model. Nevertheless, the trading strategy used was very simplistic and could be improved by using an ML model, which could learn and improve over time. They have also observed that the RNN architectures provided better results than the MLP and RF ones, and the HAN was the best model.

Martinez et al. (2009) was one of the first works to propose a stock trading system using DL models for the Brazilian stock market. The authors used an MLP to predict daily high and low prices and a rule-based system for decision-making. This work's critical contribution is to consider both a traditional ML metric (MAPE) and a trading-specific metric (annualized return).

The trading system proposed by Martinez et al. (2009) was tested with two of this market's most important stocks: PETR4 and VALE5. Four models were compared with the MLP: 1-day lag and the simple moving averages for 5, 10, and 20 days. The main inputs considered were: open, high, low, and close prices, the exponential moving average for five days of each of those prices, the Bolling bands for each of those prices, and the opening price of the current day. The MLP showed a MAPE that was 50% smaller than the best baseline model (20 day simple moving average), and the trading strategy doubled the invested capital. However, the authors did not explore the use of more elaborate econometrics models for price prediction, such as the ARIMA, nor the use of other ML

and DL models.

According to Johnman, Vanstone and Gepp (2018), ETFs have become popular as trading assets because they reduce the complexity of trading a market index. For example, instead of trading each of the 100 companies on the FTSE100 index, the authors used an ETF that highly correlates with that index. In that way, the investor only needs to trade one asset instead of a portfolio of a 100 assets.

One of the few works that consider the sentiment dimension on trading strategies using ETFs that represent market indices is the one by Johnman, Vanstone and Gepp (2018). This thesis used the same strategy, trading the BOVA11 ETF as a proxy to the Ibovespa index.

Lastly, one problem of decision-making considering price prediction models is that those do not consider transaction costs and the impact of the previous actions (LI; RAO; SHI, 2018). Therefore, the models do not learn with mistakes made in terms of trading decisions made. Dantas and Silva (2018) also point out that trading systems depend directly on the sequences of interdependent trading decisions. Therefore, sequential decision-making models such as RL or DRL could form the basis of an automated trading system, as those models can learn from experiences and from its predictions regarding its states and actions.

The following section describes the main steps and concepts related to implementing those models based on the concepts described in this section. It also contains the description of the Deep deterministic policy gradient (DDPG) and Proximal policy optimization (PPO) models implemented in this work and the state-of-the-art literature on RL and DRL use for automated stock trading.

2.6 RL and DRL for stock trading

Conegundes and Pereira (2020) described that the typical use of ML in stock trading is to provide predictions (of prices or trends), which are then used on a rules-based system for trading the stocks. The use of DRL could automate decision-making, identifying the critical rules that define a good trading strategy in a data-driven way (CONEGUNDES; PEREIRA, 2020). Additionally, supervised learning models do not incorporate essential market aspects such as liquidity and transaction costs (CONEGUNDES; PEREIRA, 2020).

According to Fischer (2018), some of the most important benefits of using RL and

DRL for stock trading are: (i) trade automation, by incorporating aspects of market prediction and decision-making, while other ML models depend on rule-based systems to execute trades; (ii) the possibility to consider important market aspects such as transaction costs, market liquidity, and risk-aversion of investors; (iii) the possibility of using different functions to be optimized (the reward functions); and (iv) considering the impact of previous actions on the current decision-making.

There are several definitions for RL which are relevant. These are explored in the following paragraphs. RL can be defined as the group of AI models and techniques designed and used for sequential decision-making (FRANÇOIS-LAVET et al., 2018). Those models are applied for decision-making in all kinds of domains, from healthcare to robotics and stock trading. François-lavet et al. (2018) provided a straightforward definition: RL is related to the use of agents that make decisions in an environment and seek to maximize their cumulative rewards (the sum of immediate and expected future rewards). The RL models can also be seen as trial-and-error methods, as the agents have to experiment with different actions for the various states in a given problem to find the best actions for each state. In this sense, the best strategy is related to the one that maximizes the cumulative rewards. According to François-lavet et al. (2018), the agent does not need to know the environment beforehand, it just needs to interact with the environment a considerable number of times and collect information and learn from it.

Vázquez-canteli and Nagy (2019) define RL as a group of agent-based AI models in which the individual agents learn the optimal policy that will dictate their interaction with the environment, aiming to maximize a reward function. The strategy used for choosing the actions is also called a policy. Li, Rao and Shi (2018) define RL as the process that the agent uses to map which action should be taken in each state to maximize a reward function. In this sense, the agent must learn which actions are the best for each state, in a deterministic or stochastic form (depending on the MDP and the model adopted to solve the problem).

François-lavet et al. (2018) describe the RL problem as a discrete-time stochastic control process that follows a set of steps (independently on the RL or DRL model chosen): (i) the agent starts at timestep t0 and state s0, gathering the first observation w0; (ii) the agent chooses the action a0 for that state (using a specific policy) from all the available actions; (iii) the agent obtains the immediate reward r1; (iv) the agent updates its policy (in different ways, depending on the model used); (v) the state transitions to s1; and (vi) the agent obtains the observation w1 from the environment.

Some of the main concepts related to RL models are: agent, environment, state space, action space, reward function, timestep, value, policy, and MDP. All of these concepts are described in the following paragraphs, except for the MDP (which is explored in-depth in section 4.2), following the definitions by François-lavet et al. (2018), Vázquez-canteli and Nagy (2019), Li (2018), and Fischer (2018).

The agent will interact with the environment and make actions, depending on its states, to maximize the cumulative rewards. The environment is related to all the factors outside of the agent in the specific problem context. It is responsible for generating and providing information to the state's agent, the possible actions, and the rewards. Additionally, the environment's dynamics also define the state transition probabilities, and it can be static or change with time. In the stock market's specific case, the environment's dynamics are unknown, and the influences and their impacts are constantly evolving.

The state space is related to all possible states that an agent can be in on a given RL problem. Similarly, the action space is related to all the possible actions the RL agent can take. It is vital to observe that some states may restrain the actions the RL agent can take. For example, in the trading domain, even if some aspects of the state may be advantageous for the agent (such as a stock being considerably undervalued at a specific period), others may restrict its actions (such as the model having no money left to invest at that specific period).

The reward function is one of the main components of the RL agent, and its design must consider critical aspects related to the problem's context and the desired objectives. As explored throughout this chapter, the reward function on trading problems can take many forms: the immediate reward from one period to the next, the overall reward in a given interval, the Sharpe-ratio on a given interval, among others.

The agent's policy is the mapping between the actions and states, and it is crucial to maximizing the reward function. In other words, it specifies which action should be taken in each state. It is updated as the agent learns what actions result in the best rewards (considering both immediate and future rewards). It is treated as a probability because the agent does not always need to choose the best action (what would characterize a greedy policy with no exploration), especially if it explores the environment.

According to Wu et al. (2019), the application of DRL (the combination of RL with DL models for learning) in the field of algorithmic trading is an important area that needs to be further explored. Li, Rao and Shi (2018) observed that the DL model can provide the agent with automated feature extraction, an essential aspect for interacting with

complex environments. According to Sajad, Schukat and Howley (2016) DRL models' main contribution is to learn relevant feature representations on problems with high-dimensional data inputs. This can also be expanded for problems with high-dimensional action or state spaces and problems with complex environment dynamics, such as the stock markets. However, Meng and Khushi (2019) and Fischer (2018) observe that the literature on algorithmic trading using DRL lacks comparisons of different agents under similar conditions and data sources, as is done in the current research.

According to Mnih et al. (2015), RL agents (without the use of DL) can be used in real-life scenarios only in contexts in which: (i) relevant features can be handcrafted; (ii) the domain can be fully observed; or (iii) in low-dimensional state and action spaces. As the algorithmic trading domain is characterized by having unknown dynamics, continuous state spaces, and doubts about the quality of handcrafted features (such as the discussions related to the effectiveness of TIs), the DRL approach may result in better performance of the trading system.

The work by Mnih et al. (2015) is one of the most important works in the RL field because it implemented the first working version of a DRL model, the deep Q network (DQN). This is an adaptation of the Q learning method with a continuous state space, being more adapted for real-life scenarios. It also introduced the use of experience replay (also called replay memory), which aims to remove correlations in the sequence of observations and train using mini-batches, improving the model's performance. The authors explored its use on different Atari games, considering an end-to-end implementation (the model obtained all the information for decision-making from the games' pixels and chose the best actions to maximize the game score). This is also an essential work to better understand the MDP framework in an DRL context, and Mnih et al. (2015) have observed that the model achieved more than 75% of the human score on 29 of the 43 games evaluated.

Wang et al. (2017) propose using DQN for trading in a model called Deep Q Trading. The authors observed better results in relation to the BH strategy and the recurrent reinforcement learning agent (RRL) in experiments conducted from 2001 to 2015 with the HSI (a Hong Kong market index) and the SP500 indexes. For the HSI, the Deep Q Trading model observed a cumulative return of 350% (versus 154% for BH and 174 for RRL), a Sharpe ratio of 0.59 (versus 0.28 for BH and 0.89 for RRL), and a maximum drawdown of 42% (versus 65% for BH and 55% for RRL). For the SP500, it presented a cumulative return of 214% (versus 169% for BH and 141% for RRL), a Sharpe ratio of 0.45 (versus 0.34 for BH and 1.23 for RRL), and a maximum drawdown of 31% (versus

57% for BH and 43% for RRL)

Nevertheless, the model proposed by Wang et al. (2017) only uses the daily closing price as an input. The present thesis considered OHLCV data, TIs, and market sentiment and price predictions as additional features to the DRL model.

According to Li, Rao and Shi (2018, 2018), Vázquez-canteli and Nagy (2019), and François-lavet et al. (2018) there are two main methods to solve RL problems: value-based and policy-based. The first focuses on using the Bellman equation to calculate the values of expected future rewards for each state, and this is used to make decisions (LI; RAO; SHI, 2018, 2018; VÁZQUEZ-CANTELI; NAGY, 2019; FRANÇOIS-LAVET et al., 2018). The traditional value-based models are Q learning and Sarsa. Policy-based methods focus on learning a policy (or decision-making strategy) to select actions without using the value function directly for selecting the actions (LI; RAO; SHI, 2018). One important policy-based method in the financial domain is the RRL method. An alternative that is explored in the present research is the actor-critic method, which considers both the value function and the policy gradient on decision-making.

According to Wang et al. (2019), one problem of using value-based DRL models for stock trading is the stock market is too complex for those models. The solution, according to those authors, is to consider policy-based approaches. Yang et al. (2020) cite another critical limitation: the state and action spaces must be discrete and finite, which is not normally observed in real-life stock trading scenarios.

One of the most important works to evaluate RL and DRL's use on stock trading is the extensive literature review conducted by Fischer (2018). In this work, the author explores the history of RL's use in the financial domain and its main models and theoretical foundations. The classification used between critic-only, actor-only, and actor-critic methods applied to stock trading is essential to understand better the advantages and potential benefits of using each model.

Fischer (2018) has observed that the critic-only approach is the most common in the stock trading context. In this approach, the agent learns the impacts of different actions directly through the value function. According to Fischer (2018), its main advantages are: (i) flexibility in terms of the MDP design; (ii) ease of implementation of different possible reward functions; and (iii) the possibility to implement direct and interpretable considerations related to immediate and future rewards, by exploring the use of the discount factor. However, its main setbacks are the limited action and state spaces and the lack of convergence in different situations.

According to Fischer (2018), the actor-only approach is related to the search for the policy (the mapping from states to actions) that leads to the maximum expected rewards is the agent's objective. Fischer (2018) states that the main advantages of this approach are: (i) being applicable to continuous action spaces; (ii) having faster convergence; and (iii) being more transparent in relation to the reasons behind the decision-making. Fischer (2018) observed that those models' main disadvantage is the need for a differentiable reward function, making them less flexible in design and applications.

Lastly, Fischer (2018) cites that the actor-critic approach, which combines the advantages of both approaches, is the least studied in this domain. In this case, the actor network determines the actions and explores different policies, while the critic network evaluates the quality of those actions and provides the actor network with quality-related feedback. The actor network then adjusts its policy. The main advantages of using this approach are (FISCHER, 2018): (i) adjusting the policy parameters gradually; and (ii) the possibility of application on problems with high-dimensional or continuous action and state spaces. However, these are more complex to implement and are explored in few works in the literature. Due to its characteristics and the lack of works exploring this approach, this thesis focused on implementing two actor-critic state-of-the-art models, DDPG and PPO. Yang et al. (2020) cite that using the actor-critic approach can bring important results for stock market trading, especially on portfolios with multiple stocks.

Li (2018) conducted a thorough review of the main concepts, models, and techniques for DRL agents. Among the main models explored are: (i) value function-related models, such as Q Learning, DQN, and general and distributional value function variations; and (ii) policy-related models, such as vanilla policy gradient, actor-critic methods (including DDPG, Trust Region Policy Optimization or TRPO, and PPO), and policy gradient with off-policy learning. The author also explores several uses of DRL in different domains, such as: games, robotics, NLP, computer vision, finance, business management, healthcare, education, energy, transportation, among others.

Arulkumaran et al. (2017) conducted an extensive DRL review, considering several domains and agent models. The authors highlight the importance of DRL for developing autonomous systems that consider high-dimensional state and action spaces. This is the case of algorithmic trading, making those models suitable for being components of an autonomous trading system. Arulkumaran et al. (2017) also highlight DRL models can obtain satisfactory results in several domains, unlike RL models, which may present problems due to memory complexity (for example, storing all values in contexts with high-dimensional state or action spaces), computational complexity (for example, due

to the necessity of exploring the state-action combinations), and sample complexity (for example, in choosing unbalanced samples from experience)

Meng and Khushi (2019) conducted an extensive literature review on RL and DRL's use in financial markets, exploring relevant works, models, and concepts. The authors cite that RL's use for stock trading is in the early development stage and demands further research on several topics to be considered a reliable group of models for trading. Nevertheless, several works have observed better results (considering returns, profit, or Sharpe ratio) than the baseline models.

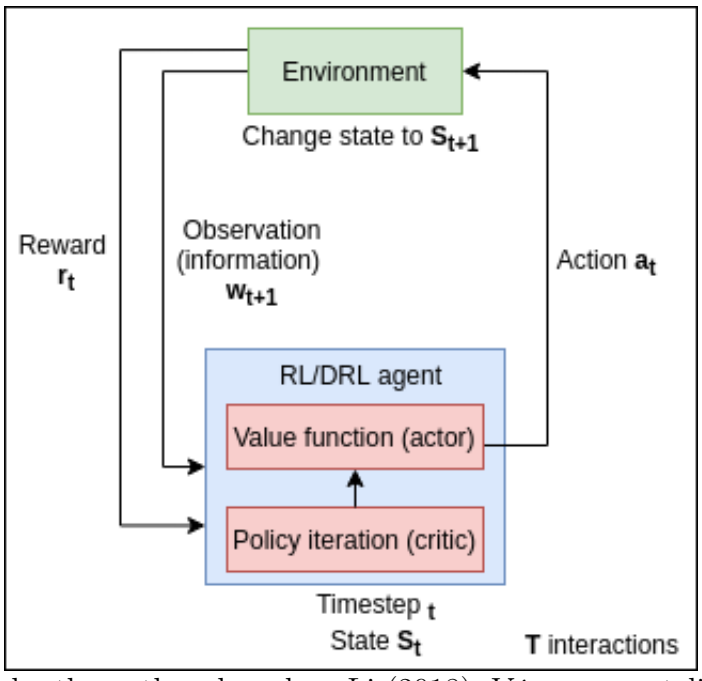
Among the main problems of the works evaluated by Meng and Khushi (2019) were: (i) not considering transaction costs, which may have significant impacts on the trading strategy; (ii) not considering liquidity issues, which may affect specific assets that are not traded with high volumes; and (iii) not considering bid and ask spreads, which may impact on defining the final trading price when live trading. The current thesis considered the transaction order costs (which is an environment hyperparameter) and the trading of the BOVA11 ETF, which is highly liquid. Lastly, the bid and ask prices were not considered as the work is related to daily trading and not intraday trading, although this could be adapted on the MT module.

It is important to note that RL and DRL models are characterized in two types of categories: (i) related to the model's prior knowledge of the environment (model-free and model-based); and (ii) related to the model's policy (on and off-policy). These are briefly described in the following paragraph.

In model-based implementations, the agent has prior knowledge of the environment and its dynamics and uses that knowledge for decision-making (VÁZQUEZ-CANTELI; NAGY, 2019; LI, 2018; FRANÇOIS-LAVET et al., 2018). In model-free implementations, which are closer to real-life situations, the agent must learn to associate the optimal actions for each specific state without prior knowledge of the transition probabilities (VÁZQUEZ-CANTELI; NAGY, 2019; LI, 2018; FRANÇOIS-LAVET et al., 2018). Due to the unknown dynamics of the stock market, the present dissertation considered the use of model-free agents. As the stock market dynamics are unknown, there is considerable noise in the data, and the future market states are unknown, Conegundes and Pereira (2020) emphasize that the use of model-free RL is a viable alternative for tackling the problem of stock market trading.

Figure 8 illustrates the RL and DRL models' general components, describing specific components of the actor-critic methods (the actor and the critic networks, in red). Never-

Figure 8 – Illustration of general actor-critic models with a RL or DRL agent



Source: elaborated by the author, based on Li (2018), Vázquez-canteli and Nagy (2019), and François-lavet et al. (2018).

theless, if the RL or DRL agent is not an actor-critic agent, its high-level working is very similar (with differences related to which of the components is used for decision-making: the value function or the policy iteration). As the DRL models' development is more recent and these are starting to be explored in the financial domain (FISCHER, 2018; MENG; KHUSHI, 2019), their description will be conducted with more depth than the past models.

According to Li (2018) and Sutton and Barto (2018), the actor-critic models learn a policy (on the actor network) and a value function (on the critic network). The critic network results are used to update the state on the actor network, which was observed to reduce variance and lower the time to convergence. Vázquez-canteli and Nagy (2019) describe these methods as having separate memory structures to store the state-action space and the state-value space.

The basic working of a DRL agent is the following (illustrated in Figure 8):

1. At state S_t (in which t represents the timestep, a crucial element on DRL models), the DRL agent receives its first observation, wt, from the environment. This is a set of information (features from the dataset) used to calculate its state. In the case of the financial domain, these could be OHLCV data, for example;

2. As the agent did not perform any action at the first timestep, it does not receive a reward;

3. Both the actor and the critic networks are initiated with random values or a specified distribution. Typically, the uniform distribution is used. If it is a transfer learning task, with previously learned weights);

4. The critic network conducts the policy iteration process, in which it will update the value function and send the actor network the Q value that will be used as a basis for decision-making;

5. The actor network uses the Q value as an input and chooses the action to be conducted (this usually involves a degree of randomness) to avoid being stuck in local optima). The DRL agent then conducts action a_t;

6. The environment receives the action at and, through interactions with its dynamics, changes to a new state, S_{t+1};

7. The environment sends to the DRL agent the new observation, w_{t+1}, and reward r_{t+1}, and this cycle continues for T interactions (also called timesteps);

8. The cycle repeats until a training mini-batch is completed. Then, the weights in the actor and critic networks are updated based on the networks' losses, and the steps continue until the ending of the training stage (which is defined by a specific hyperparameter).

It is vital to observe that the number of timesteps is one of the most important hyperparameters for DRL agents, as it will dictate how many interactions the agent will have with the environment, influencing its pattern recognition ability (LIU et al., 2020; YANG et al., 2020). It is also critical to observe that several other components help to improve the model's results on real-life scenarios, such as: (i) experience replay, in which the agent will store the last n tuples of reward, action, and states; (ii) mini-batch training, in which the agent will train using multiple data points (instead of updating the weights of the network with each data point, a time consuming and non-optimal training process); (iii) restraints on the policy update (as in the PPO model), which allow it to change gradually; and (iv) the use of ensembles of DRL agents (a field of study that is considerably new and has not been explored on the financial domain) (LIU et al., 2020; YANG et al., 2020; LI, 2018; FRANÇOIS-LAVET et al., 2018; VÁZQUEZ-CANTELI; NAGY, 2019; FISCHER, 2018; SAJAD; SCHUKAT; HOWLEY, 2016).

This research considered the first three items with two state-of-the-art models (DDPG and PPO), but the last one was out of this work's scope. The works by Li (2018), François-lavet et al. (2018), Vázquez-canteli and Nagy (2019), Fischer (2018), and Sajad, Schukat and Howley (2016) contain a thorough review of different algorithms, architectural aspects, and modeling techniques used for RL and DRL.

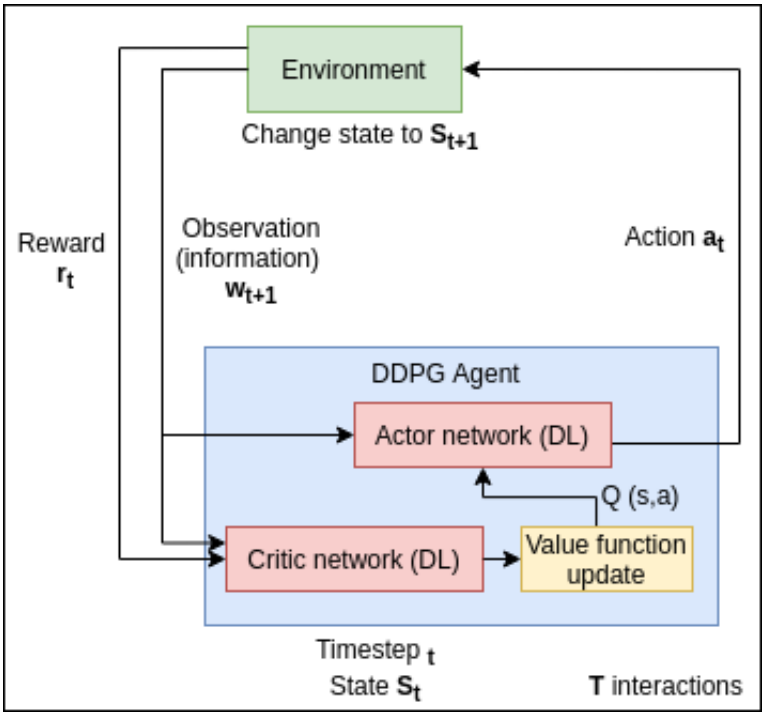
Another important categorization of RL and DRL models is between on-policy and off-policy agents. According to Vázquez-canteli and Nagy (2019) and François-lavet et al. (2018), on-policy agents (such as the Sarsa or the PPO models) have faster convergence but do not learn as well from historical data. They learn from taking actions using the current policy. The off-policy agents (such as Q-learning and DDPG) take longer to converge because they explore more actions on the environment. In general, those models consider separate policies for taking action (actor policy) and reward estimation and actions evaluation (critic policy). In other words, the actor policy proposes the action that should be taken in a specific state s (based on probability), and the critic policy predicts if that action will result in a positive or negative immediate reward for that state. Both are updated during the learning process. In this research, both on-policy (PPO) and off-policy (DDPG) state-of-the-art agents are evaluated.

Both DDPG and PPO are actor-critic based methods. However, besides their implementation and structural aspects, they present two significant differences: (i) PPO is on-policy, while DDPG is off-policy, which results in faster convergence for PPO and better capability to learn from historical data for DDPG; and (ii) DDPG was designed considering continuous state-action spaces, while PPO was designed considering both discrete and continuous state-action spaces. Therefore, the DDPG model is better suited for continuous action spaces, which would better capture the different possible actions on trading strategies.

In cases with high-dimensional state or action spaces, it is essential to use DRL models (VÁZQUEZ-CANTELI; NAGY, 2019), as these use DL for mapping the states to actions. The alternative would be to implement a Q table (a matrix containing all states and actions), but this would present several problems, such as the need to discretize values and maintain a vast, sparsely populated matrix.

The DDPG agent belongs to the actor-critic group of DRL models. It is a development proposed by Lillicrap et al. (2016) to encompass problems with high-dimensional state and action spaces, and it has observed considerably good results on complex robotics problems (LILLICRAP et al., 2016). It is an extension of the deterministic policy gradient or DPG

Figure 9 – Illustration of the DDPG model and its main components



Source: elaborated by the author, based on Li (2018), François-lavet et al. (2018), Lillicrap et al. (2016), Guo et al. (2020), Liessner et al. (2018), and Vázquez-canteli and Nagy (2019).

method, developed by Silver et al. (2014). Figure 9 illustrates the main components of this model.

As in other actor-critic models, the DDPG agent contains two networks: (i) the critic network, which aims at calculating and updating the value function and providing the actor network with the Q value for each state; and (ii) the actor network, which aims to learn the best action to take in each state (considering past states and actions), and executing that action in the environment (LI, 2018; FRANÇOIS-LAVET et al., 2018; GUO et al., 2020; LIESSNER et al., 2018; LILLICRAP et al., 2016). Unlike other models, the actor network of the DDPG agent does not receive the reward. This is only received by the critic network. Also, it is important to observe that both networks have their own targets and loss functions, updating the values of their weights after each mini-batch interaction (LI, 2018; FRANÇOIS-LAVET et al., 2018; GUO et al., 2020; LIESSNER et al., 2018; LILLICRAP et al., 2016). The DDPG agent also uses experience replay to retain information from past actions, states, and rewards. The actor contains the policy that maps states to actions, while the critic contains the value function estimates. The critic updates its values based on the experience replay and sampling memory, allowing for faster learning (LI, 2018; FRANÇOIS-LAVET et al., 2018; GUO et al., 2020; LIESSNER

et al., 2018; LILLICRAP et al., 2016).

The DDPG methods use deep neural networks as the actor and critic networks and the error backpropagation algorithm for updating all the networks' weights (LI, 2018). The DDPG model can be considered as a mix of the DPG (SILVER et al., 2014) and the DQN (MNIH et al., 2015) DRL agents. The main contributions of the DDPG model are: (i) allowing for better learning in comparison to the DQN by extending it to continuous action and state spaces; (ii) implementing the actor and critic networks on the DQN DRL model, allowing for better learning in complex scenarios; (iii) making the DPG model's learning more stable and robust, due to the use of experience replay and target networks; and (iv) using batch normalization to avoid problems due to possible different feature value ranges (LI, 2018; LILLICRAP et al., 2016). Additionally, the model addresses the exploration-exploitation dilemma by adding noises to the actor network (LI, 2018; LILLICRAP et al., 2016). Lillicrap et al. (2016) implemented the DDPG model on several simulated tasks, obtaining better results than the DQN in most tasks and a considerably lower time to convergence.

The basic working of the DDPG agent is the following (illustrated in Figure 9):

1. At state S_t, the DDPG agent receives its first observation, w_t, from the environment;
2. As the agent did not perform any action at the first timestep, it does not receive a reward;
3. Both the actor and the critic networks are initiated with an uniform distribution;
4. The critic network receives the observation wt and the reward r_t and conducts the policy iteration process, updating the value function and sending the actor network the Q value;
5. The actor network uses the Q value as an input and chooses the action to be conducted in that state. The DRL agent then conducts action a_t;
6. The environment receives the action at and changes to a new state, S_{t+1};
7. The environment sends: (i) to the actor network, the observation w_{t+1}; and (ii) to the critic network, the observation w_{t+1}, and the reward r_t. This cycle continues for T interactions or timesteps;
8. The cycle repeats until a training mini-batch is completed. Then, the weights in the actor and critic networks are updated based on the networks' losses, and the steps continue until the ending of the training stage.

The works by François-lavet et al. (2018), Li (2018), Vázquez-canteli and Nagy (2019), and Lillicrap et al. (2016) contain an in-depth description of the DDPG model's workings, along with its mathematical formulation.

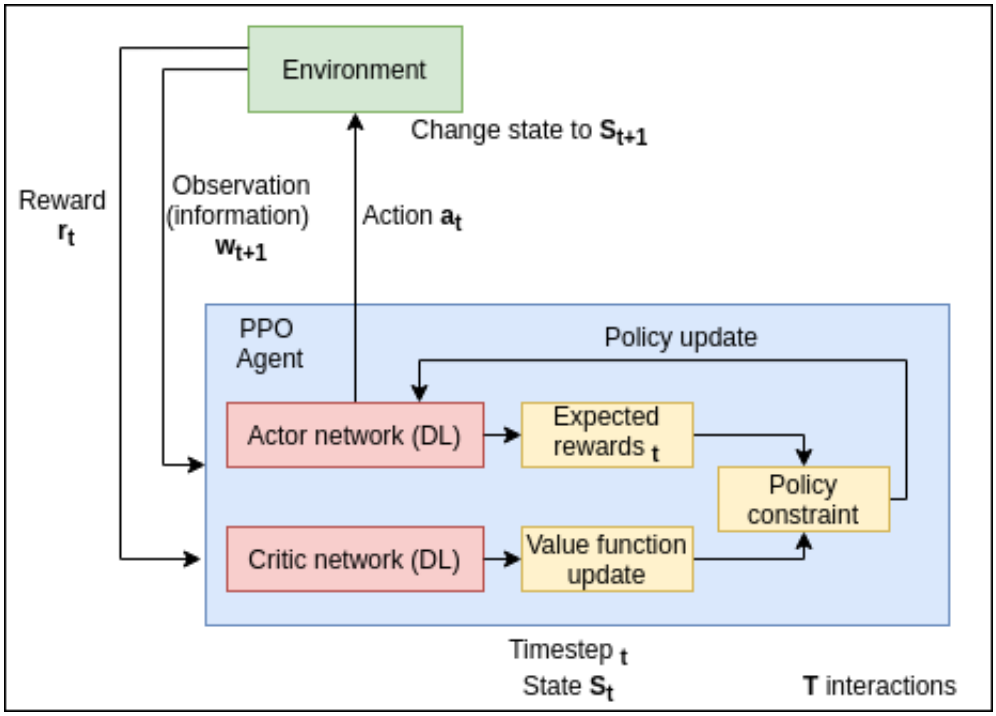
Besides the DDPG, the PPO model was another state-of-the-art model implemented in this thesis. The following paragraphs describe the main aspects of this model. The PPO model is derived from the trust-region policy optimization (TRPO) model, a vital policy gradient method that uses restraints to change the policy within defined boundaries and seek improvements (FRANÇOIS-LAVET et al., 2018; VÁZQUEZ-CANTELI; NAGY, 2019). Therefore, the policy updates' size is restrained by different constraints, limiting the model's options. This leads to faster convergence, as updates outside of the constrained boundaries are not considered (in other words, values that are predicted to lead to bad policies are not considered by the model).

As stated above, the PPO is a variation of the TRPO, which uses a different constraint to penalize bad policy changes, considering their predicted impacts on the reward function. The PPO presents several advantages compared to the TRPO DRL agent: faster learning, better generalization, and more straightforward implementation and analysis (LI, 2018; SCHULMAN et al., 2017). Due to its good results on different physics and robotics simulation tasks and its ease of use, it is used as a benchmark on the OpenAI Baselines library[2]. This model uses a constraint function to improve the policy parameters update (called a surrogate objective function) by limiting its value ranges and data sampling from the environment (LI, 2018; SCHULMAN et al., 2017; VÁZQUEZ-CANTELI; NAGY, 2019). Therefore, the model presents a fast convergence and avoids choosing parameter values that would significantly worsen policies. According to Schulman et al. (2017), the PPO DRL agent can be considered scalable in terms of task variety, data-efficiency, and robustness on tasks and environments with different dynamics and complexity.

The PPO agent also belongs to the actor-critic group of DRL models. It was proposed by Schulman et al. (2017) to encompass problems with high-dimensional state and action spaces, but considering incremental policy updates (LI, 2018; FRANÇOIS-LAVET et al., 2018; SCHULMAN et al., 2017; VÁZQUEZ-CANTELI; NAGY, 2019; LIM et al., 2020). As in the DDPG agent, the PPO actor and critic networks have the main basic functions. Nevertheless, there are several differences between this model and the DDPG DRL agent, but the most significant ones are: (i) both actor and critic networks receive as inputs the observations and the rewards; (ii) a policy constraint is used to update the policy, based on the actor and the critic networks' results; and (iii) the critic network does not

[2]https://openai.com/blog/openai-baselines-ppo/

Figure 10 – Illustration of the PPO model and its main components



Source: elaborated by the author, based on Li (2018), François-lavet et al. (2018), Lim et al. (2020), and Schulman et al. (2017).

directly provide the Q value to the actor network (LI, 2018; FRANÇOIS-LAVET et al., 2018; SCHULMAN et al., 2017; VÁZQUEZ-CANTELI; NAGY, 2019; LIM et al., 2020). Figure 10 illustrates the main components of this model.

The basic working of the PPO agent is the following (illustrated in Figure 10):

1. At state S_t, the PPO agent receives its first observation, w_t, from the environment;
2. As the agent did not perform any action at the first timestep, it does not receive a reward;
3. Both the actor and the critic networks with an orthogonal initialization;
4. The critic network receives the observation w_t and the reward r_t and conducts the policy iteration process, updating the value function and sending the Q value to a policy constraint function;
5. The actor network sends its predictions related to the expected rewards to the policy constraint function;
6. The policy constraint function sends the actor network the new Q value from the policy update;

7. The actor network selects the best action and performs action a_t;
8. The environment receives the action at and changes to a new state, S_{t+1};
9. The environment sends the observation w_{t+1} and the reward r_t to the actor and critic networks. This cycle continues for T interactions or timesteps;
10. The cycle repeats until a training mini-batch is completed. Then, the weights in the actor and critic networks are updated based on the networks' losses, and the steps continue until the ending of the training stage.

The works by François-lavet et al. (2018), Li (2018), and Schulman et al. (2017) contain an in-depth description of the workings of the PPO model, along with its mathematical formulation. The following paragraphs contain a description of state-of-the-art literature on the use of DRL agents for stock trading.

Dantas and Silva (2018) proposed using Q learning for stock trading in the Brazilian market, conducting an extensive experiment with several stocks and comparing with the BH strategy and two technical analysis strategies (Bollinger bands and MACD). The agent's inputs were: (i) four TIs (Bollinger bands, MACD, stochastic oscillator, and daily return). Its reward function was a modification of the daily return to consider the stock's current movement at time t.

Dantas and Silva (2018) evaluated their model from 2011 to 2017 (considering the last year as the test subset) on trading 22 individual stocks in the Brazilian stock market. Although it obtained better results than the BH strategy in 17 out of the 22 stocks evaluated, its cumulative returns were lower than the BH strategy considering all stocks (14.21% for the Q learning model's best configuration versus 14.53% for the BH strategy).

Nevertheless, the work of Dantas and Silva (2018) could be improved in several ways: (i) it did not consider transaction costs, which could impact significantly on the model's results; (ii) in general, it did not provide a better cumulative return that the BH strategy; (iii) it did not evaluate different trading scenarios; and (iv) it did not explore losses or risk-related metrics, such as maximum drawdown and Sharpe ratio, which are essential for decision-making in this domain. Additionally, state-of-the-art models such as DDPG could be used, with an adaptation of the proposed MDP.

Conegundes and Pereira (2020) is one of the first works to implement the DDPG DRL agent on the Brazilian stock market. Those authors proposed a trading system using the DDPG DRL agent to focus on the asset allocation problem and implemented it to evaluate

the cumulative percentage return and annual average maximum drawdown on ten indexes of the Brazilian stock market. The authors explored several interesting baseline models on test subset from 2017 to 2019: the BH strategy and the recommended stock portfolios from significant banks, provided at the beginning of each calendar year. Three variations of the DDPG agent were evaluated: two, three, and five days of window length. The DDPG DRL agent with a two days window length presented the best cumulative return (331% versus 92% for the BH strategy and 177% for the best stock portfolio recommended by the Santander bank). However, none of the models implemented presented a lower average maximum drawdown than the BH strategy (19% maximum drawdown for the DDPG model with two days window length versus 14% for the BH strategy) (CONEGUNDES; PEREIRA, 2020).

Nevertheless, the work by Conegundes and Pereira (2020) presents several drawbacks which are addressed in the present thesis: (i) it only considered one DRL agent (without a comparison with other baseline models, such as the PPO); (ii) it did not explore important hyperparameters that may influence the model; (iii) it did not consider the impact of additional features to OHLCV data, such as TIs, market sentiment, or price prediction models; and (iv) it can only be used for day trading as all positions are closed at each trading day, with significant changes needed to operate on different frequencies.

One of the main recommendations by Conegundes and Pereira (2020) for works on the use of DRL for trading is to explore the use of additional features in the state space, including technical and fundamental indicators. This aspect is explored in the current research, considering the prediction of prices and market sentiments extracted from the M1 and M2 modules, respectively.

Li, Rao and Shi (2018) developed an actor-critic method similar to DDPG for stock market trading. Its main difference was using an LSTM with a siamese structure on the actor and critic networks. In this way, they could capture temporal patterns in the data. The authors tested their model to trade the CSI300 Chinese stock market index from 2005 to 2018, considering three scenarios: (i) with high volatility and upward and downward trends (2005-2010); (ii) with a downward trend (2009-2014); and (iii) with upward and downward trends but with lower volatility (2013-2018).

The model proposed by Li, Rao and Shi (2018) considered as features the open, high, low, and close prices. The authors considered a transaction cost of 0.1% per order and evaluated three different models: (i) the proposed model, which was called DACT (deep actor-critic trader); (ii) deep direct reinforcement learning (DDRL); and (iii) DQN. The

baseline model was the BH strategy. The authors observed that: (i) the DACT model provides better returns than the others and the BH strategy in all scenarios; (ii) it reduces losses in comparison to all the other models; (iii) the DQN model only presents advantages on continuous downward trends.

It is important to note that the work of Li, Rao and Shi (2018) did not consider: (i) additional features other than raw daily prices; (ii) a thorough description of the model's MDP, making it difficult to replicate their experiments; (iii) loss-related metrics such as maximum drawdown; and (iv) a study of which components of the model impacted the most in the final results. In the current thesis, all of those aspects were explored.

Lei et al. (2020) proposed using a DRL agent with gated recurrent units (GRU) as the basis of an algorithmic trading system. The main contributions of those authors were related to: (i) considering a model that extracts patterns from historical data, providing inputs to the DRL agent; (ii) evaluating multiple trading scenarios; (iii) weighting the features at each timestep; and (iv) using a vanilla policy gradient method for the DRL agent. The proposed model results were significantly better than the baselines (BH strategy, the SFM DL model, RRL, the DDR DRL model, and a GRU-based DRL) in terms of cumulative returns, annualized returns, and annualized Sharpe ratio. Nevertheless, the authors did not evaluate: (i) the use of market sentiment as a feature; (ii) the impact of the TI features in relation to the OHLCV data; and (iii) state-of-the-art DRL methods such as DDPG.

Yang et al. (2020) proposed using an ensemble of three actor-critic DRL models: PPO, advantage actor-critic (A2C), and DDPG. The model was tested with 30 liquid stocks that are part of the Dow Jones Industrial Average from 2009 to 2020 (with 2016 to 2020 used as the test subset). It was then compared to a BH strategy of this index and the minimum-variance strategy. Five metrics were evaluated: annual returns, cumulative returns, annual volatility, Sharpe ratio, and maximum drawdown. The authors observed that the ensemble strategy obtained better returns (a cumulative return of 70.4% versus 38.6% for BH and 31.7% for the minimum-variance), lower volatility (9.7% versus 20.1% for BH and 17.8% for minimum-variance), a considerably higher Sharpe ratio (1.30 versus 0.47 for BH and 0.45 for minimum-variance), and a significantly lower maximum drawdown (-9.7% versus -37.1% for BH and -34.3% for minimum-variance). However, the authors did not consider the use of market sentiment features, price predictions from other models, or the use of statistical tests to evaluate if the differences between the models can be considered significant.

Wu et al. (2019) proposed a DRL agent that uses policy gradient and an LSTM to identify patterns in the data. The agent's inputs are OHLCV data and fifteen TIs, including: exponential moving average, triple exponentially smoothed average, relative strength index, volatility volume ratio, commodity channel index, directional movement, MACD, and Williams %R (WR), among others. Wu et al. (2019) evaluated the model for six stocks in the Chinese stock market from 2015 to 2017, concluding that the average profit for their proposed agent is significantly higher than the average profit for the baseline line (a version of the DRL agent with an MLP instead of an LSTM).

Nevertheless, Wu et al. (2019) considered only a weak version of their proposed model as a baseline, which is not recommended. The authors could have compared their model with the BH strategy, which is the most used baseline in comparing trading systems and provides good results in almost all situations, aside from periods with steep downward trends. Another problem was that the authors only considered the final profit as the evaluated metric, so it is impossible to evaluate the risk of the model's policy. Lastly, one major problem with using the model proposed by Wu et al. (2019) in real-life trading is that the authors fixed the trading order to 10.000 Chinese yuans. Flexible order size is essential for a good performance of the trading system, as different periods may present different trends, which may influence the order sizes.

Wang et al. (2019) proposed the AlphaStock model, a DRL agent with LSTMs with attention mechanisms that use the Sharpe ratio as a reward function to better balance risks and returns of the different actions. One of the authors' main concerns was the interpretability of the model, and, unlike other works in the literature, they have extensively explored the strategy that the model adopted. Another interesting aspect is that the DRL agent could also short positions.

Wang et al. (2019) conducted experiments with stock data from 1970 to 2016 (with data from 1990 to 2016 as the testing subset), and more than a 1.000 stocks were considered per year. The baseline models used were: BH strategy, cross-sectional momentum, and time-series momentum strategies, robust median reversion, fuzzy deep direct reinforcement, and two variations of the AlphaStock model. Six financial metrics were evaluated: annual returns, annual volatility, annualized Sharpe ratio, maximum drawdown, Calmar ratio, and downside deviation ratio. The AlphaStock model obtained better results on the US market than all the baselines for the evaluated metrics, except for annual volatility (however, it beat the BH strategy). On the Chinese market, the model obtained better results than all the baselines.

However, the main weaknesses of the work by Wang et al. (2019) were: (i) not considering transaction costs; (ii) not evaluating the use of TIs; (iii) the size of the portfolio and the number of holding periods are fixed; and (iv) no statistical test was made on the final model results to evaluate the different models' results better. These aspects were explored in this work.

The following section concludes the literature review chapter, summarizing the main points observed in each section.

2.7 Chapter summary

This chapter described the main theoretical foundations, concepts, and state-of-the-art research related to this research's central themes. Section 2.1 described the main concepts and difficulties of stock price and trend prediction tasks, evaluating the traditional econometrics models (ARIMA, SARIMA, and SARIMAX), including the description of the EMH hypothesis. Section 2.2 described the main concepts related to ML and DL models, exploring the results obtained with different models in several stock markets. It was observed that DL models are better suited for complex time-series prediction, with the RNNs being state-of-the-art models.

Section 2.3 explored the importance of sentiment analysis to evaluate and predict market sentiment. It was observed that the hybrid approach provides more benefits, especially when using word embeddings and sentiment lexicons or dictionaries. Section 2.5 described the importance of considering trading metrics (related to returns, risks, and losses) instead of traditional ML-based metrics (precision, recall, F1, MSE, MAE, RMSE, among others) to develop trading systems. Lastly, section 2.6 explored the main concepts of RL and DRL models and their use in stock trading. It was observed that the DRL models, which combine DL and RL, are state-of-the-art on complex environments with non-linearities, such as the stock market.

3 MATERIALS AND METHODS

After conducting an extensive literature review and identifying the main research gaps, the methodology adopted for this work consisted of an agent-based stock market simulation, using real-time series data from the BOVA11 ETF and several inputs generated by the trading system. Two relevant scenarios were considered in this simulation.

According to Negahban and Yilmaz (2014), computer simulations can be described as models created for the development of experiments, which can help gain insights into its behaviors in different scenarios. There are several simulation methods, such as discrete event simulation, Monte Carlo simulation, and agent-based simulation, among others (MACAL; NORTH, 2014). Simulation models are essential to evaluate complex stochastic systems and scenarios (MACAL; NORTH, 2014; NEGAHBAN; YILMAZ, 2014).

The agent-based simulation is a technique that models complex systems based on individual and autonomous agents that interact among themselves and with their environment (MACAL; NORTH, 2014; NEGAHBAN; YILMAZ, 2014). These agents can learn with their actions and interactions on the environment, developing and improving their strategies (MACAL; NORTH, 2014; NEGAHBAN; YILMAZ, 2014).

Macal and North (2014) observed that agent-based simulation models have been gaining importance in numbers of works and applications. RL and DRL agents are examples of agent-based simulation models. Some examples of relevant works that used this method in the financial domain are Wang et al. (2019), Wu et al. (2019), Yang et al. (2020), Lei et al. (2020), and Li, Rao and Shi (2018). Therefore, the agent-based simulation of a stock trading market was chosen in this dissertation for evaluating the proposed system and its components on different implementation and trading scenarios.

To conduct this research and implement the simulations, six main steps were adopted. To better describe these steps, they were separated into three groups: (a) Price time series, comprising activities related to time series analysis and price trend prediction; and (b) Sentiment analysis, comprising activities related to sentiment analysis and prediction. As

an exception, in step 5, there is a third group of activities (c) related to implementing the DRL models. Group (i) is related to the M1 module (stock market price prediction), group (ii) to the M2 module (stock market sentiment prediction), and group (iii) to the MT module (stock market trading). Steps 4, 5, and 6 implement the market simulation and the two evaluated scenarios. These steps were:

1. **Data collection**

   (a) **Price time series**: the daily BOVA11 index OHLCV data were collected using the yfinance[1] library. It encompassed the period from 12/02/2008 (creation of the BOVA11 ETF) to 08/31/2020. The BOVA11 index was chosen for two main reasons: (i) this ETF reflects the Ibovespa index, which is used as a representation of the status of the overall Brazilia market; and (ii) trading an ETF is more accessible in real-life scenarios and can be done by any investor;

   (b) **Sentiment analysis**: two datasets were used: (i) D2-A, a labeled dataset for model tuning and training; and (ii) D2-B, an unlabeled dataset for predicting daily market sentiment. The first dataset contained 1,134 news titles from three relevant and trustworthy financial news sources: Valor Econômico[2], UOL Economia[3], and InfoMoney[4]. These were then independently labeled by three researchers following a labeling protocol into five sentiment scores: 1 (very negative), 2 (negative), 3 (neutral), 4 (positive), and 5 (very positive). The D2-B dataset initially contained (before processing) 113,226 news titles gathering from another very relevant and trustworthy source: Investing.com. This data source was used to develop the D2-B dataset because its available news database was more extensive than the sources used for gathering data for D2-A.

2. **Data processing**

   (a) **Price time series**: the data was analyzed to detect outliers and missing data using the following packages: scikit-learn[5], Pandas[6], Seaborn[7], and Matplotlib[8]. It was also separated into subsets. The first division, used to separate

[1] https://github.com/ranaroussi/yfinance
[2] https://valor.globo.com/
[3] https://economia.uol.com.br/
[4] https://www.infomoney.com.br/
[5] http://scikit-learn.org/
[6] https://pandas.pydata.org/
[7] https://seaborn.pydata.org/
[8] https://matplotlib.org/

the trading data from the training data, generated two subsets: (i) training subset: 2008 to 2018; and (ii) trades subset: 2019 and 2020. Therefore, the model was trained on the first subset and predicted values for the second subset (which were then used by the relevant trading models as features). Nevertheless, as one of the main objectives of this work was to explore in-depth and compare various price prediction models and their hyperparameters, the first subset was further divided into two subsets: (i) training subset: 2008 to 2017; and (ii) test subset: 2018. Two important cross-validation methods (blocking time splits and time series splits) were used with five folds to better explore the different models and hyperparameters on various conditions. The following TIs were also generated with the ta library[9] and used as features by some of the models: (i) for volume: Accumulation Distribution Indicator (ADI); (ii) for volatility: Bollinger Bands (BB); (iii) for identifying trends: Moving Average Convergence Divergence (MACD); and (iv) for momentum: Relative Strength Index (RSI), stochastic RSI, and Williams %R (WR). These are used in several works in the literature (LIU et al., 2020; YANG et al., 2020; KARA; BOYACIOGLU; BAYKAN, 2011; MARTINEZ et al., 2009; DANTAS; SILVA, 2018) and are believed by some researchers and practitioners to capture various data patterns. The extensive reviews by Ryll and Seidens (2019), Meng and Khushi (2019) also observed several works in the literature using TIs for automated trading;

(b) **Sentiment analysis**: both datasets (D2-A and D2-B) were processed using the following packages: scikit-learn, pandas, seaborn, matplotlib, nltk[10], and spacy[11], for its use on the DL models implemented. This consisted of the following main activities: tokenization, lemmatization, elimination of stopwords, and identification and elimination of irrelevant news titles. On D2-A, the data was labeled independently by three researchers, following a sentiment labeling protocol that was created for this work. It consisted of evaluations of: (i) importance of the news title; and (ii) evaluation and score of the news title sentiment based on the expected impact on the market (for example, news titles such as "the worst harvesting season in the last ten years" probably would have a very negative impact on the price of the given asset so that it would be given five as its score). The labels were then aggregated using a simple

[9] https://github.com/bukosabino/ta
[10] https://www.nltk.org/
[11] https://spacy.io/

average and rounded up to result in each news title's final score. The D2-A dataset was divided into a training (80% of the total dataset or 907 news titles) and a test subset (20% of the total dataset or 227 news titles). For the final training, the D2-A was used to train the model, and the D2-B was used for model prediction.

3. **Exploratory data analysis**

   (a) **Price time series**: the time series was analyzed, considering: (i) trends; (ii) seasonality; (iii) autocorrelation; and (iv) general statistics, including mean, median, high and low prices, as well as the distribution of daily returns (also known as the daily percentage change);

   (b) **Sentiment analysis**: the D2-A dataset was evaluated in terms of sentiment score distribution throughout the dataset.

4. **Model implementation and hyperparameters analysis**

   (a) **Price time series**: four categories of models were implemented: (i) econometrics models: ARIMA, SARIMA, and SARIMAX; (ii) ML models: SVR univariate, SVR multivariate, and AdaBoost; (iii) DL models: LSTM univariate and LSTM multivariate; and (iv) ensemble models. The econometrics models were implemented using the statsmodels[12] library, and the hyperparameters evaluated (all with values from 0 to 3) were: (i) for all models: p (autocorrelation component), d (differentiation component), and q (moving average component); and (ii) only for SARIMA and SARIMAX: P, D, Q (seasonal counterparts of p, d, and q), and S (seasonal component). The ML models were implemented using the scikit-learn library, and the hyperparameters evaluated were: (i) for the SVR: kernel (linear, rbf, sigmoid, and poly), C (regularization parameter, with values 0.01, 0.1, 1, 5, 10, 20, and 50), and epsilon (penalty parameter, with values 0.001, 0.01, and 0.1); and (ii) for the AdaBoost: number of estimators (5, 10, 15, 20, 25, 30, and 100), learning rate (0.1, 0.5, 1, 2, 2.5, 3, 5, 8, and 10), and loss function (linear, square, and exponential). The DL models (LSTM univariate and LSTM multivariate) were implemented using the TensorFlow 2.0 library, and the hyperparameters evaluated were: batch size (2, 4, 8, 16, and 32), number of neurons on the LSTM layer (4, 8, 10, 50, 100, 200, 500, and 1000), window length (also known as past history, with values of 1, 2, 3, 4, 5, and 10), and number of training epochs (30,

[12]https://www.statsmodels.org/stable/index.html

100, 500, 1000, and 5000). The definition of hyperparameters and their values considered the implementations and results obtained in the literature for the different models. For all models, two additional options were evaluated: (i) the cross-validation method used, considering blocking series split and time series splits; and (ii) the use of TIs as additional features (only for the multivariate models). The loss function used for the LSTM was the MSE, and the optimizer was the Adam optimizer. The hyperparameters that showed the best result through the analysis were chosen for the implementation of the final model;

(b) **Sentiment analysis**: following the work by Ferreira et al. (2019), two models were implemented, MLP and CNN, using a Portuguese version of the GloVe 300 dimensions embedding (PENNINGTON; SOCHER; MANNING, 2014; HARTMANN et al., 2017). They were implemented using the keras library running on top of TensorFlow 2.0[13] and were trained for 30 epochs, each with a 3 splits K-Fold cross-validation, based on the work by Ferreira et al. (2019). The CNN had one convolutional layer. In both models, the following hyperparameters were analyzed: (i) dictionary, considering no dictionary, Sentilex (SILVA; CARVALHO; SARMENTO, 2012; CARVALHO; SILVA, 2015), containing a vocabulary of 82.348 words, OpLexicon (SOUZA; VIEIRA, 2012), containing a vocabulary of 32.192 words, and WordNetAffectBR (PASQUALOTTI; VIEIRA, 2008), containing a vocabulary of 291 words; (ii) batch size of 32 and 64; (iii) number of neurons of the dense layers of 50, 100, and 150; (iv) dropout rate, considering no dropout, 0.3, 0.4, and 0.5; and (v) number of hidden layers of 1, 2, 3, 5, and 10. Two additional hyperparameters were evaluated for the CNN: (i) number of filters: 128 and 256; and (ii) size of filters: 2 and 4. The loss function used was the MSE and the optimizer used was the Adam optimizer. Those models' objective was to correctly identify the specific news title's target sentiment value and re-adjust its weights during the training. The hyperparameters that showed the best result through the analysis were chosen to implement the final model.

(c) **Trading models (DRL)**: following the works by Liu et al. (2020) and the MDP description in Chapter 2, section 2.8 (which describes how the market simulation works and how the agent interacts with the environment), two DRL models for trading were implemented using the FinRL library: DDPG and PPO. They were implemented considering two trade scenarios: (i) Trade

[13]https://www.tensorflow.org/

1: 2019 and 2020; and (ii) Trade 2: 2020. For the first trading scenario, they were trained from 2011 to 2018. For the second trading scenario, they were trained from 2011 to 2019. Eight models were implemented, considering different inputs: (i) MT1A, MT1B, and MT1C considered OHLCV and the price predictions from the M1 module; (ii) MT2 considered OHLCV and the sentiment predictions from the M2 module; (iii) MT3 considered only OHLCV; (iv) MT3ta considered OHLCV and TI; (v) MT4 considered OHLCV, the price predictions from the M1 module and the sentiment predictions from the M2 module; and (vi) MT4ta considered OHLCV, the price predictions from the M1 module, the sentiment predictions from the M2 module, and TI. All models considered two options for maximum order size: 10 and 200. These values were chosen to limit the value traded per day. This analysis is essential for high volatility scenarios. The hyperparameters evaluated for the DDPG DRL model were: (i) batch size: 8, 64, and 128; (ii) timesteps: 10,000, 100,000, and 200,000; and (iii) buffer size: 1,000, 10,000, and 100,000. The hyperparameters evaluated for the PPO DRL model were: (i) number of steps: 8, 64, and 128; (ii) timesteps: 10,000, 100,000, and 200,000; and (iii) learning rate: 0.00025, 0.0005, and 0.001. For the training subset, the metrics evaluated were: mean total reward, considering 30 executions; and standard deviation of the mean total reward. These are essential to evaluate the model's learning capabilities.

5. **Final models implementation**

   (a) **Price time series**: the final models and the models ensembles were built using the best hyperparameters identified on step 4-a. They were then trained on the dataset from 2008 to 2018 and used to predict prices for the whole dataset used by the trading models (2011 to 2020). The ensembles' predictions were simple averages of the predictions of two trained models. This simulates the real trading scenario, in which only past prices and trends are known. The final predictions were used as inputs for the MT1A, MT1B, MT1C, MT4, and MT4ta trading models;

   (b) **Sentiment analysis**: the final models were built using the best hyperparameters identified in step 4-ii for the MLP and CNN models. They were trained using the D2-A dataset (labeled news titles) and used for predicting the sentiment scores for the news on dataset D2-B (unlabeled news titles from 2011 to 2020). The sentiment scores were then aggregated by day (using the simple average of all the predictions for that day). The final predictions per day were

used as inputs for the MT2, MT4, and MT4ta trading models;

(c) **Trading models (DRL)**: one model configuration (considering hyperparameters values and DRL agent) for each combination of trading model and maximum order size. Therefore, there were eight models with two maximum order sizes each (10 and 200). Each of these combinations had specific hyperparameters values and a DRL agent for that combination. Then, the models were trained in the training subset for each trading scenario and used for trading on the trading subset.

6. **Model comparison**: the final comparison of all trading models (MT1A, MT1B, MT1C, MT2, MT3, MT3ta, MT4, and MT4ta) was conducted considering the two trading scenarios and the six financial metrics: annual returns, cumulative returns, annual volatility, stability, Sharpe ratio, and maximum drawdown. The baseline model used was the BH strategy, which is the most common baseline strategy for trading models. The models were then evaluated based on all these metrics for each scenario, and the best model for each was chosen.

The implementation was done using Python on a server with the following technical specifications: Intel Corei7-8700K 3.70GHz CPU, 64GB of RAM, and 2 NVIDIA 1085Ti graphics cards. In the next section, the main results of the exploratory data analysis and the implementations are discussed. The main research question and the eight secondary research questions are answered, and the final model is chosen based on the financial metrics.

4 RESULTS

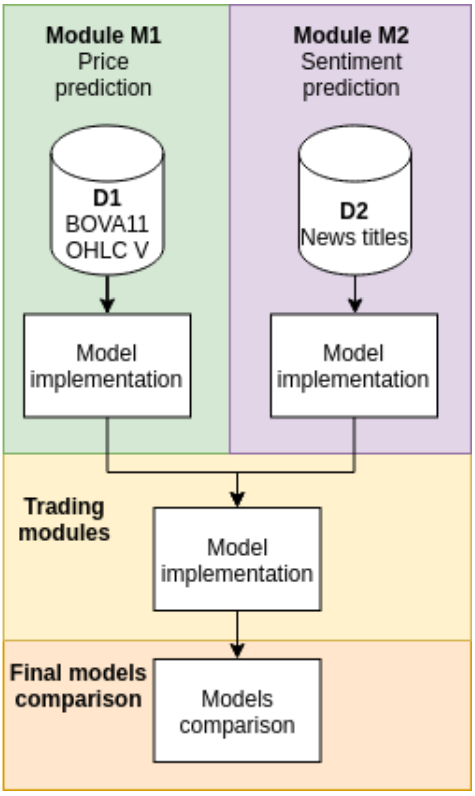
This chapter contains the main results of this research and is divided into the following sections: 4.1 explains the proposed trading system and its main components; 4.2 describes the MDP for the DRL agent used for stock trading in this work, which is based on state-of-the-art research; 4.3 describes and evaluates the models on the M1 (stock market price prediction) module; 4.4 describes and evaluates the models on the M2 (stock market sentiment prediction) module; 4.5 describes and evaluates the models on the MT (trading models); 4.6 contains the final comparison of all trading models; 4.7 answers the main and secondary research questions; and 4.8 concludes this chapter, including its key takeaways.

4.1 Proposed system and its components

The proposed system in this work is composed of three main modules: the stock price prediction module (M1), the stock market sentiment module (M2), and the trading module (MT), as illustrated in Figure 11. The inputs for the M1 module are OHLCV data for each day. The inputs for the M2 module are the relevant news titles in Portuguese during that trading day. Lastly, several variations of the MT module were evaluated, considering various features. These will be described in-depth during this section.

The final models comparison was conducted considering six relevant economic indicators: Annual returns, Cumulative returns, Annual volatility, Sharpe ratio, Stability, and Maximum drawdown. As stated in Chapter 2, this is extremely important for evaluating trading models in the financial domain. This is because the models may achieve good results for machine learning metrics (such as MSE, MAE, RMSE, R2, among others) and still present inferior economic results (SEZER; GUDELEK; OZBAYOGLU, 2020). This could lead to low returns, low Sharpe ratio, high volatility, among others problems. Therefore, using relevant financial indicators to evaluate trading systems that use machine learning and artificial intelligence models is more appropriate than using traditional machine learning regression or classification metrics.

Figure 11 – Simplified illustration of the proposed trading system

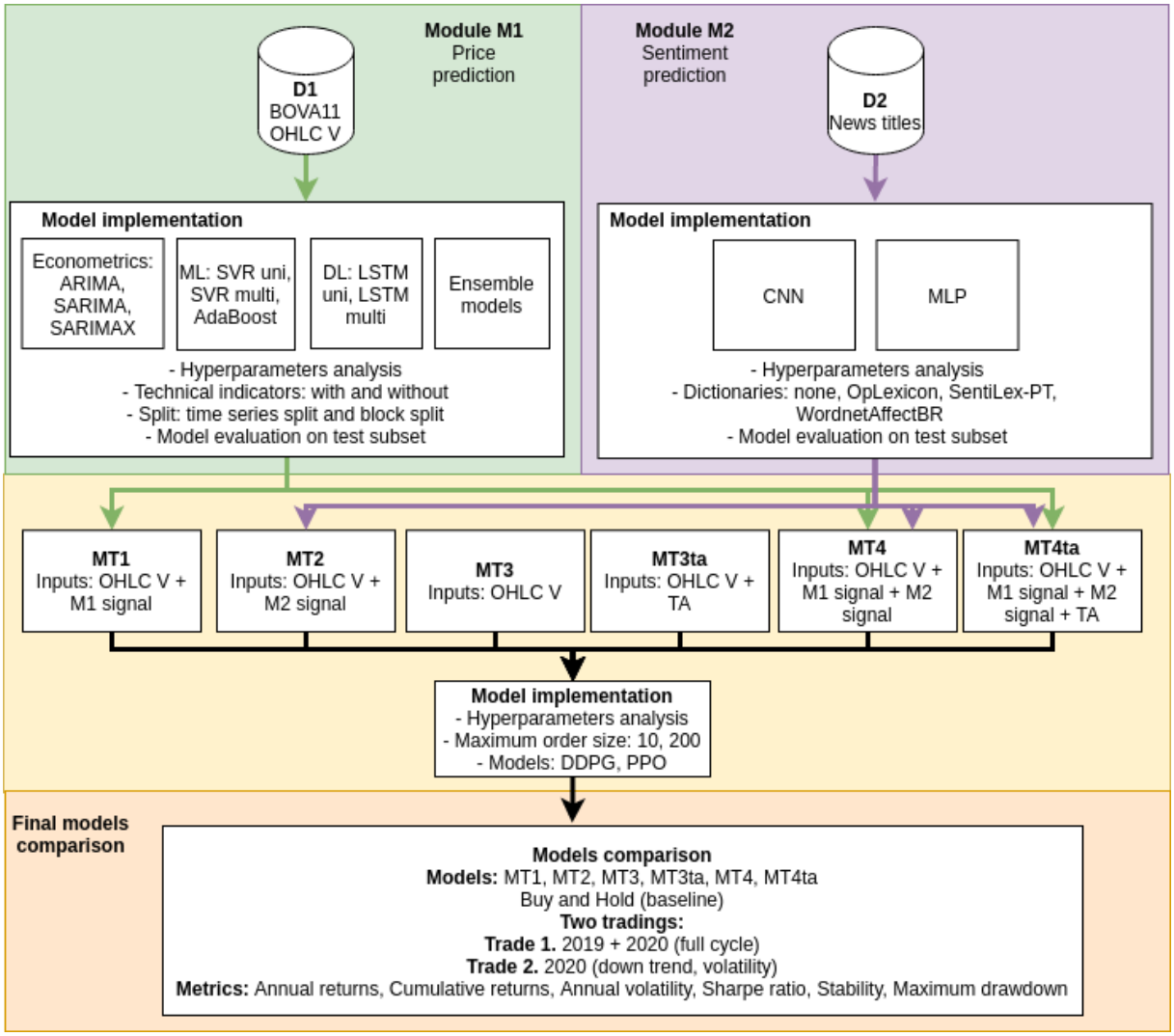


Source: elaborated by the author.

Figure 12 contains an in-depth illustration of all the modules of the proposed system. On the M1 module, which will be explored in section 4.3, several models were implemented. These were divided into the following categories, due to their main characteristics: (i) econometrics models: ARIMA, SARIMA, and SARIMAX; (ii) ML models: SVR univariate, SVR multivariate, and AdaBoost; (iii) DL models: LSTM univariate and LSTM multivariate; and (iv) ensemble models. This categorization followed important works in the literature, such as Rundo et al. (2019), Ballings et al. (2015), Chong, Han and Park (2017), Ryll and Seidens (2019), and Mehtab, Sen and Dasgupta (2020). It is important to observe that the LSTM models are considered state of the art on time series prediction tasks and have been widely used for predicting stock prices and trends (MEHTAB; SEN; DASGUPTA, 2020; RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020).

The main hyperparameters of each model were evaluated on the training and validation subsets. The use of relevant TIs and two cross-validation splits (time series splits and block splits) were evaluated. The final models were then trained with the train and validation subsets and evaluated on the test subset. The model with the best results in terms of MSE was chosen for use in this module. This model's outputs (denominated

Figure 12 – In-depth illustration of the proposed trading system



Source: elaborated by the author.

stock price prediction signals in this work) were used as inputs for the following trading models: MT1, MT4, and MT4ta.

On the M2 module, which will be explored in section 4.4, two state-of-the-art DL models for sentiment analysis were implemented: MLP and CNN. In a related work (FERREIRA et al., 2019), their use was thoroughly explored for the English language. The same model architectures were used in this work due to their satisfactory results (FERREIRA et al., 2019). The main hyperparameters of each model were evaluated on the training and validation subsets. The use of different relevant dictionaries (OpLexicon, SentiLex-PT, WordnetAffectBR, or no dictionaries) was also evaluated, based on the assumption that the knowledge on a lexicon could improve the results of the model. This is supported by several works on the literature, such as Ferreira et al. (2019), Mansar et

al. (2017), and Nassirtoussi et al. (2014).

The final models were then trained with the train and validation subsets and evaluated on the test subset. The model with the best results in terms of MSE was chosen for use in this module. This model was then used to predict the market sentiment based on the set of news titles in each trading day. This model's outputs (denominated stock market sentiment prediction signals in this work) were used as inputs for the following trading models: MT2, MT4, and MT4ta.

For trading modules (MT), explored in section 4.5, six different models were evaluated, varying their inputs. All of them considered the traditionally used features for algorithmic and DRL trading: volume, open, high, low, and close prices, as in most of the works surveyed by Meng and Khushi (2019) and Fischer (2018). Besides those features, the following models also considered as features: (i) MT1: predicted price from M1, with three variations (MT1A, MT1B, and MT1C); (ii) MT2: predicted sentiment from M2; (iii) MT3: no additional features; (iv) MT3ta: relevant TIs; (v) MT4: predicted price from M1 and predict sentiment from M2; and (vi) MT4ta: predicted price from M1, predict sentiment from M2, and relevant TIs.

The primary purposes of evaluating multiple models were to: (i) better estimate the impact of the system modules on the different trading scenarios; (ii) explore the impacts of the different DRL models used (DDPG and PPO); and (iii) explore the impacts of the two maximum order sizes chosen (10 and 200). This allowed a better understanding of each model's importance on the different trading scenarios and their impacts on the evaluated quality metrics.

Lastly, the different trading modules were evaluated at a task denominated Final models comparison. This evaluation is described in section 4.6. In the final models' comparison, the six financial indicators that were chosen as quality metrics (annual returns, cumulative returns, annual volatility, Sharpe ratio, stability, and maximum drawdown) were evaluated on two trading scenarios: (i) Trade 1, which consisted of data for two years with both bull and bear trends; and (ii) Trade 2, which consisted of data for one year with a bear trend and high volatility. This evaluation's primary assumption is that different trading system configurations (different models and hyperparameters used on the trading modules) could provide better results on different scenarios. This is a fundamental assumption explored in only a few works, such as Li, Rao and Shi (2018) and Lei et al. (2020).

4.2 MDP for DRL for stock trading

According to Vázquez-canteli and Nagy (2019), the MDP is the formal framework used to describe (and to implement) the main elements of an RL or DRL model. It follows the Markov property, which is related to the assumption that the past observation contains all the important information of the observation history (LI, 2018; FRANÇOIS-LAVET et al., 2018; FISCHER, 2018). In other words, it assumes that the agent's state at time t contains all the relevant past information for decision making (DANTAS; SILVA, 2018). The majority of works on the trading domain have used the MDP to design and describe the RL or DRL solution, as observed in the reviews by Meng and Khushi (2019) and Fischer (2018).

According to Vázquez-canteli and Nagy (2019), the MDP is composed of four main elements: a set of states S (the state space), a set of actions A (the action space), a reward function r (which should consider both the states and actions), and the transition probabilities between the states P. A policy π is used to map the states to specific actions in a deterministic or stochastic manner depending on the model. As Vázquez-canteli and Nagy (2019) stressed, the value function V represents the expected returns (considering immediate and future returns) for the agent at the state s and following the policy π. For each action taken, the agent receives a reward R, a, s, s' (which can be positive or negative). The objective of the agent is to find the optimal policy π^*, which maximizes the expected future rewards.

Vázquez-canteli and Nagy (2019) describe that the MDP is solved by identifying the optimal policy (the one that maximizes the expected returns). Therefore, it is directly connected with the reward function r, the transition probability, and the system's dynamics. The MDP is a 5-tuple $\langle S, A, T, R, \gamma \rangle$ (FRANÇOIS-LAVET et al., 2018):

- S **is the state space**, which contains all possible states;
- A **is the action space**, which contains all possible actions to be performed by the agent;
- $T : SxAxS \rightarrow [0, 1]$ **is the transition function**, which contains the transition probabilities between the states;
- $R : SxAxS \rightarrow R$ **is the reward function**, which is a continuous value between the extreme negative value (which can be 0 in some cases or a negative value, depending on the design of the reward function) and $Rmax$;

- $\gamma \in [0, 1]$ **is the discount factor**, an hyperparameter of most RL and DRL models, which can vary as the model learns (to better balance between exploration and exploitation).

In summary, the main objective of the MDP is to describe the main components of the problem design, including: (i) how the environment is simulated; (ii) if the environment's dynamics are known; and (iii) what are the main components of the tuple S (state space), A (action space), and R (reward function); and (iv) the RL or DRL agents implemented (VÁZQUEZ-CANTELI; NAGY, 2019; FRANÇOIS-LAVET et al., 2018). These items are described in the following paragraphs. The MDP used in this research for the automated trading using the DRL agents is described in the following paragraphs. It is based on the MDP used by Liu et al. (2020), implemented on the FinRL library, which was used to implement the trading module (MT).

The environment in this thesis was implemented using the FinRL[1] library. In this library, the environment design follows the OpenAI Gym[2] framework (LIU et al., 2020). The stock market is simulated based on providing data from an imported dataset (containing the environment's features). One data point is provided at each timestep, and the agent will make an action.

The reward is then calculated based on the observation returned from the environment (features from the dataset), and the agent adjusts the weights of the actor and critic networks. The agent's main objective is to maximize the rewards, and it will adjust its policy (which can be considered a trading strategy that is automatically generated, implemented, and updated) according to the results of its actions and the rewards received. After one epoch (when the agent has visited all data points of the dataset), the environment will start sending the observations from the beginning of the dataset. This process continues until the number of timesteps (defined as a hyperparameter) is reached. In the test subsets, the agent will not change the weights of its parameters, only applying the last policy (trading strategy) that was learned during the training stage.

This work focuses on using policy gradient methods for the financial trading module (MT). These were chosen because: (i) they can provide good results on problems with continuous state and action spaces (making it easier to escalate the module to consider multiple stocks, portfolio optimization, and more actions); (ii) there is a gap in the literature for exploring those models on financial trading, especially on developing mar-

[1] https://github.com/AI4Finance-LLC/FinRL-Library
[2] https://gym.openai.com/

kets; and (iii) these are considered state-of-the-art in several domains, such as robotics (FRANÇOIS-LAVET et al., 2018; LI, 2018; FISCHER, 2018). Additionally, the works of Yang et al. (2020) and Liu et al. (2020) have observed good results on applying policy gradient methods for stock trading.

According to François-lavet et al. (2018), DRL policy gradient methods' primary objective is to find a suitable policy using DL by constantly improving the current policy using experimentation and the stochastic gradient ascent algorithm. The DDPG model can be defined as an expansion of the DQN algorithm to consider continuous actions (FRANÇOIS-LAVET et al., 2018). In this way, it can better balance the exploration and exploitation of actions, as greedy policy improvement in continuous action spaces would demand considerable computational resources to maximize the policy function at each timestep (FRANÇOIS-LAVET et al., 2018).

The implementation in this work considers that the dynamics of the environment are unknown. This reflects the real-life of predicting and trading on the stock market: there are many known and unknown factors that may influence the asset prices (NASSIRTOUSSI et al., 2014; YADAV; JHA; SHARAN, 2020; JIN; YANG; LIU, 2020; HU et al., 2018). Nevertheless, many factors were not considered because they were out of the scope of this work. Factors such as fundamental analysis data could be better explored in future works, using the same MDP and methodology used in this research, with a few adaptations in the state and action spaces and the reward function.

The MDP used in this work is based on the one proposed by Liu et al. (2020), which was implemented in the FinRL library. The main changes that were made are related to the state space by incorporating additional features. The main components of the tuple $S, A, and R$ were:

- **State space** S: the state space contains the observations received by the agent from the environment at each timestep. These observations were vectors composed of multiple features that belong to the dataset used for simulating the trading environment. These features were:
  - For all models: OHLCV, number of shares owned, and balance in the account (amount of money in the account that can be used for buying shares);
  - Only for the models that considered the price prediction feature: the price prediction feature from the M1 module;
  - Only for the models that considered the sentiment feature: the market senti-

ment prediction feature from the M2 module;

  - Only for the models that considered TIs: for volume (ADI), for volatility (BBH), trends identification (MACD), and momentum (RSI, stochastic RSI, and WR).

- **Action space** A: the action space contains all the actions that the DRL agent can take at each timestep. Three possible actions were considered: buy (action +1), hold (action 0), and sell (action -1) the number of shares. Directly related to the action taken is the maximum order size, which limits the maximum number of shares that can be negotiated in a given timestep;

- **Reward function** $r(s, a, s')$: the reward function guides the agent towards the learning process and the development of better policies (or trading strategies). In this work, the reward function considered was the same as in Liu et al. (2020), the total portfolio value change. This can be defined, for each timestep, as the equation below. R represents the reward, No represents the number of shares owned, C represents the closing price, and B represents the balance (amount of money available that was not spent buying stocks).

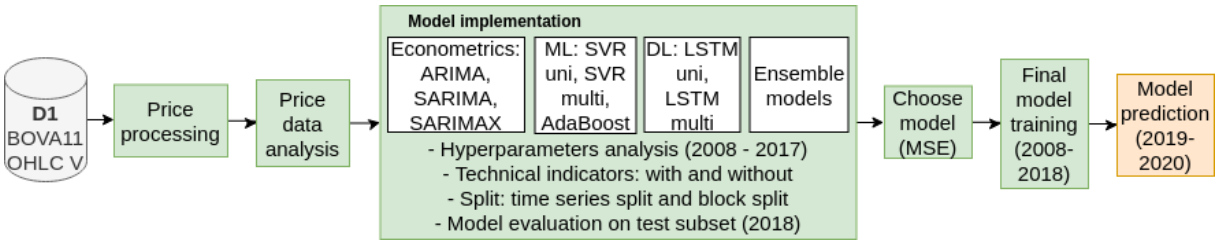
$$R_t = ((No * C) + B)_t - ((No * C) + B)_{t-1} \tag{4.1}$$

Several adaptations can be made to the MDP in future works. For adopting the model for trading multiple assets, it is necessary to adapt the state space to incorporate all the data from the shares themselves and the calculation of the account balance. Changes in the action space are also needed so that the final action space will be 3^n (where n is the total number of assets). For adopting the model for shorting stocks or operating with options, it is necessary to adapt the action space, including those new actions. Lastly, for adopting new reward functions, such as log returns, Sharpe ratio, among others, changes in the reward function must be made. Nevertheless, it is essential to notice that the MDP used can be easily adapted to those other use cases.

Lastly, the implemented DRL agents were the DDPG and PPO agents. The FinRL library was implemented using the Stable Baselines 2 library[3] and TensorFlow 1.14[4]. The Stable Baselines library contains several improvements (and more extensive documenta-

[3]https://stable-baselines.readthedocs.io/en/master/
[4]https://www.tensorflow.org/

Figure 13 – Main steps for building and evaluating the M1 module - Stock market price prediction



Source: elaborated by the author.

tion) of the RL and DRL models present on the OpenAI Baselines library[5]. The three main hyperparameters for each model were evaluated and the maximum order size per day, and the number of timesteps.

4.3 M1 - Stock market price prediction module

This section contains the main results of the M1 module, considering different time series analysis models. It is divided into the following subsections: 4.3.1 contains an exploratory analysis of the BOVA11 price dataset, exploring its main characteristics and the division between train, validation, and test subsets; and 4.3.2 describes the main results obtained on the hyperparameters analysis and the final models on the test subset. Lastly, three models were chosen for testing on the two trading scenarios. This module's main objective is to provide an additional feature for the trading module to its trading results.

Figure 13 illustrates the M1 module. It contains the following steps: (i) OHLCV data gathering from a stock market price provider; (ii) price processing, detecting and dealing with outliers and missing data, as well as generating TIs when needed; (iii) price data analysis, which is used for generating charts and statistics to analyze the data (described in the following subsection); (iv) model implementation, considering all the evaluated models, its main hyperparameters, the use or not of TI, the type of cross-validation split (time series split or block split), and final model evaluation on the test subset, considering the MSE as the quality metric; (v) final model training, considering the whole dataset; and (vi) model prediction on the trading subsets. The results of step (vi) are then fed to the trading module, as described in section 4.5.

[5]https://github.com/openai/baselines

Figure 14 – BOVA11 price chart from 2008 to 2020

Source: elaborated by the author.

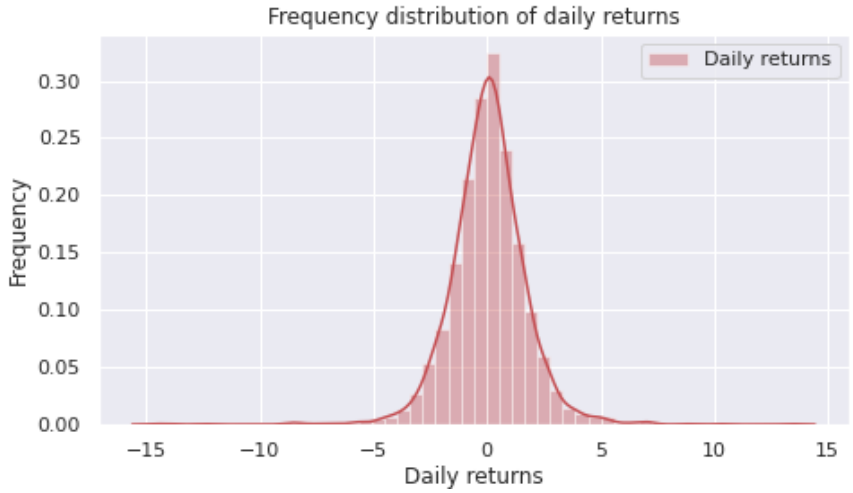
4.3.1 Exploratory data analysis

The dataset used for training the M1 module was composed of five features: open prices (first price on trading day t), high prices (highest price on trading day t), low prices (lowest price on trading day t), close prices (final price on trading day t), and volume (total volume of shares of the asset traded on trading day t). These are typically denominated OHCLV (open, high, low, close, and volume) on the financial trading domain. The data was collected using the yfinance library, which gathers data from the Yahoo! Finance platform. Figure 14 illustrates the close price from 2008 to 2020. This is the most widely used feature as a target on trading and price prediction works, such as in most of the works reviewed by Ryll and Seidens (2019) and Sezer, Gudelek and Ozbayoglu (2020).

It is possible to observe that the prices varied a lot during the period, presenting clear upward trends between 2008 and 2010, 2016 and 2018, and 2019. It presented steep downward trends between 2018 and 2019 and in 2020. The latter was the most significant decline in stock prices since the creation of the index. It was caused by the economic effects of the Covid-19 pandemics and the widespread uncertainty related to the future.

A statistical analysis of the closing prices led to the following observations: (i) the lowest price observed in the series was R$35.31, while the highest was R$115.21; (ii) the average price throughout the whole series was R$63.88; and (iii) the standard deviation throughout the whole series was R$16.74. Nevertheless, it is essential to note that, as the series is non-stationary, both mean and standard deviation values vary widely on different periods.

Figure 15 – BOVA11 frequency distribution of daily returns



Source: elaborated by the author.

To further evaluate the non-stationarity of the dataset, the Augmented Dickey-Fuller Test was used. It showed that this series could be considered non-stationary, as the prices on the period showed various trends on the different periods. Therefore, it is challenging to model its behavior (in relation to stationary series). For this reason, the daily percentage changes (also denominated daily returns on the financial domain) were calculated.

Figure 15 illustrates the frequency distribution of the daily returns for this asset. It is important to note that most of the daily returns are concentrated around 0%. According to the EMH theory, this is expected, which indicates that valuable assets (in terms of daily returns in this case) would become the target of many investors. In this way, the daily return would return, after a period of excess returns, to center around 0%.

It is also important to note the role of outliers, which are related to both extreme market conditions (the steep growth in 2019, due to an increase in market optimism, and the following fall in 2020, due to the impacts of Covid-19) and high market daily volatility (mainly due to the impacts of Covid-19 and market uncertainty), among other aspects. This helps explain the highly negative (-14.57% on 03/12/2020) and positive (+13.40% on 03/13/2020) data points. These could impact considerably on the different models' prediction capacity.

Several other statistical analyses were conducted, such as the analysis of the autocorrelation (ACF) and partial autocorrelation (PACF) functions, both indicating that the daily percentage change and the prices presented an autocorrelation with the immediately previous period, as was expected. As traditional econometrics models, such as

ARIMA, SARIMA, and SARIMAX, are more suited for identifying patterns and predicting stationary time series, several differentiation components' values were analyzed. In the AdaBoost, SVR, and LSTM models, it is common to use the price itself (as observed in the works by Siami-Namini, Tavakoli and Namin (2019) and Eapen, Verma and Bein (2019)), so no differentiation was conducted for those models.

First, the prices dataset was divided into two subsets: the training subset and the testing subset. The testing subset was further divided into two trade scenarios: Trade 1, encompassing the years of 2019 and 2020; and Trade 2, encompassing 2020.

These trades represent two critical scenarios: Trade 1 represents a scenario with upward and downward trends, and Trade 2 represents a scenario with a steep downward trade and high volatility. These are illustrated in Figure 16.

Figure 16 – BOVA11 close prices for final training

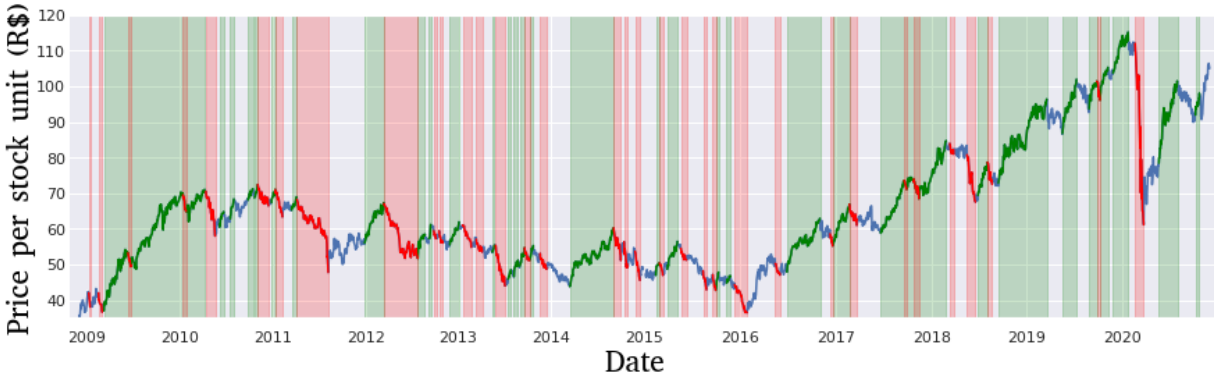
Legend: The final training dataset was divided into training (2008-2018) and test or trades (2019-2020) subsets. The test or trades subset was further divided into two trades: Trade 1 (2019-2020) and Trade 2 (2020).

Source: elaborated by the author.

Figure 17 illustrates the ten days upward and downward trends of the prices of the BOVA11 ETF throughout the dataset used. This was elaborated using the trendet [6] library, a Python library that focuses on trend detection for stock market data. As a result, it is possible to observe that: (i) there are several critical downward trends in 2011, 2012, 2013, and 2018, related to financial crises, the impacts of general elections, the impacts of the exchange rates, among others; (ii) the most crucial downward trend was due to the Covid-19 pandemics in the first semester of 2020; (iii) the years with the highest upward trends were 2009, 2010, 2014, 2018, and 2019, due to exportation

[6] https://trendet.readthedocs.io/

Figure 17 – BOVA11 price chart from 2008 to 2020 with upward and downward trends



Legend: In red: 10 days downward trend; in green: 10 days upward trend.

Source: elaborated by the author.

and commodity prices increases, favorable exchange rates, among others; and (iv) the second semester of 2020 illustrates a potential economic recovery from the impacts of the Covid-10 pandemics in the first semester of 2020.

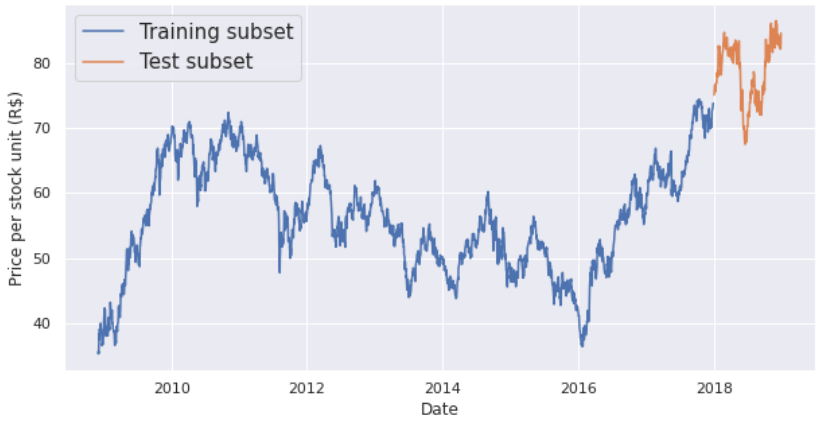
Based on the analysis of Figure 17, it is possible to infer that: (i) the training subset in this work contains both upward and downward trends, an essential aspect for data-driven model training; (ii) the data in the subsets is imbalanced, with high volatility in the year 2020; and (iii) the year of 2020 presents a crucial negative impact on the stock market, which is considerably difficult to predict. Therefore, it is expected that the implemented models will provide higher errors on the test subset for both trading scenarios.

In this module, the training subset (2008-2018) will be used for the final model training, and the test subset will contain the predictions of those models for the years 2019 and 2020. In this way, it is possible to guarantee that there is no data leakage and that the inputs for the trading module (described in section 4.5) are reasonably generated, as in a real-world trading scenario.

However, it is essential to both choose the best model and its correct hyperparameters values. For this reason, an in-depth hyperparameters analysis was conducted with all implemented models. The training subset (2008 to 2018) was further divided into two subsets to conduct the hyperparameters analysis: training (2008 to 2017) and test (2018), as illustrated in Figure 18.

Therefore, the training subset was used for model training, the validation for hyperparameters tuning, and the test for identifying the best model that should be part of the M1 module. The use of the best models' ensembles was also evaluated, and two es-

Figure 18 – BOVA11 close prices for hyperparameters training, divided into train (2008-2017) and test (2018) subsets



Source: elaborated by the author.

sential cross-validation methods for time series prediction: blocking time series split and time series split, both with five folds. It is essential to evaluate different cross-validation methods, as they may allow for distinct pattern recognition, especially for DL time series prediction models, such as the LSTM.

The blocking time series split with five splits divides the training subset into five parts of equal size with no shared data points between them. Each part is then divided into an 80% training subset (the first 80% of the data points) and a 20% testing subset (the last 20% of the data points), with no data shuffling (to preserve its autocorrelation). The model is then trained on the training subset of the first part and tested on this part's testing subset. A new model is then trained on the training subset of the second part, tested on the testing subset of the second part, and so on.

The time series split with five splits is similar because it also divides the training subset into five parts. The division between training and testing subsets is still the same: 80% for training and 20% for testing. Nevertheless, its main difference is that each split incorporates the data points of the previous split. This is done without data shuffling, so the autocorrelation of the data is maintained. Therefore, split 1 contains training (80% of split 1) and testing (20% of split 2) subsets. Split 2 contains a training (first 80% of all data points contained in splits 1 and 2) and a testing subset (the last 20% of all data points contained in splits 1 and 2), and so on.

The hyperparameters analysis results and the final models implementation for the M1 module are discussed in the following subsection.

4.3.2 Model implementation and results

As described in Chapter 3, the main models that were implemented in this module were: (i) econometrics models: ARIMA, SARIMA, and SARIMAX; (ii) ML models: Adaboost, SVR univariate and SVR multivariate; (iii) DL models: LSTM univariate and LSTM multivariate; and (iv) three ensemble models. The models chosen for building the ensembles will be described in this subsection.

It is essential to observe that the grid search using both cross-validation types was the most resource-intensive task, as a considerable amount of models was implemented and tested. This task took around seven days of processing time, considering all models and hyperparameters values. The time for training each final model individually using the chosen hyperparameters and cross-validation type was considerably different for each model, with the SVR and the AdaBoost models taking less than one minute and the LSTM taking less than five minutes. The prediction of any of the final models on the whole validation subset took less than one minute. Therefore, it is possible to observe that the M1 module training and prediction times fulfill the requirements for daily trading in the stock market.

It is possible to observe that the blocking time series splits presented better results than the time series splits. This is an interesting result, as few works in the trading domain literature (especially for the Brazilian market) explore the impacts of using different cross-validation methods for time series. Considering the final models in Table 1, the use of the blocking time series splits presented an MAE 5.53% lower and an MSE 9.07% lower than the use of the time series splits. It is also important to note that the best configurations of the SVR multivariate, SVR univariate, LSTM univariate, AdaBoost, and ARIMA models used the blocking time series cross-validation. It is important to note that the blocking time series implies that data that is closer to the prediction timestep provides more value than data that is further into the past. Both SARIMAX models with TI presented very similar errors.

As for the best models for each category, it is essential to note that the SARIMAX with both time series splits and the use of TI were the best econometrics models, with very similar MAE (0.490 for blocking and 0.489 for time series splits) and the same MSE (0.657). The SARIMAX was also the best overall model. The best ML models were the SVR multivariate (both splits) without TI, with an MAE of 0.524 and an MSE of 0.735. It is interesting to note that the different cross-validation methods had no difference from the SVR, an observation that should be explored in future works.

Table 1 – Results of the final models on the test subset, considering the MAE and MSE metrics

Model group	Model	Best hyperparameter values	Split type	TI	MAE	MSE
Econ.	ARIMA	p:2, d:1, q:1	Block		12.950	270.778
Econ.	ARIMA	p:2, d:0, q:2	TSplit		19.826	519.883
Econ.	SARIMA	p:2, d:0, q:1, P:1, D:0, Q:3, S:3	TSplit		20.007	528.728
Econ.	SARIMA	p:2, d:1, q:2, P:4, D:2, Q:4, S:3	Block		22.718	670.373
Econ.	**SARIMAX**	**p:1, d:0, q:1, P:4, D:0, Q:4, S:2**	**Block**	**X**	**0.490**	**0.657**
Econ.	**SARIMAX**	**p:1, d:0, q:1, P:4, D:0, Q:1, S:3**	**TSplit**	**X**	**0.489**	**0.657**
Econ.	SARIMAX	p:1, d:0, q:1, P:4, D:0, Q:4, S:3	TSplit		0.525	0.693
Econ.	SARIMAX	p:2, d:0, q:2, P:3, D:1, Q:3, S:3	Block		0.876	1.438
ML	AdaBoost	Lr:1, Loss:square, Ne:100	Block		3.177	20.990
ML	AdaBoost	Lr:1, Loss:square, Ne:100	TSplit		3.177	20.990
ML	AdaBoost	Lr:2, Loss:square, Ne:5	Block	X	4.053	33.037
ML	AdaBoost	Lr:2, Loss:square, Ne:5	TSplit	X	4.053	33.037
ML	**SVR_Multi**	**C:10, Ep:0.001, K:linear**	**Block**		**0.524**	**0.735**
ML	**SVR_Multi**	**C:10, Ep:0.001, K:linear**	**TSplit**		**0.524**	**0.735**
ML	SVR_Multi	C:10, Ep:0.01, K:linear	Block	X	0.604	1.149
ML	SVR_Multi	C:10, Ep:0.01, K:linear	TSplit	X	0.604	1.149
ML	SVR_Uni	C:50, Ep:0.01, K:linear	Block		1.556	5.421
ML	SVR_Uni	C:50, Ep:0.01, K:linear	TSplit		1.556	5.421
DL	**LSTM_Multi**	**Ba:32, Nn:1000, Ph:3, E:30**	**TSplit**	**X**	**1.069**	**1.771**
DL	LSTM_Multi	Ba:8, Nn:1000, Ph:1, E:30	TSplit		1.270	2.406
DL	LSTM_Multi	Ba:8, Nn:1000, Ph:1, E:100	Block		1.411	4.070
DL	LSTM_Multi	Ba:32, Nn:1000, Ph:1, E:30	Block	X	1.842	6.018
DL	LSTM_Uni	Ba:2, Nn:1000, Ph:1, E:30	Block		2.069	7.614
DL	LSTM_Uni	Ba:2, Nn:1000, Ph:1, E:30	TSplit		2.227	8.772

Legend: In bold: best model for each category considering the MSE. In green: best models considering all categories. TI: technical indicators; Lr: learning rate; Ne: number of estimators; Ep: epsilon; K: kernel; Ba: batch size; Nn: number of neurons; Ph: past history or window length; E: number of epochs.

Source: elaborated by the author.

The best DL model was the LSTM multivariate with time series splits and TI, with an MAE of 1.069 and an MSE of 1.771. It is interesting to note that, probably due to the dataset's small size, the LSTM provided considerably worse results (MAE and MSE more than 100% higher) than the best SARIMAX and SVR multivariate models.

As for the individual models, it is essential to note that: (i) the best LSTM was the LSTM multivariate; (ii) the best SVR was the SVR multivariate; (iii) the best AdaBoost did not use IT; and (iv) the ARIMA and SARIMA presented the worst results for all metrics, having a considerably higher error.

Concerning the best hyperparameters values, it is possible to observe that: (i) for most ARIMA, SARIMA, and SARIMAX models, the best value for the p component was 1 or 2, for the d component was 0 or 1, and for the q component was 1 or 2; (ii) for SARIMA and SARIMAX, the best value for P was 4, for D was 0, for Q was 3 or 4, and for S was 3; (iii) for AdaBoost, the best value for the loss was the square method; (iv) for the SVR models, the best value for the C was 10, for the epsilon was 0.01 or 0.0001, and for the kernel was linear; (v) for the LSTM models, the best value for the number of neurons was 1,000, for the window length or past history was 1 or 3, and for the number of epochs was 30. The values differed among the model configurations for all the other hyperparameters, and no clear conclusions can be drawn without further studies. Nevertheless, those results can form the basis for further exploration, both in the Brazilian and in other stock markets worldwide.

Therefore, is it possible to infer that the best models were: SARIMAX, SVR multivariate, and LSTM multivariate. The worst model was the SARIMA with the blocking time series split and without TI. Lastly, the best models that used TI were the SARIMAX and the LSTM multivariate.

Table 2 contains the description of the models selected for developing the three ensembles. These are: (i) E1, composed of the best model (SARIMAX) and the best ML model (SVR multivariate); (ii) E2, composed of the best model (SARIMAX) and the best DL model (LSTM multivariate); and (iii) E3, composed of the best ML model (SVR multivariate) and the best DL model (LSTM multivariate). The ensembles' predictions were the simple average of the component models' predictions for each day.

Table 3 contains the best model results in each category and the ensembles on the test subset. It is possible to observe that the E1 (ensemble of SARIMAX and SVR) is the best model, with an MAE of 0.447 and an MSE of 0.498. Models E2 and E3 both provided worse results in comparison to the SARIMAX and SVR models. Lastly, three

Table 2 – Description of the three ensemble models to be evaluated on the M1 module

Code	Description	Model 1	Split	TI	Model 2	Split	TI
E1	Best model and best ML model	SARIMAX	Block	X	SVR_Multi	Block	
E2	Best model and best DL model	SARIMAX	Block	X	LSTM_Multi	TSplit	X
E3	Best ML model and best DL model	SVR_Multi	Block		LSTM_Multi	TSplit	X

Source: elaborated by the author.

Table 3 – Results of the final models on the test subset, considering the MAE and MSE metrics

Model group	Model	MAE	MSE
Econometrics	SARIMAX	0.490	0.657
ML	SVR	0.524	0.735
DL	LSTM	1.069	1.771
Ensembles	**E1 (SARIMAX + SVR)**	**0.447**	**0.498**
Ensembles	E2 (SARIMAX + LSTM)	0.579	0.548
Ensembles	E3 (SVR + LSTM)	0.636	0.655

Legend: In bold: best model considering the MSE. In green: models chosen to be evaluated on the trading module (MT).

Source: elaborated by the author.

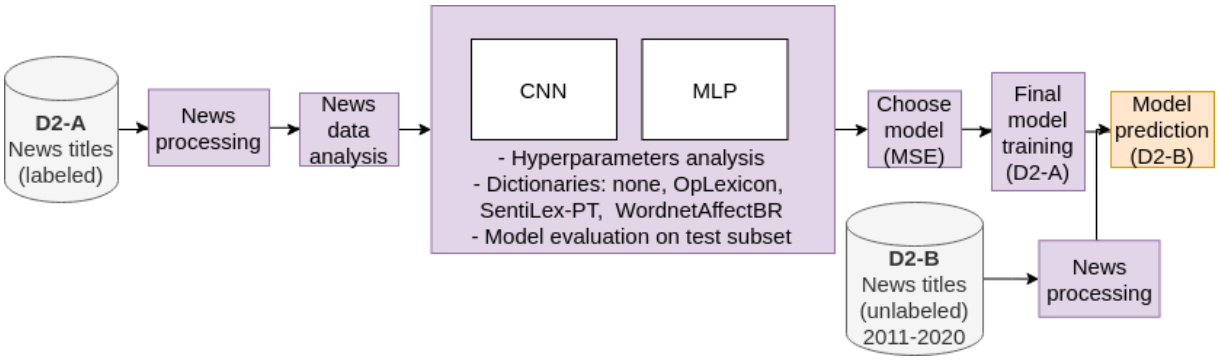
models were chosen for test on the trading module: E1 (best overall model), which will provide stock price predictions for the trading model MT1A; SARIMAX (best individual model), which will provide stock price predictions for the trading model MT1B; and LSTM (state of the art model), which will provide stock price predictions for the trading model MT1C. The SARIMAX model was also used to provide stock price predictions for the trade models MT4 and MT4ta (that use predictions from both M1 and M2 modules as additional inputs).

The following section contains the description of the M2 module and the exploratory data analysis results, the hyperparameters analysis, and the final models implementation.

4.4 M2 - Stock market sentiment prediction module

This section contains the main results of the stock market sentiment prediction module (M2), considering different sentiment analysis models. It is divided into the following sub-

Figure 19 – Main steps for building and evaluating the M2 module - Stock market sentiment prediction



Source: elaborated by the author.

sections: 4.4.1 contains an exploratory analysis of the news titles dataset used for model training, exploring its main characteristics and the division between train, validation, and test subsets; and 4.4.2 describes the main results obtained on the hyperparameters analysis and the final models on the test subset. Lastly, one model was chosen for predicting news titles on an unlabeled dataset. Those sentiment predictions were then used as a feature for three trading models (MT2, MT4, and MT4ta, as described in section 4.5).

This module's main objective is to provide an additional feature for the trading module that could improve its trading results, specifically related to market sentiment. As observed in several works in the literature (NASSIRTOUSSI et al., 2014; SOHANGIR et al., 2018), this could improve the model's results on very volatile markets due to an increase in its responsiveness to events that may severely impact the assets' prices.

Figure 19 illustrates the M2 module. Unlike the M1 module, the predictions on the M2 module are not directly related to a time series. That happens because each datapoint (the title of a news article) is analyzed and predicted individually. For this reason, and since the main objective is to improve already pretrained structures (such as the GloVe word embeddings and the dictionaries used in the implementation), this module uses two datasets. The first (D2-A on Figure 14) is the dataset used to improve those structures, so it is a labeled dataset with validated sentiments for each news title.

The first step of the M2 module is gathering the news titles, eliminating non-relevant news, and labeling the final news. This will then constitute the D2-A dataset. On the second step, the D2-A dataset is processed for use on an NLP model, with the following operations: tokenization, elimination of stopwords, lemmatization, and normalization of the sentiment scores. The third step is to conduct an exploratory analysis of the data, considering statistical analysis and the distribution of the sentiments on the dataset

(contained in section 4.4.1). Step four is related to model implementation, considering the division of the D2-A dataset into training, validation, and test subsets (maintaining the same balance between the sentiment classes on the different subsets), the analysis of hyperparameters, and the use of various sentiment dictionaries. The final models are then evaluated on the test subset, considering MSE as the quality metric.

The fifth step is related to final model training, considering the whole D2-A dataset. Then, the model will be fine-tuned for use on the financial domain dataset. The sixth step is to gather a dataset of unlabeled news titles of the relevant period for the trading model training (in this work, all data available was used, encompassing the periods from 2011 to 2020). This dataset was called D2-B and, in the case of this work, contained 86,674 news titles. The seventh step is to apply the same news processing techniques from step two on the D2-B dataset. The final step is to use the final trained model to predict the sentiments of the news titles. These are then aggregated by day, using a simple average method, and this will constitute the time series of predicted sentiments per day, which will be used as a feature of the trading models that consider the market sentiment.

4.4.1 Exploratory data analysis

As described in section 4.4, the first labeled dataset (D2-A) was used for hyperparameters analysis and model training. It contained 1,134 news titles collected from three relevant and trustworthy market news sources: Valor Econômico, UOL Economia, and InfoMoney. These were already analyzed for relevance in terms of the financial domain. Each news title was labeled independently by three researchers, following the same labeling procedure and guidelines.

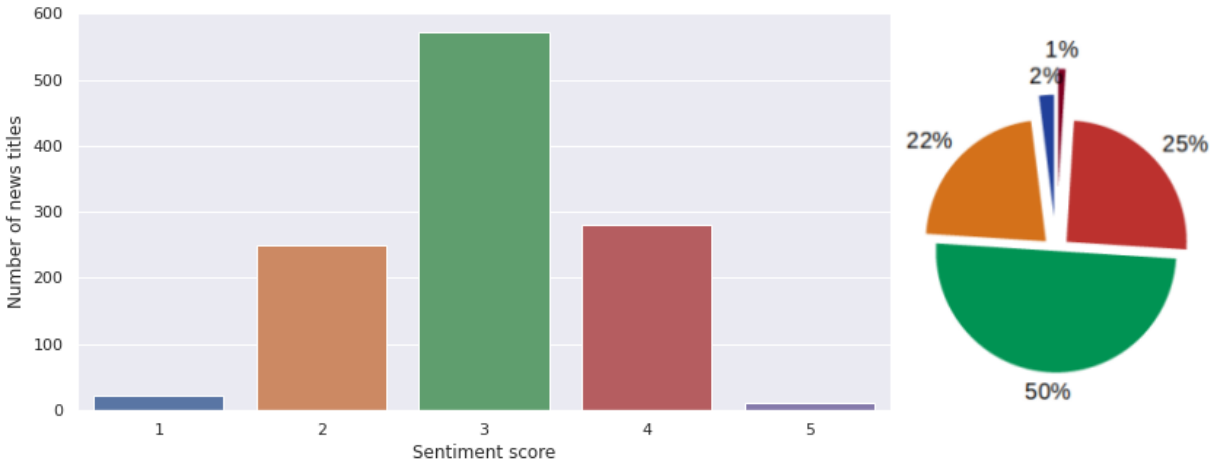
To better describe the sentiment, five scores were used: 1 for very negative, 2 for negative, 3 for neutral, 4 for positive, and 5 for very positive. This is because several sentiment analysis works infer that using only three sentiment classes or scores (negative, neutral, and positive), as used in most of the literature, may not be enough to capture all the sentiments in a text (BOLLEN; MAO; ZENG, 2011; NASSIRTOUSSI et al., 2014).

Then, the final sentiment score was calculated by the simple average of the three researchers' labels' values, rounded up. This is crucial, as news sentiment values may vary considerably within the same day. Aggregating multiple news sentiments in a day may lead to a better evaluation of market sentiment. For example, some days have very negative news for some sectors (for example, negative quarterly results for a bank), while others may have very positive news (for example, an increase in digital sales from a

retailer).

Figure 20 illustrates both the number and percentage of news titles in each sentiment score category. It is important to observe that: (i) 50% of the dataset presented a neutral sentiment; (ii) around 25% of the dataset presented a positive sentiment; (iii) around 22% of the dataset presented a negative sentiment; and (iv) there were almost no news titles on the very positive or very negative extremes. This distribution seems to reflect the reality of financial news during regular periods (without very high volatility), in which there are very few very negative or very positive news, and most are relatively neutral in terms of sentiments. It is crucial to notice that the same methodology can be used for specific sub-domains (for example, commodities) by selecting and analyzing only news in that sub-domain, as was done in Marto et al. (2019). Table 4 contains examples of news titles in each sentiment score or class.

Figure 20 – Market sentiments distribution on the D1-A (labeled) dataset



Legend: Left: sentiment score count by category. Right: sentiment score distribution by category.

Source: elaborated by the author.

As was already described, the D2-A dataset was processed and used for model fine-tuning and training (which will be further explored in section 4.4.2). This training's main objective was to improve the M2 module's capability of predicting the market sentiment based on all the news titles collected each day. Therefore, the use of various sentiment dictionaries was also evaluated. Initial experiments have shown that the GloVe word embedding in Portuguese was the best available word embedding option for this task. In English, the GloVe word embedding is considered a state-of-the-art word embedding, being used in several works related to sentiment analysis, such as in Ferreira et al. (2019) and Mansar et al. (2017).

Table 4 – Number of data points, percentage, and example of sentences for each sentiment score on the D1-A dataset

Sentiment score and class	**Number of data points and percentage of the total dataset**	**Example of a news title on that class**
1 - Very negative	21 / 1.85%	S&P 500 has the worst May in nine years with the increase in commercial conflicts
2 - Negative	250 / 22.05%	Uncertainties in the economy reach the highest level since September, shows FGV
3 - Neutral	572 / 50.44%	Raízen tests solar power generation to provide energy to gas stations and partners
4 - Positive	280 / 24.69%	Good prospects for the harvesting season guarantee an increase in the sales of machinery
5 - Very positive	11 / 0.97%	Sales of hydrous ethanol from sugarcane plants score record sales in February

Source: elaborated by the author.

After training on the D2-A dataset, the final model was then used to predict an unlabeled dataset's market sentiments, denominated D2-B. This simulates the real-life scenario, in which the sentiment is not known and must be predicted in real-time to provide an input that the trading module can use. The D2-B dataset was developed by collecting 113,226 news titles from 2011 to 2020 from another trustworthy source of financial news, the website Investing.com. These were collected with date timestamps.

However, before predicting the news titles' sentiments in the D2-B dataset, a cleaning process was conducted. This was conducted in two stages: (i) elimination of keywords that are related to irrelevant news titles (such as: "summary of today's main news" and "interview with"); and (ii) sampling and analysis of 5% of the whole dataset for identification of additional keywords that indicate news that are irrelevant for the Brazilian stock market as a whole (such as the name of countries that are not relevant in terms of trading or influence on the Brazilian stock market). All these keywords were used to identify and eliminate news titles that were not relevant to this work. This resulted in a final dataset of 86,674 relevant news titles.

The final model trained on the D2-A dataset was then used for predicting the sentiment of all news titles on the cleaned and processed D2-B dataset. Then, the timestamps

were used to aggregate the news sentiment scores per day, using a simple average method for each day (ex: the sentiment score of one day with 30 news titles was the simple average of the predicted sentiment scores of the 30 news titles). In the case of weekends, the news were used to predict the sentiment of the following Monday. Therefore, this module can be applied to real-life trading scenarios, providing sentiment scores aggregated by a defined period (daily in this work, but could be hourly or by the minute). Also, it can be used (without sentiment aggregation) to provide real-time sentiment based on one specific news title.

In the following subsection, the models implementations and results for this module will be analyzed.

4.4.2 Model implementation and results

The main models that were implemented in this module were: MLP and CNN. Both are deep neural networks, and the second is considered one the state of the art models for sentiment analysis on the financial domain, as in the works by Ferreira et al. (2019) and Mansar et al. (2017). As with the M1 module, the first step was conducting the hyperparameters analysis to choose the best hyperparameter values for each model, considering different hyperparameter values and dictionaries. This was implemented with K-fold cross-validation with three balanced folds (considering approximately the same distribution of news titles on the different sentiment scores as in the whole dataset). The training subset contained 80% of the news titles, while the testing subset contained the remaining 20%. This distribution is commonly used in the literature (NASSIRTOUSSI et al., 2014; SOHANGIR et al., 2018).

A grid search was used to evaluate the different model and hyperparameter values combinations, considering the MSE as the quality metric. After choosing the best hyperparameter values, the final models were then trained on the whole training subset and evaluated on the testing subset. Table 5 contains the main hyperparameter values chosen for each model and the MSE on the test subset for the different options of dictionaries.

It is crucial to observe that, as with Module M1, the grid search was the most resource-intensive task for this module. However, as pre-trained word embeddings were used, the total processing time of the grid search took around two days. The training time for each final model individually using the chosen hyperparameters on the D2-A dataset was considerably different between the MLP and the CNN (with the MLP being around 50% faster to train). However, both training times were short, with the CNN taking around

Table 5 – Results of the final models on the test subset, considering the MSE metric

Model	Best hyperparameter values	Dictionary	MSE	Diff. with best model
MLP	Batch size: 32 Number of neurons: 50 Dropout rate: 0 Number of hidden layers: 1	ND	0.634	62.60%
		OL	0.629	61.25%
		SL	0.630	61.38%
		WN	**0.616**	57.98%
		Average of all models	0.627	
CNN	Batch size: 32 Number of neurons: 50 Dropout rate: 0 Number of hidden layers: 1 Filter size: 2 Number of filters: 256	ND	0.400	2.53%
		OL	0.393	0.70%
		SL	0.397	1.66%
		WN	**0.390**	
		Average of all models	0.395	

Legend: In bold: best configuration for each model. In green: best model. ND: no dictionary; OL: OpLexicon; SL: SentLex; WN: WordNetAffectBR.

Source: elaborated by the author.

five minutes to train on the whole dataset. The prediction of any of the final models on the whole D2-B dataset took less than five minutes. The prediction for any individual news title took less than five seconds, and the aggregation of all news in a day for calculating the average sentiment score also took less than five seconds. Therefore, it is possible to observe that the M2 module training and prediction times fulfill the requirements for daily trading in the stock market.

First, it is crucial to observe that the best values for all the shared hyperparameters between both models were the same: the batch size of 32, 50 neurons, a dropout rate of 0, and 1 hidden layer. For the additional hyperparameters for the CNN, the best values were: a filter size of 2 and 256 filters. Also, it is crucial to observe that the best configurations for both models included the use of the WordNetAffectBR dictionary.

The best model was the CNN with the WordNetAffectBR dictionary, with an MSE of 0.390. Although using this dictionary improved the model by only 0.7% in relation to the use of the OpLexicon, this was also observed for the MLP model. The worst models for each category were the models with no dictionaries, and the MLP with no dictionary was the worst model implemented (with an MSE of 0.634, more than 60% higher than the best model). Therefore, it is possible to infer that using this dictionary led to better results than the other dictionaries. Lastly, it is possible to observe that the MSE for the MLP was considerably higher than for the CNN models (0.627 for MLP versus 0.395 for CNN). Therefore, the CNN model is more well suited for this task, as was observed in

Table 6 – Impacts and ranking of importance of the different hyperparameters on the final models, considering the MSE on the test subset

Hyperparameter	CNN		MLP	
	Importance	Impact on MSE	Importance	Impact on MSE
Number of hidden layers	1	179.42%	1	133.41%
Batch size	2	36.27%	4	22.21%
Dropout	3	35.51%	3	40.84%
Number of neurons	4	19.57%	2	51.75%
Number of filters	5	3.51%		
Dictionary	6	2.37%	5	1.48%
Filter size	7	1.69%		

Source: elaborated by the author.

the literature review chapter.

Due to the increasing need for more works on sentiment analysis on the financial domain in Portuguese, an in-depth hyperparameters analysis was conducted to understand better: (i) what hyperparameters impact the most on the final models (considering the ones implemented); (ii) how much they impact on the models; and (iii) if sentiment dictionaries are, as stated in several works in the literature (LOUGHRAN; MCDONALD, 2011; FERREIRA et al., 2019), one of the most important factors to be considered when analyzing sentiments on the financial domain. Table 6 contains the results of this analysis.

The MSE loss function of the deep neural networks was considered for model evaluation, mainly for two reasons: (i) it is widespread in the literature; and (ii) it is also used in the models implemented in section 4.3 (M1 module). It is essential to observe that the factors that impact the most on the MSE for the implemented models (considering the difference in MSE between using the best value for that hyperparameter versus the worst value) are: (i) number of hidden layers (impact of 179.42% for the CNN and of 133.41% for the MLP); (ii) batch size (impact of 36.27% for the CNN and of 22.21% for the MLP); (iii) dropout (impact of 35.51% for the CNN and of 40.84% for the MLP); and (iv) number of neurons (impact of 19.57% for the CNN and of 51.75% for the MLP). It is possible to observe that the dictionaries are one of the hyperparameters that impact the least on the final results (impact of 2.37% for the CNN and of 1.48% for the MLP).

As the dataset is considerably small for a sentiment analysis task, the fact that the number of hidden layers impacted the most in the final results is in line with the literature. This is because this hyperparameter adds significant complexity to the models, increasing

the need for additional data for parameters training (LECUN; BENGIO; HINTON, 2015; JORDAN; MITCHELL, 2015; KOUTSOUKAS et al., 2017). With a considerably large labeled dataset, this hyperparameter may not impact as much on the final results. The batch size can be influenced by both the dataset size and the word embedding used. Therefore, increasing the dataset size could also reduce this hyperparameter's potential impact on the final results.

Using a dropout rate increases the difficulty for the model to train, improving its generalization. However, it also impacts considerably on the results of small datasets, as there are few training examples. Therefore, the difficulty of identifying the correct patterns in the data was increased by using dropout. According to the literature (ZHANG; WALLACE, 2015), high dropout rates tend to cause this behavior. Nevertheless, more experiments are needed to evaluate if this behavior would persist on a larger dataset. Lastly, the number of neurons is also impacted by the size and complexity of the dataset. Therefore, the same observation for the other hyperparameters is also true: increasing the dataset size could decrease this hyperparameter's impact. Notwithstanding, more studies are needed to define those behaviors on different dataset sizes in the financial domain.

Therefore, it is possible to conclude that: (i) the most critical factor to be considered is which model to use (and that the CNN provides significantly better results than the MLP); (ii) the second most important factor is the number of hidden layers (in this work, the higher the number of hidden layers, the worst were the results of both models); (iii) the impact of the other hyperparameters vary considerably on the different models; and (iv) the use of dictionaries (and the choice of which dictionary to use) impact marginally on the final results (although this becomes important in real-life scenarios).

Two possible explanations for the lack of impact of the dictionaries used are: (i) the amount of information contained on the word embeddings versus the dictionaries (the word embedding is contributing much more for the sentiment values due to its characteristics and amount of words and relations considered); and (ii) the fact that the dictionaries are not specific for the financial domain. There are no validated and well-accepted dictionaries for the financial domain in Portuguese. In English, one important financial domain dictionary is the one by Loughran and Mcdonald (2011).

Considering all those factors and each model's results on the test subset, the final model chosen for the M2 module was the CNN with a 300 dimension GloVe word embedding and the WordNetAffectBR dictionary. As described in section 4.4.1, this was used to train on the whole D2-A dataset (labeled) and predict the sentiment values for all news

titles on the D2-B dataset (unlabeled). The resulting predictions were then aggregated by day and used as an input for the trading module (models MT2, MT4, and MT4ta).

In the following section, the implementation and the main results of the trading module will be explored, considering: different models, DRL agents, financial indicators, and two trading scenarios.

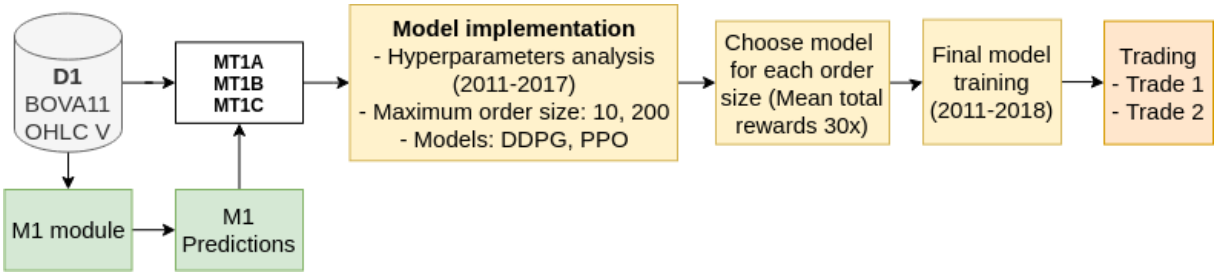
4.5 MT - Trading module

This section contains the main results of the trading module, considering the different modules evaluated. It is divided into the following subsections: 4.5.1 explores the models that consider the stock price prediction signals from the M1 module as a feature (MT1); 4.5.2 explores the models that consider the stock market sentiment signals from the M2 module as a feature (MT2); 4.5.3 explores the models that consider only the OHLCV features (MT3) and the same features with TIs (MT3ta); and 4.5.4 explores the models that consider the stock market prediction and stock market sentiment signals as features (MT4) and the same features with TIs (MT4ta).

All models were executed 30 times (both for training and testing) to better account for the inherent variations of the DRL models' parameters, as recommended in the literature. The MDP considered was the one described in section 4.2, and implemented on the FinRL library. Also, the impact of the maximum order size was explored on those models because initial experiments pointed that, in a scenario with high volatility, order sizes of 100 or 200 (generally used in the literature, as in Liu et al. (2020) presented the worst results, with more exposition to risk and worse results.

For the training time of each model configuration implemented on the trading module, it is crucial to observe that: (i) the total training time for the system was composed of the sum of the training times of the individual modules; and (ii) that after modules M1 and M2 have been trained, the time for training any individual model in the MT module was similar (regardless of its inputs). The most resource-intensive task for this module was the grid search, and the PPO and DDPG agents presented a similar training time. The whole grid search task took around seven days of computation time, and the time for training an individual model took around twenty minutes (depending on the number of timesteps used for training and the number of executions of the model). The model trading (without retraining) for any particular day took less than one minute. Therefore, it is possible to observe that the proposed system fulfills the requirements for daily trading

Figure 21 – Illustration of the steps for the MT1 model and its three variations: MT1A (ensemble SVR multivariate + SARIMAX), MT1B (SARIMAX), and MT1C (LSTM multivariate)



Source: elaborated by the author.

in the stock market.

4.5.1 MT1 - trading model considering price prediction signals

In this subsection, three trading models will be explored, using stock prediction signals from the M1 module as additional features. These models are:

- MT1A: using as inputs OHLCV and the prediction signal (output) from the E1 model (ensemble of the SVR multivariate and the SARIMAX models) from the M1 module;
- MT1B: using as inputs OHLCV and the prediction signal (output) from the SARIMAX model from the M1 module;
- MT1C: using as inputs OHLCV and the prediction signal (output) from the LSTM multivariate model from the M1 module.

Figure 21 illustrates the MT1 trading model and its variations. It is important to note that two DRL agents were evaluated for implementing the trading models: DDPG and PPO. The three main hyperparameters were evaluated for each model, as described in Chapter 3 of this work. Also, two maximum order sizes were evaluated: 10 and 200. The main objective was to identify, through hyperparameters analysis on the subset of prices from 2011 to 2017: the best model and its hyperparameter values for each order size. These were then trained on the data subset from 2011 to 2018 and used for conducting both trading scenarios: Trade 1 (2019-2020) and Trade 2 (2020).

Table 7 contains the results of the hyperparameters analysis of the MT1A, MT1B, and MT1C models on the validation subset (2011 - 2017) for each maximum order size.

It shows the two best models for each combination of model and maximum order size because some combinations, although presenting the best results, had a considerably higher standard deviation than the second-best model. Therefore, in these cases, the second-best model was chosen. The first two models on the table can exemplify this: although the PPO DRL agent presented the highest mean total reward for the MT1A model with a maximum order size of 10, its standard deviation was around nine times higher than the second-best model (DDPG), which had a total reward that was only around 2% lower.

The same can be observed for the MT1B model with a maximum order size of 200: the best model (with the highest total reward of the implemented models) had a total reward of 13,264, while the second-best model (also using the DDPG DRL agent) obtained a total reward of 11,304 (a reduction of around 15%). Nevertheless, the standard deviation of the second-best model's total reward was more than 1000 times lower. In terms of coefficient of variation (CV), while the best model obtained a value of 36.22%, the second observed 0.0003%. As lowering the uncertainty is of primary importance for stock trading, the second model is much more applicable to real-life scenarios than the first, even though its total reward is lower.

It is possible to observe that, in general, the DDPG model presented better results than the PPO model for both maximum order sizes. Using a maximum order size of 200, the mean total reward was considerably higher (around 49% higher considering all model configurations). The high rewards observed are in line with the increase in the prices of the BOVA11 asset during the validation period (2018), explored in section 4.3.1. The MT1B with the DDPG DRL agent and maximum order size of 200 obtained the best results considering both the total reward and its standard deviation, with a total reward of 11,304.02. The second-best model was the MT1A with the DDPG DRL agent and maximum order size of 200, with a total reward of 11,303.49. Lastly, the best MT1C model used the DDPG DRL agent with a maximum order size of 200, obtaining a total reward of 11,276.91.

For the DDPG DRL agent, the best hyperparameter values considering all models were: a batch size of 8 or 128 (depending on the M1 model used to provide the stock prediction signal), 200,000 timesteps, and a buffer size of 10,000. Those hyperparameter values allowed the agents to identify the data patterns better, leading to better actions and a higher total reward. In terms of the number of trades, they all presented similar values.

Table 7 – Results of the hyperparameters analysis for the MT1A, MT1B, and MT1C models on the validation subset, considering the mean total reward on 30 executions and the two best models for each maximum order size

Model	Maximum order size	DRL agent	Best hyperparameter values	Mean total reward	Stdev total reward
MT1A	10	PPO	Ns: 64 / T: 200,000 / Lr: 0.001	7,601.96	429.68
	10	**DDPG**	**Ba: 8 / T: 200,000 / Bu: 10,000**	**7,423.79**	**48.93**
	200	DDPG	Ba: 128 / T: 200,000 / Bu: 1,000	12,625.37	3,941.67
	200	**DDPG**	**Ba: 8 / T: 10,000 / Bu: 10,000**	**11,303.49**	**1.45**
MT1B	10	DDPG	Ba: 128 / T: 10,000 / Bu: 1,000	7,459.57	120.41
	10	**DDPG**	**Ba: 128 / T: 200,000 / Bu: 1,000**	**7,407.93**	**4.00**
	200	DDPG	Ba: 64 / T: 10,000 / Bu: 1,000	13,264.90	4,805.83
	200	**DDPG**	**Ba: 8 / T: 10,000 / Bu: 10,000**	**11,304.02**	**3.31**
MT1C	**10**	**DDPG**	**Ba: 128 / T: 200,000 / Bu: 1,000**	**7,913.95**	**892.53**
	10	PPO	Ns: 8 / T: 200,000 / Lr: 0.001	7,527.33	537.51
	200	**DDPG**	**Ba: 128 / T: 200,000 / Bu: 10,000**	**11,276.91**	**149.61**
	200	DDPG	Ba: 64 / T: 200,000 / Bu: 10,000	10,856.97	4,097.02

Legend: In bold: best configuration for each maximum order size and model.

Source: elaborated by the author.

Table 8 – Final hyperparameters and DRL agents chosen for both maximum order sizes for the MT1A, MT1B, and MT1C models, based on the mean total reward on 30 executions

Model	Maximum order size	DRL agent	Best hyperparameter values	Mean total reward for 30 executions
MT1A	10	DDPG	Ba: 8 / T: 200,000 / Bu: 10,000	7,423.79
	200	DDPG	Ba: 8 / T: 10,000 / Bu: 10,000	11,303.49
MT1B	10	DDPG	Ba: 128 / T: 200,000 / Bu: 1,000	7,407.93
	200	**DDPG**	**Ba: 8 / T: 10,000 / Bu: 10,000**	**11,304.02**
MT1C	10	DDPG	Ba: 128 / T: 200,000 / Bu: 1,000	7,913.95
	200	DDPG	Ba: 128 / T: 200,000 / Bu: 10,000	11,276.91

Legend: In bold, the model with the best mean total reward. Ns: number of steps; T: timesteps; Lr: learning rate; Ba: batch size; Bu: buffer size.

Source: elaborated by the author.

For the PPO DRL agent, the best hyperparameter values considering all models were: number of steps of 8 or 64 (depending on the M1 model used to provide the stock prediction signal), 200,000 timesteps, and a learning rate of 0.001. As was observed for the DDPG DRL agent, these models presented similar values in terms of the number of trades.

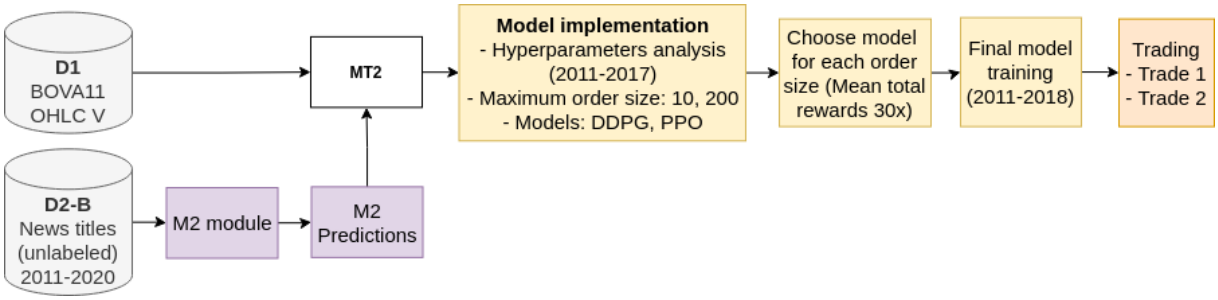
Table 8 contains the final variations of the MT1 models that were implemented for the two trading scenarios. It is possible to observe that the model that presented the best total reward was the MT1B with a maximum order size of 200, closely followed by the MT1A model with a maximum order size of 200. The models with a maximum order size of 10 presented the worst total rewards, and the worst model was the MT1A.

4.5.2 MT2 - trading model considering market sentiment signals

In this subsection, the MT2 trading model will be explored. It uses the OHLCV data and the market sentiment signal from the M2 module, which uses a CNN with a 300 dimension GloVe word embedding and the WordNetAffectBR dictionary to predict the market sentiment based on news titles in Portuguese.

Figure 22 illustrates the MT2 trading model. As with the MT1 trading model, the

Figure 22 – Illustration of the steps for the MT2 model



Source: elaborated by the author.

DDPG and PPO DRL agents were evaluated on this model, evaluating the same hyperparameters and hyperparameter values for the maximum order sizes (10 and 200). The hyperparameters analysis was conducted on the data subset from 2011 to 2017 to find the best model and its hyperparameter values for each order size. These were then trained on the final model training stage on the subset of data from 2011 to 2018 and used for conducting both trading scenarios.

Table 9 contains the results of the hyperparameters analysis of the MT2 model on the validation subset (2011 - 2017) for each maximum order size. It shows the two best models for each combination of model and maximum order size. Unlike what was observed for the MT1 models, the best models in terms of total reward also contained low standard deviations, so they were chosen as the final models for implementation. The best model was the DDPG with a maximum order size of 200, resulting in a mean total reward of 11,333.73. For the maximum size of 10, the PPO DRL agent resulted in the best model, with a mean total reward of 7,360.60.

Unlike the MT1 model, both DRL agents performed well on the MT2 model. Their lower standard deviation is also an important point to be observed. It is possible to infer, based on those results, that the use of a sentiment-related input could reduce the variations on the model. Although further experimentation is needed, considering other markets, assets, and trading frequencies, it is possible to hypothesize that this could be related to the model's better capability to understand the market structure or its relevant patterns. This model could be interesting for risk-averse players or high volatility scenarios if that is the case.

The maximum order size of 200 presented a reward that was around 54% higher than the one for the maximum order size of 10. All rewards were positive and in line with what was observed for model MT1 and its variations. For the DDPG DRL agent, the best hyperparameter values considering all models were: a batch size of 8, 10,000 or 100,000

Table 9 – Results of the hyperparameters analysis for the MT2 model on the validation subset, considering the mean total reward on 30 executions and the two best models for each maximum order size

Model	Maximum order size	DRL agent	Best hyperparameter values	Mean total reward	Stdev total reward
MT2	**10**	**PPO**	**Ns: 8 / T: 100.000 / Lr: 0.00025**	**7,360.60**	**388.17**
	10	DDPG	Ba: 8 / T: 10.000 / Bu: 10.000	7,092.61	1,661.48
	200	**DDPG**	**Ba: 8 / T: 100.000 / Bu: 10.000**	**11,333.73**	**91.90**
	200	PPO	Ns: 8 / T: 100.000 / Lr: 0.001	11,303.15	0.00

Legend: In bold: best configuration for each maximum order size and model.

Source: elaborated by the author.

Table 10 – Final hyperparameters and DRL agents chosen for both maximum order sizes the MT2 model, based on the mean total reward on 30 executions

Model	Maximum order size	DRL agent	Best hyperparameter values	Mean total reward for 30 executions
MT2	10	PPO	Ns: 8 / T: 100,000 / Lr: 0.00025	7,360.60
	200	**DDPG**	**Ba: 8 / T: 100,000 / Bu: 10,000**	**11,333.73**

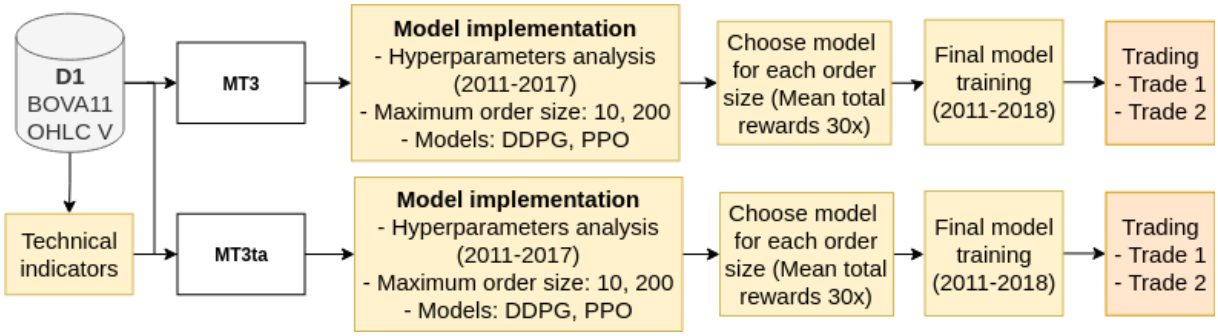
Legend: In bold, the model with the best mean total reward.

Source: elaborated by the author.

timesteps, and a buffer size of 10,000. For the PPO DRL agent, the best hyperparameter values considering all models were: number of steps of 8, 100,000 timesteps, and a learning rate of 0.001 or 0.00025. For both DRL agents, the number of trades on the subset was similar.

Table 10 contains the final MT2 model implemented for the two trading scenarios for both maximum order sizes. The model that presented the best total reward was using the DDPG DRL agent and a maximum order size of 200.

Figure 23 – Illustration of the steps for the MT3 and MT3ta models



Source: elaborated by the author.

4.5.3 MT3 and MT3ta - trading models considering only OHLCV

In this subsection, the MT3 and MT3ta trading models will be explored. The MT3 uses only the OHLCV data as its inputs. These are the most used inputs in the RL trading literature, as observed by Meng and Khushi (2019) and Fischer (2018). The MT3ta considers TI's use, being also explored in the RL trading literature (MENG; KHUSHI, 2019; FISCHER, 2018). From the vast number of options of TI, the most relevant ones were chosen in this thesis (the same ones used for evaluating the use of TI on the M1 module): (i) for volume: ADI; (ii) for volatility: BB; (iii) for identifying trends: MACD; and (iv) for momentum: RSI, stochastic RSI, and WR.

Figure 23 illustrates the MT3 and MT3ta trading models. As with the other trading models implemented, the DDPG and PPO DRL agents were evaluated on this model, analyzing the same hyperparameters and hyperparameter values for both maximum order sizes (10 and 200). The hyperparameters analysis was conducted on the data subset from 2011 to 2017 to find the best model and its hyperparameter values for each order size. These were then trained on the final model training stage on the subset of data from 2011 to 2018 and used for conducting both trading scenarios.

Table 11 contains the results of the hyperparameters analysis of the MT3 and MT3ta models on the validation subset (2011 - 2017) for each maximum order size. It shows the two best models for each combination of model and maximum order size. For both models, the best DRL agent for the maximum order size of 10 was PPO, and for the maximum order size of 200 was the DDPG agent. The best model was the MT3ta with the DDPG agent and a maximum order size of 200, while the worst model was the MT3 with the PPO agent and a maximum order size of 10. However, even though the MT3ta presented a higher total reward (indicating that TI's use may improve pattern extraction and decision making), the standard deviation was considerably higher. The CV for MT3

Table 11 – Results of the hyperparameters analysis for the MT3 and MT3ta models on the validation subset, considering the mean total reward on 30 executions and the two best models for each maximum order size

Model	Maximum order size	DRL agent	Best hyperparameter values	Mean total reward	Stdev total reward
MT3	**10**	**PPO**	**Ns: 8 / T: 200,000 / Lr: 0.001**	**7,490.73**	**171.27**
	10	PPO	Ns: 8 / T: 200,000 / Lr: 0.00025	7,406.62	0,00
	200	**DDPG**	**Ba: 128 / T: 10,000 / Bu: 1,000**	**11,300.00**	**18.61**
	200	DDPG	Ba: 8 / T: 200,000 / Bu: 100,000	11,286.86	48.87
MT3ta	**10**	**PPO**	**Ns: 128 / T: 200,000 / Lr: 0.00025**	**7,656.00**	**562.39**
	10	DDPG	Ba: 128 / T: 10,000 / Bu: 1,000	7,175.22	571.53
	200	**DDPG**	**Ba: 8 / T: 200,000 / Bu: 10,000**	**11,927.69**	**1,766.02**
	200	DDPG	Ba: 8 / T: 100,000 / Bu: 1,000	11,303.15	0.01

Legend: Ns: number of steps; T: timesteps; Lr: learning rate; Ba: batch size; Bu: buffer size.

Source: elaborated by the author.

was around 0.006%, while for the MT3ta, it was around 7.62%, indicating that the use of TI may increase trading risks. Nevertheless, more exploration is needed to understand better and verify those observations, considering multiple assets and markets.

The maximum order size of 200 presented a reward that was around 51% higher than the one for the maximum order size of 10 for MT3, and 56% higher for MT3ta. All rewards were positive and in line with what was observed for the other trading models. For the DDPG DRL agent for both the MT3 and MT3ta models, the best hyperparameter values differed. There was no clear best choice. This may be because the MT3 and MT3ta models' behavior is different, even though in the RL trading literature, the impacts of TI on the trading results are not explored in depth. The same is observed for the PPO DRL agent, except for the number of timesteps: for these models, a higher number of timesteps (200,000) led to the best results. As with other models, the number of trades on the subset was similar.

Table 12 contains the final MT3 and MT3ta models implemented for the two trading scenarios for both maximum order sizes. The model that presented the best total reward

Table 12 – Final hyperparameters and DRL agents chosen for both maximum order sizes for the MT3 and MT3ta models, based on the mean total reward on 30 executions

Model	Maximum order size	DRL agent	Best hyperparameter values	Mean total reward for 30 executions
MT3	10	PPO	Ns: 8 / T: 200,000 / Lr: 0.001	7,490.73
	200	DDPG	Ba: 128 / T: 10,000 / Bu: 1,000	11,300.00
MT3ta	10	PPO	Ns: 128 / T: 200,000 / Lr: 0.00025	7,656.00
	200	**DDPG**	**Ba: 8 / T: 200,000 / Bu: 10,000**	**11,927.69**

Legend: In bold, the model with the best mean total reward. Ns: number of steps; T: timesteps; Lr: learning rate; Ba: batch size; Bu: buffer size.

Source: elaborated by the author.

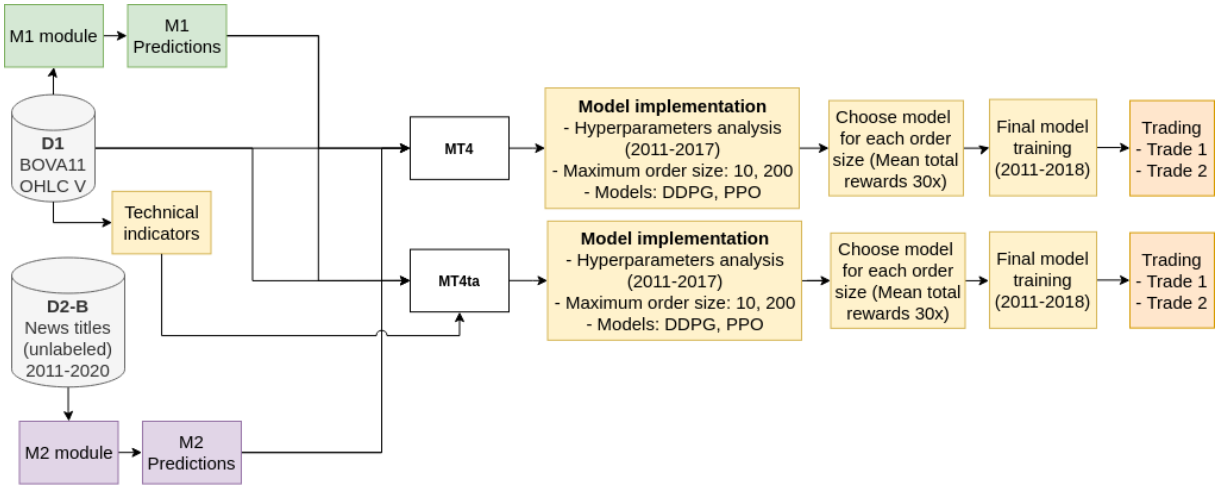
was the MT3ta using the DDPG DRL agent and maximum order size of 200.

4.5.4 MT4 and MT4ta - trading models considering price prediction and market sentiment signals

In this subsection, the MT4 and MT4ta trading models will be explored. The MT4 uses as its inputs: the OHLCV data, the stock prediction signal from the M1 module, and the market sentiment signal from the M2 module. According to the extensive literature reviewed by Meng and Khushi (2019) and Fischer (2018), there are very few works that explore RL trading models using those inputs, especially the sentiment signal. However, those works do not compare the models with alternatives without part of the inputs, making it difficult to pinpoint the most critical component: the price prediction or the sentiment prediction. The MT4ta also considers those inputs and the use of TI. This model is not explored in-depth in the literature. The TI used is the same as in the model MT3ta.

Figure 24 illustrates the MT4 and MT4ta trading models. As with the other trading models implemented, the DDPG and PPO DRL agents were evaluated on this model, analyzing the same hyperparameters and hyperparameter values for both maximum order sizes (10 and 200). The hyperparameters analysis was conducted on the data subset from 2011 to 2017 to find the best model and its hyperparameter values for each order size. These were then trained on the final model training stage on the subset of data from 2011

Figure 24 – Illustration of the steps for the MT4 and MT4ta models



Source: elaborated by the author.

to 2018 and used for conducting both trading scenarios.

Table 13 contains the results of the hyperparameters analysis of the MT4 and MT4ta models on the validation subset (2011 - 2017) for each maximum order size. It shows the two best models for each combination of model and maximum order size. For MT4, the PPO DRL agent resulted in the best performance (considering both the mean total reward and its standard deviation) for both maximum order sizes. For MT4ta, the opposite was observed: the DDPG was the best DRL agent. The best model was the MT4ta with the DDPG agent and a maximum order size of 200, while the worst model was the MT4 with the DDPG agent and a maximum order size of 10. This indicates that the TI may have improved the models' results while also decreasing its risks (illustrated by the mean total reward's lower standard deviation). Nevertheless, as with the observations regarding models MT3 and MT3ta, more exploration is needed to understand better and verify those observations, considering multiple assets and markets.

The maximum order size of 200 presented a reward around 41% higher than the one for the maximum order size of 10 for MT4 and 51% higher for MT4ta. This was in line with what was observed for the other trading models implemented. All rewards were positive and in line with what was observed for the other trading models. As was observed on the MT3 and MT3ta models, the best hyperparameter values differed for the models and the DRL agents. There was no clear best choice besides a batch size of 8 for the DDPG, and 100,000 timesteps or higher for all models, except for MT4ta with DDPG for a maximum order size of 200. This may be because the behavior of the MT4 and MT4ta models are different, as observed in those models' results.

Table 13 – Results of the hyperparameters analysis for the MT4 and MT4ta models on the validation subset, considering the mean total reward on 30 executions and the two best models for each maximum order size

Model	Maximum order size	DRL agent	Best hyperparameter values	Mean total reward	Stdev total reward
MT4	**10**	**PPO**	**Ns: 8 / T: 100,000 / Lr: 0.001**	**7,506.22**	**243.98**
	10	DDPG	Ba: 64 / T: 100,000 / Bu: 1,000	7,458.37	300.74
	200	DDPG	Ba: 64 / T: 200,000 / Bu: 100,000	10,828.06	6,187.92
	200	**PPO**	**Ns: 8 / T: 200,000 / Lr: 0.0005**	**10,587.92**	**1,943.54**
MT4ta	10	PPO	Ns: 8 / T: 100,000 / Lr: 0.0005	7,518.13	314.58
	10	**DDPG**	**Ba: 8 / T: 200,000 / Bu: 10,000**	**7,475.29**	**201.31**
	200	DDPG	Ba: 128 / T: 100,000 / Bu: 10,000	12,272.43	3,562.22
	200	**DDPG**	**Ba: 8 / T: 10,000 / Bu: 100,000**	**11,304.80**	**4.33**

Legend: In bold: best configuration for each maximum order size and model. Ns: number of steps; T: timesteps; Lr: learning rate; Ba: batch size; Bu: buffer size.

Source: elaborated by the author.

Another possible explanation is that the models had more difficulties identifying patterns considering all the various features used, partly explaining the high CV for those models. This is a very interesting topic that could be explored in more detail in future works, considering different markets, assets, and trading scenarios. As with the other trading models, the number of trades on the subset was similar.

Table 14 contains the final MT4 and MT4ta models implemented for the two trading scenarios for both maximum order sizes. The model that presented the best total reward was the MT4ta using the DDPG DRL agent and maximum order size of 200.

The following section contains the final results of all trading models on the two trading scenarios, evaluating their impacts on the six financial indicators in relation to the BH strategy.

Table 14 – Final hyperparameters and DRL agents chosen for both maximum order sizes for the MT4 and MT4ta models, based on the mean total reward on 30 executions

Model	Maximum order size	DRL agent	Best hyperparameter values	Mean total reward for 30 executions
MT4	10	PPO	Ns: 8 / T: 100,000 / Lr: 0.001	7,506.22
	200	PPO	Ns: 8 / T: 200,000 / Lr: 0.0005	10,587.92
MT4ta	10	DDPG	Ba: 8 / T: 200,000 / Bu: 10,000	7,475.29
	200	**DDPG**	**Ba: 8 / T: 10,000 / Bu: 100,000**	**11,304.80**

Legend: In bold, the model with the best mean total reward. Ns: number of steps; T: timesteps; Lr: learning rate; Ba: batch size; Bu: buffer size.

Source: elaborated by the author.

4.6 Final models comparison

This subsection contains the final models comparison. It is divided into three parts: (i) an in-depth analysis of the different DRL agents and hyperparameters for the final models, considering total reward and CV on the test subset (2018); and (ii) the analyses of the implementations of the final models on the two trading scenarios.

Table 15 contains the results of the final models implemented on the test subset, considering both mean total reward and CV. In bold, the most important models, which contain a low risk (CV lower than 2%) and a high total reward (more than 11,000), are highlighted. Those models, in theory, should present better results in trading scenarios. This will be further explored through this section.

As was explored in section 4.5, the maximum order size of 200 led to the best results for all models. The DDPG DRL agent was also more prevalent than the PPO on the final models due to its better results on the test subset. This indicates that it could better identify the data patterns, which led to better trades and a better total reward. Three of the five PPO models implemented on the final models presented a CV higher than 5%. The highest CV (18.36%) was observed on the MT4 model with the PPO DRL agent and a maximum order size of 200.

The best model was the MT2 with the DDPG DRL agent and the maximum order size of 200. However, the MT1A, MT1B, MT1C, MT3, and MT4ta with maximum order

Table 15 – Final models implemented on the trading module and their results in terms of mean total reward and coefficient of variation (CV) on the test subset (2018)

Model	Maximum order size	DRL agent	Mean total reward	CV
MT1A	10	DDPG	7,423.79	0.66%
	200	**DDPG**	**11,303.49**	**0.01%**
MT1B	10	DDPG	7,407.93	0.05%
	200	**DDPG**	**11,304.02**	**0.03%**
MT1C	10	DDPG	7,913.95	11.28%
	200	**DDPG**	**11,276.91**	**1.33%**
MT2	10	PPO	7,360.60	5.27%
	200	**DDPG**	**11,333.73**	**0.81%**
MT3	10	PPO	7,490.73	2.29%
	200	**DDPG**	**11,300.00**	**0.16%**
MT3ta	10	PPO	7,656.00	7.35%
	200	DDPG	11,927.69	14.81%
MT4	10	PPO	7,506.22	3.25%
	200	PPO	10,587.92	18.36%
MT4ta	10	DDPG	7,475.29	2.69%
	200	**DDPG**	**11,304.80**	**0.04%**

Legend: In bold: models that have a coefficient of variation lower than 2% and a mean total reward higher than 11,000. Ns: number of steps; T: timesteps; Lr: learning rate; Ba: batch size; Bu: buffer size.

Source: elaborated by the author.

sizes of 200, presented very similar results in terms of mean total reward. The model with the highest total reward (11,927.69), MT3ta with a maximum order size of 200, also presented a considerably high CV (14.81%), difficulting its use in real life scenarios.

Table 16 compares the DRL agents implemented on the final models in terms of the number and percentage of winning models in relation to the total number of final models implemented and mean total reward and CV. It is crucial to note that the DDPG was the best DRL agent considering all the final models, not only because it was the DRL agent used by 11 of the 16 models, but also because its total reward was 23.12% higher and its CV was 60% lower than the PPO DRL agent. This indicates that the DDPG DRL agent may obtain better trading results with lower risks than the PPO DRL agent on RL trading tasks. This is an interesting result, as few works in the literature explore the impact of using different DRL agents, hyperparameters values, and maximum order sizes on RL trading.

Table 17 contains a similar comparison to Table 16 but separating the results by

Table 16 – Comparison of the DRL agents implemented, considering all the final models on the test subset

DRL agent	Winners (count)	Winners (%)	Mean total reward	CV
DDPG	**11**	**69%**	**9,997.42**	**2.90%**
PPO	5	31%	8,120.29	7.30%

Legend: In bold: best DRL agent.

Source: elaborated by the author.

Table 17 – Comparison of the DRL agents implemented, considering all the final models on the test subset, separated by maximum order size

	Maximum order size: 10			Maximum order size: 200		
DRL agent	Winners (count / percentage)	Mean total reward	CV	Winners (count / percentage)	Mean total reward	CV
DDPG	**4 / 50%**	**7,604.34**	**4.88%**	**7 / 88%**	**11,392.95**	**2.46%**
PPO	4 / 50%	7,503.39	4.54%	1 / 12%	10,587.92	18.36%

Legend: In bold: best DRL agent.

Source: elaborated by the author.

the two maximum order sizes. This is a fundamental analysis because, on very volatile markets, limiting the maximum order size may lead to more risk-averse strategies (as the model can buy only a small number of stocks per day). On the other hand, this may reduce the potential gains in a growing market, as shown in Table 17. It is also essential to observe that the DDPG DRL agent presented a higher mean total reward on average than the PPO DRL agent. Also, the DDPG DRL agent showed a considerably lower CV than the PPO for a maximum order size of 200 (2.46% for DDPG versus 18.36% for PPO).

For the maximum order size of 10, the PPO DRL agent showed a slightly lower CV (4.88% for PPO versus 4.54% for DDPG). Those results point out that the DDPG model might be considerably better on optimistic scenarios, leading to higher gains but slightly riskier on volatile scenarios. It is also important to observe that, in the test subset (2018), there was a growing trend throughout the price time series, which may have influenced those results. This is an additional reason for evaluating the models in different scenarios. This point will be discussed in the analysis of the results of the two trading scenarios.

Table 18 – Hyperparameters analysis for the winning models for the DDPG DRL agent

Hyperparameter	Values	Mean total reward	CV	Winners (count)
Batch size	**8**	**10,.296.12**	**2.72%**	**7**
	64	0.00	0.00%	0
	128	9,474.70	3.21%	4
Timesteps	10,000	11,303.08	0.06%	4
	100,000	**11,333.73**	**0.81%**	**1**
	200,000	8,904.26	5.14%	6
Buffer size	1,000	8,873.96	3.83%	3
	10,000	10,292.13	2.91%	7
	100,000	**11,304.80**	**0.04%**	**1**

Legend: In bold: best hyperparameter value considering both mean total reward and the CV for each hyperparameter.

Source: elaborated by the author.

Table 18 contains the hyperparameters analysis's main results for the winning models (the ones that were chosen to be the final models) for the DDPG DRL agent. The best hyperparameters values were: a batch size of 8, 100,000 timesteps, and a buffer size of 100,000. Based on those results, it is possible to infer that smaller batch sizes and larger buffer sizes provide better results. It would be interesting to further explore lower values for batch size (such as 4 and 2) and higher values for buffer sizes (such as 200,000) to evaluate their impacts on the final model results. As for the number of timesteps, it is important to note that, after 100,000 timesteps, the model seemed to start overfitting, which may explain the reduced total reward. As for the CV, it is important to note that it was considerably lower for all the best hyperparameters values (less than 3%).

Table 19 contains the hyperparameters analysis's main results for the winning models for the PPO DRL agent. The best hyperparameters values were: a number of steps of 8, 200,000 timesteps, and a learning rate of 0.0005. It is crucial to note that this DRL agent presented much higher CVs for all hyperparameter values than the DDPG DRL agent. For the best value for the learning rate, the CV was 18.36%, a considerably high value (making it very likely unsuitable for financial trading due to the high variation in the model's results).

A higher number of timesteps improved the model's rewards but also increased the CV. Although more experiments with different datasets are needed, it is possible to infer that this model may be identifying incorrect patterns, leading to worse results by part of its executions (what explains the high CV on several executions). As for the number of

Table 19 – Hyperparameters analysis for the winning models for the PPO DRL agent

Hyperparameter	Values	Mean total reward	CV	Winners (count)
Number of steps	**8**	**8,236.37**	**7.29%**	**4**
	64	0.00	0.00%	0
	128	7,656.00	7.35%	1
Timesteps	10,000	0.00	0.00%	0
	100,000	**7,433.41**	**4.26%**	**2**
	200,000	**8,578.22**	**9.33%**	**3**
Learning rate	0.00025	7,508.30	6.31%	2
	0.0005	**10,587.92**	**18.36%**	**1**
	0.001	**7,498.48**	**2.77%**	**2**

Legend: In bold: best hyperparameter value considering both mean total reward and the CV for each hyperparameter.

Source: elaborated by the author.

steps, lower values tended to provide better results, although also with a high CV. The learning rate was the hyperparameter that most impacted on both the total rewards and the CV, making it the most critical hyperparameter to tune for this DRL agent in this specific use case. Also, as was observed before, the models' total rewards with the PPO DRL agent were considerably lower, mainly because the chosen PPO models were used on the maximum order size of 10.

In the following paragraphs, the two trading scenarios' results will be explored, considering all final trading models, the six financial metrics chosen for analysis, and the BH strategy.

Table 20 shows the results of all the trading models and the BH strategy for Trade 1 (2019-2020). It is important to note that the six financial indicators can be divided into four groups: (i) returns-related metrics: annual returns and cumulative returns; (ii) volatility-related metrics: annual volatility and stability; (iii) risk-related metrics: Sharpe ratio; and (iv) loss-related metrics: maximum drawdown. Therefore, the results will be analyzed considering the groups and the individual metrics.

First, it is possible to observe that for the returns-related and risk-related metrics, only the MT1C model with a maximum order size of 200 beat the BH. This indicates that, for this specific trading scenario (in which there is a considerable difference between peaks and valleys on prices and both upward and downward trends), this is the only suitable model for an investor that is not risk-averse. The results were significantly better on the

Table 20 – Final results for the trading models for Trade 1 (2019-2020) and the BH strategy, considering the average of the financial metrics on 30 executions

Model	MOS	DRL	AR	CR	AV	SR	ST	MD
MT1A	10	DDPG	3.48%	5.79%	0.267	0.272	0.144	-36.26%
	200	DDPG	5.46%	9.18%	0.299	0.371	0.121	-39.15%
MT1B	10	DDPG	4.08%	6.79%	0.323	0.320	0.041	-43.97%
	200	DDPG	5.97%	9.98%	0.331	0.345	0.034	-43.88%
MT1C	10	DDPG	3.34%	5.55%	0.298	0.195	0.100	-40.72%
	200	**DDPG**	**8.53%**	**14.77%**	0.278	**0.482**	**0.219**	-32.85%
MT2	10	PPO	1.71%	2.83%	**0.187**	0.056	0.205	**-26.10%**
	200	DDPG	6.27%	10.63%	0.304	0.385	0.122	-39.05%
MT3	10	PPO	3.04%	5.04%	0.274	0.056	0.128	-37.44%
	200	DDPG	5.85%	9.98%	0.283	0.358	0.155	-36.27%
MT3ta	10	PPO	3.14%	5.22%	0.300	0.285	0.053	-41.27%
	200	DDPG	6.22%	10.43%	0.308	0.465	0.099	-40.14%
MT4	10	PPO	3.22%	5.34%	0.297	0.292	0.047	-40.65%
	200	PPO	3.70%	6.22%	0.251	0.193	0.196	-33.40%
MT4ta	10	DDPG	2.76%	4.59%	0.264	0.212	0.122	-36.22%
	200	DDPG	5.41%	9.23%	0.271	0.174	0.186	-35.48%
BH			7.13%	11.83%	0.356	0.375	0.010	-46.92%

Legend: In bold: best model considering all metrics. In green: models that performed better than the BH model for the specific metric. In red: worst model for the specific metric. MOS: maximum order size; DRL: DRL agent; AR: annual returns; CR: cumulative returns; AV: annual volatility; SR: Sharpe ratio; ST: stability; MD: maximum drawdown.

Source: elaborated by the author.

financial trading scenario, with the MT1C presenting annual returns of 8.53% (versus 7.13% for BH) and cumulative returns of 14.77% (versus 11.83% for BH). This model also showed the best results regarding stability: 0.219 (versus 0.010 for BH).

As for the risk-related metrics, this model also showed better results than the BH, with a Sharpe ratio of 0.482 (versus 0.375 for BH). Several other models also presented a Sharpe ratio higher than BH: MT2 with a maximum order size of 200 (0.385) and MT3ta with a maximum order size of 200 (0.465).

Nevertheless, the most exciting results are related to the volatility-related metrics and the loss-related metrics. For both categories, all models performed better than the BH strategy, with the MT2 with an order size of 10 providing significantly better results than the BH strategy: annual volatility of 0.187 (versus 0.356 for BH), the stability of 0.205 (versus 0.010 for BH), and a maximum drawdown of -26.10% (versus -46.92%).

However, this model also presented the worst results in related to the returns-related and risk-related metrics. Therefore, its use is not recommended in this trading scenario, as the order size limited its potential results.

As for the use of TIs on the MT3ta and MT4ta, it improved the return-related metrics (especially for the models with a maximum order size of 200) and the risk-related metrics for the MT3ta, in relation to the respective models that did not use TI (MT3 and MT4). Notwithstanding, the results on the volatility-related and the loss-related metrics were considerably worse by using TI. Therefore, it is possible to infer that, at least for this trading scenario and the maximum order sizes evaluated, TI's use as an input for the DRL agent did not improve its results, leading to an increase in volatility and losses.

The use of the predictions of the M1 model as inputs for the DRL agent, especially from the LSTM (MT1C model), provided better results in relation to the standard approach of using only OHLCV inputs (MT3) ou OHLCV with TI (MT3ta) considering mainly the returns-related and risk-related metrics. It is important to observe that the MT1A and MT1B models obtained worse results for the volatility-related and loss-related metrics in relation to the standard approach methods. Therefore, it is possible to infer that the M1 module with the LSTM model improves the overall trading results of the DRL agent, but the same does not apply to the ensemble approach (MT1A) and the SARIMAX model (MT1B).

As for the M2 module, it is important to observe that the model with a maximum order size of 200 presented better results than the MT3 and MT3ta in terms of the returns-related metrics. It also presented better results for the risk-related metrics in relation to the MT3 and for the volatility-related and loss-related metrics in relation to the MT3ta. Therefore, it is possible to observe that the market sentiment prediction helped to better trade in a scenario with volatility in relation to TI use. It also provided better returns.

As for the use of both M1 and M2 for providing features for the DRL agents, it is possible to observe, by analyzing the MT4 and MT4ta models, that they provide better results on the returns-related metrics and the loss-related metrics in relation to the MT3. In relation to the MT3ta, the MT4 and MT4ta models provide better results for the volatility-related metrics and loss-related metrics but worse results for the returns-related and risk-related metrics. Therefore, they may be suited for more risk-averse investors (in relation to the MT3ta model).

It can be concluded from this analysis that: (i) the maximum order size of 10 is not suitable for this trading scenario, as was expected from the experiments and anal-

Table 21 – Comparison and statistical analysis of the best models with the BH strategy for Trade 1 (2019-2020), considering financial metrics on 30 executions

Model	MOS	DRL	AR	CR	AV	SR	ST	MD
MT1C	200	DDPG	8.53%	14.77%	0.278	0.482	**0.219**	**-32.85%**
MT2	10	PPO	**1.71%**	**2.83%**	**0.187**	**0.056**	**0.205**	**-26.10%**
BH			7.13%	11.83%	0.356	0.375	0.010	-46.92%

Legend: In bold: values with statistical difference (p <0.01% using Student's T-test) from the BH model. In green: models that performed better than the BH model for the specific metric. In red: worst model for the specific metric. MOS: maximum order size; DRL: DRL agent; AR: annual returns; CR: cumulative returns; AV: annual volatility; SR: Sharpe ratio; ST: stability; MD: maximum drawdown.

Source: elaborated by the author.

yses conducted on section 4.5; (ii) the MT1C with maximum order size of 200 was the best trading model evaluated; (iii) the MT1C with maximum order size of 200 presented significantly better results than the BH in this trading scenario; (iv) all trading models presented better results for the volatility-related and loss-related metrics, in relation to the BH strategy; (v) although the MT2 with maximum order size of 10 presented the best results for the volatility-related and loss-related metrics, it also presented the worst results for the returns-related and risk-related metrics (being a model that is not suited for this trading scenario); (vi) the use of the M1 and M2 modules, both in isolation (MT1C and MT2) and together (MT4 and MT4ta) presented better results, with some models being more indicated for risk-averse investors.

Table 21 contains further analysis of the models that presented the best (MT1C with a maximum order size of 200) and worst (MT2 with a maximum order size of 10) for the evaluated metrics, as well as the baseline used (the BH strategy). The Student's T-test statistical analysis was conducted to evaluate which value differences were statistically significant in relation to the BH strategy.

It is possible to observe in Table 21 that, for the MT1C, the stability (a volatility-related metric) and the maximum drawdown (a loss-related metric) were statistically lower than the BH, while the other indicators were not statistically different. This means that this model may provide similar returns with a lower probability of loss than the BH strategy, making it an important candidate for trading strategies with the studied asset. As for the MT2, it is possible to observe that its results were statistically worse than the BH strategy, especially for the returns-related and risk-related metrics. That means

that this is not a good model for a trading strategy, even though its volatility-related and loss-related metrics are statistically better than the BH strategy because the MT2's returns are too low.

Table 22 shows the results of all the trading models and the BH strategy for Trade 2 (2020). First, it is essential to observe that this scenario is considerably different: it is much more volatile than Trade 1, and a considerable amount of its data points are on a steep downward trend. Therefore, it is expected that models that can adapt faster to market changes and deal better with daily volatility (or that make smaller investments at each timestep) will provide better results in a trading scenario with those characteristics.

With a few exceptions for the risk-related metric (MT1A, MT1C, MT2, MT3ta, and MT4 with maximum order sizes of 200), all models presented better results than the BH strategy for all metrics evaluated.

Table 22 – Final results for the trading models for Trade 2 (2020) and the BH strategy, considering the average of the financial metrics on 30 executions

Model	MOS	DRL	AR	CR	AV	SR	ST	MD
MT1A	10	DDPG	10.66%	6.82%	0.226	0.411	0.213	-18.76%
	200	DDPG	-18.12%	-12.34%	0.460	-0.274	0.088	-42.06%
MT1B	10	DDPG	8.57%	5.43%	0.197	0.630	0.327	-16.44%
	200	DDPG	-17.03%	-11.74%	0.472	-0.142	0.103	-42.77%
MT1C	**10**	**DDPG**	**12.34%**	**7.86%**	0.181	**0.773**	**0.400**	-13.90%
	200	DDPG	-18.70%	-12.98%	0.463	-0.224	0.109	-42.43%
MT2	10	PPO	7.51%	4.82%	**0.156**	0.240	0.229	**-12.91%**
	200	DDPG	-17.97%	-12.19%	0.483	-0.287	0.062	-43.92%
MT3	10	PPO	10.40%	6.66%	0.212	0.542	0.264	-17.46%
	200	DDPG	-9.18%	-6.57%	0.323	0.058	0.238	-29.24%
MT3ta	10	PPO	11.08%	7.10%	0.233	0.346	0.236	-19.35%
	200	DDPG	-19.34%	-13.14%	0.489	-0.238	0.053	-44.74%
MT4	10	PPO	10.02%	6.42%	0.213	0.400	0.253	-17.69%
	200	PPO	-18.36%	-12.50%	0.441	-0.358	0.105	-40.67%
MT4ta	10	DDPG	10.77%	6.89%	0.245	0.424	0.216	-20.57%
	200	DDPG	-16.00%	-10.85%	0.458	-0.108	0.056	-41.49%
BH			-20.85%	-13.95%	0.520	-0.187	0.021	-46.92%

Legend: In bold: best model considering all metrics. In green: models that performed better than the BH model for the specific metric. In red: worst model for the specific metric. MOS: maximum order size; DRL: DRL agent; AR: annual returns; CR: cumulative returns; AV: annual volatility; SR: Sharpe ratio; ST: stability; MD: maximum drawdown.

Source: elaborated by the author.

Although the models with a maximum order size of 200 presented the best results in the first trading scenario (especially the MT1C), in Trade 2, they presented worse results than their counterparts with a maximum order size of 10. This further supports the assumption that, in high volatility and downward trend scenarios, the maximum order size may play an essential part in the trading strategy.

The best model in this scenario was the MT1C with a maximum order size of 10. It presented the best results in the returns-related metrics (12.34% on annual and 7.86% on cumulative returns versus -20.85% on annual and -13.95% on cumulative returns for the BH strategy). This model also obtained the best results for the risk-related metrics (0.773 versus -0.187 for the Sharpe ratio for the BH strategy) and a lower maximum drawdown than the BH strategy (-12.91% for MT1C versus -46.92% for BH). Therefore, this is the most recommended model for this trading scenario.

As with Trade 1, the MT2 with a maximum order size of 10 presented the lowest annual volatility (0.156 versus 0.520 for BH) and maximum drawdown (-12.91% versus -46.92% for BH). Therefore, this model presented promising results for this scenario, considering a risk-averse investor. Nevertheless, unlike in Trade 1, the stability and the maximum drawdown were not considerably lower than the MT1C.

As for the use of TIs, the results from the MT3, MT3ta, MT4, and MT4ta show that there was a slight improvement in terms of return-related metrics. However, except for the Sharpe ratio for MT4ta in relation to MT4, all the other metrics were worse on the models using TI. Therefore, the results point out that TI's use did not improve the results of the models.

As was observed for Trade 1, the use of the predictions of the M1 model as inputs for the DRL agent improved the results considerably for all models. The best model in both scenarios was the MT1C (that uses the prediction of the LSTM as an additional feature). This strengthens the assumption that using the LSTM on the M1 module improves the model's trading results.

In relation to using the M2 module, the results were slightly different from Trade 1. In isolation (MT2), the M2 module provided inputs that made the model more risk-averse, improving the losses-related and volatility-related metrics.

Nevertheless, this effect was not observed when used together with the M1 module (MT4 and MT4ta). The MT4 and MT4ta had very similar results to the MT3 and MT3ta models. Therefore, it can be inferred that using both M1 and M2 modules together, specifically for scenarios with high volatility, does not make the model as responsive as

Table 23 – Comparison and statistical analysis of the best models with the BH strategy for Trade 2 (2020), considering financial metrics on 30 executions

Model	MOS	DRL	AR	CR	AV	SR	ST	MD
MT1C	10	DDPG	**12.34%**	**7.86%**	**0.181**	**0.773**	**0.400**	**-13.90%**
MT2	10	PPO	**7.51%**	**4.82%**	**0.156**	0.240	0.229	**-12.91%**
BH			-20.85%	-13.95%	0.520	-0.187	0.021	-46.92%

Legend: In bold: values with statistical difference (p <0.01% using Student's T-test) from the BH model. In green: models that performed better than the BH model for the specific metric. In red: worst model for the specific metric. MOS: maximum order size; DRL: DRL agent; AR: annual returns; CR: cumulative returns; AV: annual volatility; SR: Sharpe ratio; ST: stability; MD: maximum drawdown.

Source: elaborated by the author.

the use of only the M2 module. This observation should be better explored in the future, as this may help considerably in choosing models and features for a more risk-averse strategy.

It can be concluded from this analysis that: (i) all models provided better results for all metrics in relation to the BH strategy, with an exception of MT1A, MT1C, MT2, and MT4ta with a maximum order size of 200 for the Sharpe ratio; (ii) all models with maximum order size of 10 provided significantly better results for all metrics; (iii) all models with maximum order size of 200 presented negative returns-related metrics and maximum drawdown lower than 20%; (iv) the MT1C with maximum order size of 10 was the best trading model evaluated; (v) the best results in terms of volatility and maximum drawdown were obtained by the MT2 with a maximum order size of 10; (vi) the use of TI did not improve the models' results in this scenario; (vii) the use of the M1 and M2 modules in isolation (MT1A, MT1B, MT1C, and MT2) presented better results for all metrics in relation to the BH strategy; and (vii) the use of both M1 and M2 modules together (MT4 and MT4ta) did not significantly improve the models' results in comparison to the MT3 and MT3ta models.

Table 23 contains a comparison between the MT1C with a maximum order size of 10 (best model for Trade 2) and the MT2 with a maximum order size of 10 (the model with the lowest risk that was evaluated on both Trades 1 and 2) models for the evaluated metrics, as well as the baseline used (the BH strategy). Similar to the analysis for Trade 1, the Student's t-test statistics were used to evaluate which value differences were statistically significant in relation to the BH strategy.

It is interesting to observe in Table 23 that, except for the Sharpe ratio and stability for the MT2 model, all the other metrics were statistically better in relation to the BH strategy. The annual returns were positive (while on the BH strategy, they were -20.85%), the Sharpe ratios were positive, and the maximum drawdown was considerably higher (-13.90% for MT1C and -12.91% for MT2 versus -46.92% for BH). Taking all those factors into account, it is possible to conclude that the use of those models was better than the BH strategy and that, unless the investor is exceptionally risk-averse, she should use the MT1C model.

Finally, four critical aspects must be considered: (i) after the hyperparameters have been chosen for each module, retraining the whole system (M1, M2, and MT modules) takes less than one day with a computer with the technical specifications used in this work (described in Chapter 3); (ii) using the proposed system for prediction purposes (without retraining it), considering all modules, takes less than 10 minutes for each data point (allowing for intraday predictions if necessary); (iii) the main factor that may increase the training time considerably is evaluating multiple assets; and (iv) the results of the M2 module may be improved by using a more extensive training dataset. The results obtained in the models' experiments and analyses on both trades are used to answer the main and secondary research questions in the next section.

4.7 Answers for the research questions

This subsection contains the answers to the main research question (4.7.1) and the secondary questions (4.7.2).

4.7.1 Main research question

The experiments' results showed better trading results for several models for the various metrics in relation to the BH strategy, especially for the second trading scenario (trading in 2020). For the first trading scenario (2019-2020), the price prediction module (M1) was a central component of the MT1C model with a maximum order size of 200, which showed the best results for the following metrics: annual and cumulative returns, Sharpe ratio, and stability. The sentiment prediction module (M2) was a central component of the MT2 model with a maximum order size of 10, which showed better results for the annual volatility and maximum drawdown metrics. The statistical analysis showed that the MT1C obtained significantly better results for the returns-related metrics in rela-

tion to the BH strategy. It also showed that the MT2 model obtained significantly better results for the annual volatility and maximum drawdown metrics.

For the second trading scenario (2020), the MT1C with a maximum order size of 10 was the best model evaluated. As in Trade 1, the MT2 with a maximum order size of 10 showed the best results for the annual volatility and maximum drawdown metrics. The statistical analysis showed that, except for the Sharpe ratio and the stability for MT2, all the metrics for MT1C and MT2 were significantly better than for the BH strategy. Even more critical, their results were not negative, and the maximum drawdown was much more positive.

The experiment results showed that, at least for the BOVA11 ETF, the use of the sentiment feature extracted from news headlines did not improve the model significantly in terms of returns or TIs as inputs for the DRL agent but improved its responsiveness in the presence of volatility. The MT2 models were more risk-averse for both scenarios, reducing the risk of losses.

Notwithstanding, the returns for Trade 1 were very low for this model. Although more studies are needed, this infers that, for certain types of markets and scenarios, sentiment features may not be as impactful as improving the predictions' quality for improving models' results. Also, evaluating both trading periods, it is possible to observe that one of the most important factors, besides the careful selection of input features, the model choice, and the hyperparameters tuning, is the maximum trading size for each period. In more volatile periods (such as during the Covid-19 pandemics in 2020), the use of smaller lot sizes improved considerably on the results compared to both the baseline and larger lot sizes. In this case, the proposed system provided significantly better results for all metrics in relation to the BH strategy. The proposed system also showed satisfactory results when operating for more extended periods (such as in the first trading scenario).

Therefore, it is possible to conclude that the proposed system presents better trading results than the BH strategy for both scenarios' evaluated metrics. This research also addresses and complements the work by Liu et al. (2020) and addresses several points on the work of Fischer (2018) by incorporating:

1. A price prediction module that produces a price prediction input for the DRL agent: M1;

2. A sentiment prediction module, which produces a sentiment prediction input for the DRL agent: M2;

3. A thorough evaluation of the various state-of-the-art models for price prediction, sentiment prediction, and DRL trading.

4.7.2 Secondary research questions

The following paragraphs contain the answers for each of the secondary research questions.

SRQ1: Does the use of DRL with sentiment analysis improve stock trading in terms of profits in relation to the use of the BH strategy?

The use of sentiment analysis did not improve the trading model results in terms of returns-related metrics. It improved the models' volatility-related and losses-related metrics but provided very low returns, at least in Trade 1. Using a price prediction module (such as M1) would provide results that are more in line with investors' expectations. Therefore, sentiment analysis is recommended for more risk-averse strategies, especially in very volatile markets with a downward trend. In other situations, using price predictions as features for the DRL agent is more recommended due to its higher returns without excessive risks. An option to be explored is to incorporate sentiment from other sources, such as social media, company reports, analysts' reports, among others. Nevertheless, this may not be effective for daily trading, as impactful news can be quickly incorporated into an asset's price. More studies are needed comparing the impact of sentiment features on trading for different frequencies (especially minute and hourly trading) and assets.

SRQ2: Which model results in the best forecast of market indices prices? Econometrics, ML, or DL models?

For the price prediction module (M1), the models that had the best results were: the ensembles (SARIMAX and SVR, SARIMAX and LSTM, and SVR and LSTM), the SARIMAX, and the SVR multivariate models. Using the blocking time series splits provided better results than using the traditional time series splits, and the best single model (SARIMAX) used TIs as features, besides prices and volume. The best models were the multivariate versions, which better identified the patterns used for price prediction, resulting in lower MAE and MSE. Therefore, it is possible to conclude that using ensembles is very important for price prediction, especially by mixing econometrics and machine learning models. It is also possible to infer that the LSTM may have presented worse results than the SARIMAX and SVR due to the training dataset's small size.

SRQ3: Does the use of TIs as features improve the forecasts of market

indices prices?

For the price prediction module (M1), TIs' use as features improved the SARIMAX and the LSTM multivariate models. Nevertheless, it did not improve the AdaBoost, LSTM univariate, SVR univariate, and SVR multivariate models. Therefore, it is possible to conclude that, for some models, the use of TIs may improve the forecasts' results. As the SARIMAX was the model with the best results for price prediction, and as the LSTM multivariate was part of the trading model that presented the best results for both trades analyzed in this research, it is possible to conclude that the TIs should be considered on price prediction modules.

SRQ4: Which model (considering MLP and CNN) best predicts market sentiment? and **SRQ5: Does the use of a dictionary (considering Sentilex, Oplexico, and WordnetAffectBR) improve the prediction of market sentiment for financial news headlines in Portuguese?**

For the sentiment prediction module (M2), the model that best predicted market sentiment was the CNN. For the dictionaries, it was observed that their use improved the MLP and CNN models' results. The WordNetAffectBR presented the best results for both models. Therefore, it is possible to conclude that dictionaries are essential to improve the results of the sentiment prediction module.

SRQ6: Does the use of the price prediction module improve stock trading results?

Based on the experiments conducted, it is possible to conclude that the use of the price prediction module (M1) improves the trading results of the DRL agent, especially when using the predictions of the multivariate LSTM (model MT1C). Therefore, it is crucial to use a price prediction module for the daily trading of the BOVA11 index. Although more analyses are needed with various assets, it can be inferred, based on the experiments, that this can also be applied to the most significant assets in the stock market (as they directly influence the prices of the BOVA11 index).

SRQ7: Does the use of sentiment analysis of news headlines improve stock trading results compared to the BH strategy?

Based on the experiments conducted, it is possible to conclude that the sentiment prediction module (M2) reduced the risk of the DRL agent's strategy but did not improve on the returns, especially on Trade 1. The models that used the sentiment analysis module (MT2, MT4, and MT4ta) presented worse results than the MT1C and the BH strategy for

both trades, especially for returns-related and risk-related metrics. Therefore, considering all metrics, the sentiment analysis module's use only improved the stock trading results considerably compared to the BH strategy on Trade 1. On Trade 2, all models presented better results than the BH strategy, and the main advantage of using M2 in isolation (MT2) was to improve its risk-averseness. It is essential to further research the use of sentiments for trading in the Brazilian market by considering various news sources, dictionaries, and trading frequencies (especially for high-frequency trading).

SRQ8: Does the use of TIs improve the stock trading results of DRL models?

According to the experiments conducted, using TIs as inputs for the DRL agent did not improve trading results consistently. However, it was also observed that using TIs on the price prediction module (M1) resulted in better price predictions for some models. The trading models that used features derived from the M1 module presented better results for most of the evaluated metrics: annual and cumulative returns, Sharpe ratio, and stability. Therefore, it is possible to conclude that using TIs as features of the DRL agent may not lead to good results but that these TIs must be evaluated as features of the price prediction module. One of the reasons that may explain those results is that some of the models used on the price prediction module, especially the SARIMAX and LSTM, were designed to extract patterns from data with autocorrelation. As a result, they may be better suited to find those patterns on small datasets (as was the case in this research) with the addition of handcrafted features.

4.8 Chapter summary

In this Chapter, the proposed system was explored in-depth, and experiments were conducted to evaluate all its components separately and used together in two different trading scenarios. The research questions were also answered based on the analyses of the obtained results.

The proposed system is composed of three modules: (i) M1, which is used for predicting stock prices based on time series analysis; (ii) M2, which is used for predicting market sentiment based on news titles; and (iii) MT, which is used for stock trading. Two trading scenarios were evaluated: (i) Trade 1 (2019-2020), which contained both upward and downward trends; and (ii) Trade 2 (2020), which mainly contained a steep downward trend, as a result of the effects of the Covid-19 pandemic on the economy. Eight trading

models and configurations of the proposed system were evaluated.

For both trades, the use of the M1 module resulted in models that were better than the BH strategy in all metrics. The use of the M2 module resulted in more risk-averse models, and that presented very low returns on Trade 1. The use of both modules together did not improve the model's overall results that used only the M1 module as inputs.

Therefore, it is possible to conclude that the proposed system should be used: (i) with only the M1 module (MT1C with a maximum order size of 200) for long term or daily trading during less volatile scenarios; (ii) with only the M1 module (MT1C with a maximum order size of 200) if the investor is not risk-averse; and (iii) with only the M2 module (MT2 with a maximum order size of 10) if the investor is risk-averse and is on a very volatile scenario, especially on a steep downward trend.

5 DISCUSSIONS

This section contains several essential discussions regarding the proposed system, its possible use, and the main impacts of this work on different trading tasks. It is divided into the following sections: 5.1 describes the main contributions of this work; 5.2 describes the main limitations of this research; 5.3 contains several suggestions for application of the proposed trading system; 5.4 contains the most relevant recommendations for future work; and 5.5 concludes this chapter, with a summary of its main points.

5.1 Main contributions of this work

This section describes the main contributions of this work, analyzing the results of the modules of the proposed system and the system as a whole. Section 5.1.1 explores the main contribution of this work: the use of time series and sentiment predictions from state-of-the-art DL models as additional features for the DRL agent. This applies not only to the financial domain but to other sub-domains as well, such as agricultural products, iron, steel, oil, and other commodities. The following subsections explore the work's minor contributions, which apply mainly for stock trading: 5.1.2 explores the use of multiple features to predict Brazilian stock indices prices; 5.1.3 explores the use of market sentiment extracted from news titles in Portuguese; 5.1.4 explores the use of DRL for stock trading on the Brazilian stock market; and 5.1.5 explores the use of financial domain metrics (versus classical ML metrics) to evaluate stock trading models.

5.1.1 Using time series and sentiment predictions as additional features for the DRL agent

The main contribution of this work was the proposal of a system for using additional features to DRL agents that may help capture additional information on the environment. This is especially useful for complex scenarios with unknown dynamics and low signal-to-noise. The additional features proposed and analyzed on the experiments on automated

stock market trading were: (i) time series prediction features; and (ii) sentiment prediction features. It is essential to observe that the proposed system can be used on several different sub-domains that present those characteristics (such as agricultural products, iron, steel, oil, and other commodities) and that the models used in each component can also be changed and evaluated for those sub-domains.

The Brazilian stock market was chosen as a case study of the proposed system due to five main factors: (i) it presents the main aspects of complex scenarios; (ii) its dynamics are (and tend to remain) unknown and constantly changing; (iii) it is a developing stock market, what means that gains may be obtained by better extracting relevant information from the data available; (iv) it has a strong baseline that is difficult to beat, the BH strategy; and (v) there is much interest on improving the results of stock market trading, as this can be translated to capital gains.

Several models were evaluated to generate the additional features, including econometrics, traditional ML, and DL models. State-of-the-art models were evaluated for each task, and different input configurations were considered. The results obtained in the evaluated scenarios support the claim that the proposed system improves the results in relation to the baseline model and the current implementation of the DRL agents evaluated. This indicates that the proposed use of the additional features improved the quality of the information extracted by the DRL agents from the environment.

Additionally, this work addresses several important concerns present on the DRL literature (FISCHER, 2018; FRANÇOIS-LAVET et al., 2018; VÁZQUEZ-CANTELI; NAGY, 2019; LI; RAO; SHI, 2018; WU et al., 2019; MENG; KHUSHI, 2019): (i) the need to further explore state-of-the-art models for algorithmic trading, considering different scenarios; (ii) the lack of studies of actor-critic models for automated stock trading; (iii) exploring the impact and the importance of different hyperparameters for the DRL agents; and (iv) exploring different DRL agents in a standard task. Lastly, the transaction costs are also considered, which is a very relevant aspect that is not considered in most models in the literature.

5.1.2 Considering multiple features to predict Brazilian stock indices prices

This work's second contribution is the M1 module by considering and evaluating multiple features and models to predict the stock indices prices. It is also one of the few works to explore in-depth different models and hyperparameters for predicting Brazilian

stock indices prices. The use of TI for price prediction was also analyzed. Finally, the impacts of using two important cross-validation methods were evaluated for all models and ensembles with different compositions. Those aspects complement the works of Kara, Boyacioglu and Baykan (2011), Persio and Honchar (2016), Fischer and Krauss (2018), Mehtab, Sen and Dasgupta (2020), Eapen, Verma and Bein (2019), Yadav, Jha and Sharan (2020), and Pauli, Kleina and Bonat (2020), which are very relevant for stock market price prediction.

In relation to the use of most traditional models, the DL models have four main advantages: (i) the possibility of identifying complex non-linear patterns in the data (JORDAN; MITCHELL, 2015; NELSON; PEREIRA; OLIVEIRA, 2017; LECUN; BENGIO; HINTON, 2015; SEZER; GUDELEK; OZBAYOGLU, 2020); (ii) being entirely data-driven, in the sense that there is no need to implement rules on the model's behavior (it is based solely on the model's architecture, the hyperparameters chosen, and the dataset used); (iii) the possibility of transferring learning between assets and sub-domains; and (iv) automatic feature generation, detecting trends that would demand manual labor using other techniques. The DL models can also use various data types as inputs, such as text (to extract market sentiment), as done in the M2 module.

Most ML and DL models in the literature tend to consider only OHLCV data, as in many of the works explored by the extensive literature reviews conducted by Ryll and Seidens (2019) and Sezer, Gudelek and Ozbayoglu (2020). Nevertheless, TI's use is important in improving the results of several models, and these must be evaluated together with the different model hyperparameters values and cross-validation methods. Lastly, the methodology used in this work and the model implementations can also be applied to other time series analysis problems, not only on the stock market.

The following subsection explores this work's second contribution: considering as an input for the trading model market sentiment extracted from financial news titles in Portuguese.

5.1.3 Considering market sentiment extracted from financial news titles in Portuguese

This work's third contribution is related to: (i) considering the market sentiment extracted from final news titles as input for trading in the Brazilian market; and (ii) evaluating different dictionaries, hyperparameters, and state-of-the-art models for sentiment analysis in Portuguese on the financial domain. This is one of the few works that address

news in Portuguese for algorithmic trading purposes. Most of the literature focuses on the English language. Also, it is essential to note that there are no open financial news datasets in Portuguese with timestamps available for trading purposes.

Considering the sentiment input extracted from the news titles is important to increase the models' responsiveness, making it more suitable for risk-averse strategies (NASSIRTOUSSI et al., 2014; BOLLEN; MAO; ZENG, 2011). It was also observed that, for daily trading, the impact is not that significant as the use of a price prediction signal as an input for the trading agent. This may be because financial news may be incorporated on the price faster than one day. Some research, such as the work by Souma, Vodenska and Aoyama (2019), observed good results on sentiment analysis for intraday trading. One of the reasons that may explain those results was the fact that it allowed for better prediction in a short timeframe. However, it is possible to adapt the proposed system for intraday trading.

As the word embedding used (GloVe in Portuguese with 300 dimensions) provided a vast amount of information for the sentiment analysis, it was observed that the dictionaries themselves only improved the models by around 2%. Nevertheless, even though this is a small number, it is significant in the trading domain, as any increase can be turned into profit by the investor. Lastly, the methodology implemented in this work can be used to evaluate different models, additional hyperparameters, and hyperparameter values and inputs (such as social media messages, texts from relevant press releases, and corporate documents, among others). The following subsection explores the third contribution of this work: using DRL for automatic stock trading of Brazilian assets.

5.1.4 Using DRL for automatic stock trading of Brazilian stock indices

As shown by the experiments' results, the use of DRL for trading can improve the results compared to trading strategies based only on TI or on price prediction models for the Brazilian market. One important consideration is that the reward function used best reflects the decision-makers' objectives in relation to the use of price prediction models. Most of the works in the literature, such as the ones by Pauli, Kleina and Bonat (2020), Nelson, Pereira and Oliveira (2017), Oliveira, Nobre and Zárate (2013) and Eapen, Verma and Bein (2019), use only price prediction models, which are not directly related to the quality of each decision. In other words, even if those models provide good predictions (predictions with small errors), these can still be used for bad strategies or lead to losses

(SEZER; GUDELEK; OZBAYOGLU, 2020).

The DRL agents can also complement the use of other models, as was done in this research, by: (i) better identifying the patterns and relations between OHLCV data, market sentiment, and price predictions, which may be challenging to identify using other types of models; (ii) allowing for training in small datasets, which is very difficult for highly complex models, such as the LSTM; (iii) allowing for transfer learning between assets and stock markets, what is very difficult to implement with other types of models; and (iv) allowing for the identification of trends and non-linear patterns, which may be challenging to identify using econometrics models. The use of DRL can also provide better results than the BH strategy on high volatile scenarios, as was explored in the second trading scenario. These results are in line with the literature of other stock markets, as was explored in Chapter 2.

According to Fischer (2018), Yang et al. (2020), Liu et al. (2020), and Meng and Khushi (2019), RL models (specially DRL models) can detect highly complex patterns and provide good trading results. However, most of the research was: (i) conducted on mature markets and relatively stable scenarios; and (ii) did not consider additional important features as inputs such as sentiment prediction and price prediction. According to Fischer (2018) and Meng and Khushi (2019), most of the works rely on RL models, which are not as well-suited for ambiguous and dynamic scenarios (reflected mainly on the model's state spaces) as the DRL models.

The fourth contribution of this work is that it complements the works of Dantas and Silva (2018), Conegundes and Pereira (2020), Li, Rao and Shi (2018), Wu et al. (2019), and some of the main future work recommendations by Fischer (2018) and Meng and Khushi (2019) on using state-of-the-art DRL with transaction costs and multiple features for stock trading. Different in-depth hyperparameters were explored, incorporating sentiment and price prediction modules, and exploring different trading scenarios in a development market.

Also, as the FinRL library was used as the basis of this module, it is possible to develop different trading scenarios, rules for the agents, modifications on the MDP, and implementation of additional DRL agents (using the OpenAI Gym framework). It is also possible to implement functionalities to use the model on real-life automatic trading, incorporating aspects of stop-loss, stop gain, changes in transaction costs, multiple assets trading, among various other aspects. The following subsection explores this work's fourth contribution: using financial domain models to evaluate the stock trading models.

5.1.5 Using financial domain metrics to evaluate models for stock trading

The fifth and last contribution of this work was to use on the proposed system financial domain metrics to evaluate the models instead of traditional ML metrics (such as precision or recall for classification and MSE or MAE regression tasks). Several works in the literature, such as the ones explored in Section 2.2, Chapter 2, use traditional ML metrics to evaluate the trading models. Most of the ones that use financial metrics focus only on returns metrics, such as in the works by Dantas and Silva (2018) and Wu et al. (2019). Some works, such as the ones by Li, Rao and Shi (2018) and Lei et al. (2020) considered multiple metrics. However, they did not consider metrics that evaluated all the aspects that are relevant for the investors (returns, volatility, risks, and losses).

The use of the six financial metrics in this work allowed for a better comprehension of the impact of the differences in the trading models (and their inputs) in terms of returns, volatility, risks, and losses. Therefore, it was possible to address the concerns related to metrics' use on trading with DRL cited by Fischer (2018) and Meng and Khushi (2019). The DRL agent's reward was also a relevant metric for the financial domain: the portfolio's difference between timesteps. Therefore, the model's main aim was to generate and follow a policy that would lead to the highest gain possible.

Lastly, it is essential to note that the proposed system can be adapted for using: (i) different metrics, which may be relevant on specific situations and sectors; and (ii) a different reward function for the DRL agent, which could lead to policies that were significantly different from the ones observed in this work. Exploring different configurations of the MDP, especially on the reward signal, the state spaces, and the action spaces, is one of the main concerns of the RL and DRL researches. The following section contains the main limitations of this research.

5.2 Limitations of the research

The main limitations of this research can be divided into three main categories: (i) difficulties related to the stock market itself; (ii) difficulties related to the components of the proposed system; and (iii) difficulties related to the lack of open datasets and code.

There were three main limitations on this study related to the characteristics of the stock market itself. The first is related to the difficulty in identifying trends and cycles on the stock market, which is also observed in several stock markets and throughout

the trading literature (NASSIRTOUSSI et al., 2014; YADAV; JHA; SHARAN, 2020; JIN; YANG; LIU, 2020; HU et al., 2018; RYLL; SEIDENS, 2019; SEZER; GUDELEK; OZBAYOGLU, 2020).

The second significant limitation was related to the complexities of the stock market itself. Currently, it is too difficult to obtain data from the agents themselves, their strategies, and the actions taken by them in real-time (and how the actions of big hedge funds, for example, may influence asset prices). To address this limitation, the news titles' sentiment was used as a proxy of market sentiment, which is generally used in the literature to infer the market's movements as a whole, as in the many of the works evaluated by Nassirtoussi et al. (2014), and on the works by Mansar et al. (2017), Dridi, Atzeni and Recupero (2019), and Ferreira et al. (2019). Nevertheless, in future works, it would be very interesting to explore the simulation of those very influential agents (for example, by using several DRL agents on the MT module, each simulating a hedge fund).

The last limitation related to the stock market itself was that the implemented models do not consider behavioral finance findings, which state that the stock market agents do not always behave rationally. Therefore, their actions may be motivated by other factors that a rational agent should not consider in decision-making (BONDT; THALER, 1985; NASSIRTOUSSI et al., 2014; JIN; YANG; LIU, 2020). Some examples are: overselling in some periods of the week (before the weekend) or year (to rebalance the portfolio) and different trading behavior according to the weather (influencing risk-averseness on decision-making) (BONDT; THALER, 1985; NASSIRTOUSSI et al., 2014). Although those factors are not considered in the algorithmic trading literature, it would be interesting to explore them in future works, especially on the MT module.

The second category is related to the components of the proposed system. Two main limitations were observed and addressed. These impact most of the works in the area, demanding more exploration. The first is related to the lack of sentiment dictionaries (especially related to financial terms) in Portuguese. Several options of lexicons are available in English for the financial domain, such as the work by Loughran and Mcdonald (2011).

However, in Portuguese, only general sentiment dictionaries are available. As observed by Loughran and Mcdonald (2011) and Nassirtoussi et al. (2014), this is a problem because financial terms may have different sentiments in a general dictionary. The best available option that was found to address this problem during the present research was the use of word embeddings. These can complement the sentiment dictionaries, as they consider the relations between the different words, as observed by Ferreira et al. (2019), Zhang,

Wang and Liu (2018), and Mansar et al. (2017).

The second limitation is related to the difficulty of implementing RL and DRL models for stock trading. This is a work in progress by several research groups in practitioners, with the FinRL[1] being one of the most accessible libraries to use and adapt to different situations. It also has an active community and extensive documentation, helping both researchers and practitioners implement DRL for stock trading. Some of the other libraries that were evaluated on the experiments before choosing the FinRL library as the MT module basis were: Tensortrade[2], FinGym[3], Stock Trading with RRL[4], RLTrader[5], TradingGym[6], and KerasRL[7]. Nevertheless, all those libraries had problems with documentation or a lack of essential functionalities for implementation. Therefore, the FinRL was the best available option.

The last category of limitations is related to the lack of datasets and open code. There were two main limitations to this category. The first was that, unlike works in other domains, only a few works share their code and the datasets used in the stock trading domain. This increases the complexity of evaluating different models and comparing works in the literature, as subtle differences in hyperparameters configuration (as explored in-depth in Chapter 4) can significantly change the models' results.

The second limitation is related to the lack of open datasets of news titles in Portuguese. This is because most news providers do not allow sharing individual news or news titles. Therefore, the few works in the literature that use sentiment analysis in Portuguese on the financial domain cannot share the raw data used to generate the sentiment signals. For this reason, a web scraping tool was used to gather the data from these websites. It is also crucial for this domain that the datasets contain timestamps, which is not always necessary for other sentiment analysis domains, such as product evaluation. The following section contains several suggestions for the application of the proposed system.

5.3 Suggestions for application

This section explores several suggestions for applications of the proposed system and is divided into the following subsections: 5.3.1 explores its use on daily trading; 5.3.2

[1]https://github.com/AI4Finance-LLC/FinRL-Library
[2]https://www.tensortrade.org/en/latest/
[3]https://github.com/entrpn/fingym
[4]https://github.com/rajatgarg149/Stock-Trading-using-RRL
[5]https://github.com/notadamking/RLTrader
[6]https://github.com/Yvictor/TradingGym
[7]https://github.com/keras-rl/keras-rl

explores its use for high-frequency trading (hourly, by the minute or by second); and 5.3.3 explores the use of transfer learning between assets and markets.

5.3.1 Using the proposed system for daily trading

Even though the proposed system was designed considering the options of trading different assets on different time windows, it is crucial to notice that all evaluations and experiments were conducted on a single asset's daily trading. This was done because it is the most commonly explored case in the literature, and daily trading is the most common form of trading for most investors in real-life scenarios. Additionally, it is considerably easier to gather data (both OHCLV data and news titles data) on a daily frequency. Intraday prices are available only using scraping tools or paid services, which may not be available for all investors.

Although the results on one scenario cannot be directly replicated on different scenarios due to the stock market's complexity, the proposed system can be used for daily trading the BOVA11 market ETF with the final model configuration (described in Chapter 4, sections 4.4 and 4.5). Nevertheless, it is critical to evaluate if: (i) the current scenario is closer to the first scenario (Trade 1, in which there are upward and downward trends) or the second scenario (Trade 2, in which there is a steep downward trend); and (ii) the risk-averseness of the agent.

It is critical to note that the best use of the proposed system in real-life scenarios in its current form is not to use it to trade directly but to provide additional information for the decision-makers. Few of the trading models are used without evaluating their outputs due to the high complexity of the stock markets (and the variety of known and unknown influences on market prices). For example, the decision-maker could simulate different scenarios (generating different input values) and evaluate the proposed system's actions. Then, she could consider important information from other sources, such as: (i) fundamental analysis indicators; (ii) movements from players that have considerable influence on the stock market, such as big hedge funds; and (iii) external factors that may influence the market, such as the interest rate in the USA or the commodity prices in China. Lastly, she could take action on the market based on all the information gathered.

It is also important to observe that monthly or bi-monthly price intervals could reduce the volatility observed on the daily prices. In this context, the proposed system (especially the component M1) may present lower prediction errors. Also, fundamental data (such as data extracted from companies' reports and macroeconomic data) could be used as

additional inputs. Therefore, it would be interesting to evaluate the model on medium and long-term applications.

Finally, it is crucial to observe that the proposed system could be implemented in two main configurations in the context of multiple assets: (i) with individual models and agents trained for each asset; and (ii) with agents that consider multiple assets. The first configuration can be readily implemented without additional changes in the implemented code. Nevertheless, this would consider that the assets are independent, which may not always be the case. The second configuration would consider the relations between the assets and demand the implementation of portfolio optimization concepts on the agents' MDP.

5.3.2 Using the proposed system for high-frequency trading

Intraday and high-frequency trading are considered, in general, as significantly riskier than daily or long-term trading. The high-frequency trading scenario is considerably different from the daily trading scenario for two main reasons: (i) decisions must be made faster, which leads to more automation and less time evaluating model outputs before deciding to buy or sell; and (ii) when trading on minutes or seconds, the prices behave much more like a random walk. In this case, speed in taking action is vital, and one of the primary sources of information that may improve decision-making is market sentiment.

For using the proposed system for high-frequency trading, it is essential to: (i) re-evaluate all the models and hyperparameters of all models, as their behavior could be very different in this scenario; (ii) implement functionalities for the trading module for shorting assets, using leverage, and using stop loss and stop gain options, which are very common in intraday trading; and (iii) further explore additional sources for the M2 module, such as: relevant social media messages, movements of big players in the market, and real-time market sentiment prediction. In the M1 module, it is important to gather data in real-time and evaluate the book order prices and volumes traded by the different agents, as these may improve the price prediction.

Unlike the daily trading scenario, the models for high-frequency trading must all run in real-time, increasing the need for computational resources. The cost of operating such a system is considerably higher, not being recommended for individual investors. Lastly, it is vital to observe that, as there will be less time for evaluating the inputs of the trading module (especially for trading on minutes and seconds), it will need to be connected to the brokerage service and trade automatically.

5.3.3 Transferring the learning for other assets and markets

One important aspect that was out of this work scope but that could be explored is the use of transfer learning for trading with a DRL agent. The transfer learning could be used on the MT module by training the system on a different asset or market and then transferring the MT module to a new asset or market. This may improve the trading system returns because there may be assets with more extended time series (for example, the SP500) and markets similar in terms of its development (such as the stock markets in China).

Therefore, it would be important to consider transfer learning for trading in three primary contexts: (i) inside the same sub-domain (for example, between two assets of the same stock market); (ii) between sub-domains with similar characteristics (for example, between two similar assets from different stock markets); and (iii) between different sub-domains (for example, between assets without similar characteristics from different markets). Transfer learning between market segments (for example, from commodities to utilities) could also be explored. These explorations could also improve the financial domain's knowledge, helping to understand better market dynamics and the influence between different assets and stock markets.

Transfer learning is also a significant field of study on RL and DRL, considering different domains. In general, transfer learning improves decision-making in situations when there are small datasets or when the decisions are very complex. In those cases, if high-quality datasets from other domains exist, the agent can pre-train on those datasets and then fine-tune on the desired dataset, improving its pattern recognition and decision-making. Transfer learning is widely used in robotics, from transferring knowledge obtained on computer simulations to real-life robots.

Lastly, the methodology used and the system proposed in this work can be adapted to several other assets and products: soybean futures, iron, steel, oil, and other agricultural products. The main requirements for its use on different sub-domains are: (i) availability of a high-quality dataset for the sentiment analysis, containing news, news titles, or messages from social media; and (ii) availability of a high-quality dataset for the time series analysis and price prediction. Ideally, both datasets should encompass periods of high and low trends.

The following section discusses the main recommendations for future work.

5.4 Recommendations for future work

Throughout this work, several important aspects that could be considered in future works were cited. This section contains the most relevant ones for improving the results of stock market trading, divided into seven categories: (i) related to model inputs; (ii) related to the traded asset; (iii) related to the specific stock market; (iv) related to price prediction (M1 module); (v) related to market sentiment prediction (M2 module); (vi) related to stock market trading using DRL; and (vii) related to new evaluation scenarios.

The first main aspect that could be explored in future works regarding model inputs is the implementation of concepts from behavioral finance, as these are rarely considered in algorithmic trading. These could allow identifying new patterns on the data, which could be used as inputs of the trading module. Another very important aspect is considering the use of unsupervised models (such as k-means, self-organizing maps, t-sne, among others) to new feature identification and asset selection. These could lead to better portfolios, considering multiple assets. Additionally, the impacts of using the volatility as a model input could be explored, as an increase in the asset price's volatility on a specific period could be connected to more frequent trades in that period.

The second group is related to the traded asset itself. One of the most critical points to explore in future works is related to adapting the proposed trading system for a portfolio with multiple assets. This would allow for diversification of the portfolio and the ability to capture the movements of different assets. This could lead to better gains and a lower impact on highly volatile scenarios, mainly if the portfolio were composed of a diversified group of assets. Several important works are related to portfolio optimization, and these could be incorporated into the proposed systems.

Three other very important aspects that could be explored related to the market assets are: (i) the implementation of functionalities for trading derivatives, which present additional complexity and volatility in comparison to the traditional assets; (ii) the implementation of functionalities for trading cryptocurrencies; and (iii) the implementation of functionalities for considering different asset classes, such as real estate investment trusts (REIT) and bonds. In the case of items one and two, intraday trading functionalities are essential for improving the trading system's results. In the case of the third item, it is essential to incorporate changes in the MDP so that the rewards also consider the long-term reward of investing in bonds (or the model may only invest in stocks due to their potential immediate gains).

Lastly, it is vital to observe that the BOVA11 ETF is a proxy of the Ibovespa index, which represents the overal stock market in Brazil. Therefore, sector-specific shocks (such as the impacts of the discovery of a competitive fuel source on the oil sector) are diluted in this asset. One important future work is related to the exploration of the proposed system to trade assets on specific sectors (such as the oil or utilities sectors) or with high price volatility (such as small sized companies or small caps).

The third group is related to the stock market itself, in which the proposed system is used. It is essential to evaluate its results (and the models and the hyperparameters chosen for its components) on different stock markets. Its characteristics and model fine-tuning results are related to the Brazilian stock market, but these may be different for other markets (for example, for trading the SP500 on the American market). Therefore, the proposed system can be implemented as described in this work on the Brazilian market, but it needs experimentation and fine-tuning for other stock markets. A portfolio optimization technique could be used to trade on multiple markets simultaneously, considering the proposed systems as assets in a complex portfolio.

The fourth group is related to future works to improve the price prediction module (M1). The central aspect that could be explored is the use of generative and adversarial models (such as generative adversarial networks) for price prediction. Those models have presented very interesting results in other ML domains. Another aspect that could be explored is the use of unsupervised models and autoencoders to detect distinct patterns on the data (compared to the typical time series analysis models). Lastly, a crucial aspect that must be explored is using different fundamental analysis indicators to improve the predictions.

The fifth group is related to market sentiment prediction and the M2 module. Several essential points could be explored in-depth, as this area is considerably less developed in Portuguese than in English. To identify different patterns and try to improve the models' prediction, models such as autoencoders and transformers could be used. Also, siamese neural networks could be used to identify patterns in the data better.

To improve the quality of the market sentiment prediction, several options could be further explored: (i) the use of social media messages as inputs; (ii) the use of the full texts of the news, instead of the news titles; and (iii) the development of sentiment dictionaries in Portuguese that are specific for the financial domain. The use of rules based on expert knowledge (such as expert systems that use fuzzy logic) could also be explored to improve the predictions' quality, as this is a very complex domain. Lastly, a severe limitation of

this field could be addressed: developing open datasets containing news and news titles in Portuguese with their timestamps. However, this is unlikely to happen soon for two main reasons: (i) it involves proprietary material from the news agencies, which are not willing to open them as raw data; and (ii) it would be necessary to label the data to use it for model fine-tuning, which is a very resource-intensive task.

Finally, transfer learning could be used to improve the quality of the sentiment prediction module. Two interesting options that could be explored are: (i) pre-training the sentiment analysis model using datasets from other sentiment analysis domains (such as product reviews and movie reviews, which have extensive datasets available); and (ii) translating the financial news headlines to English and using the NLP resources for the English language (such as the VADER sentiment analysis tool, domain-specific dictionaries, and the BERT model). Both options have the potential to improve the quality of the sentiment prediction module by incorporating more data for model training (in the first option) and more precise models (in the second option).

The sixth group is related to the use of DRL for stock market trading. One essential aspect that different researchers are already exploring is the use of different DRL agents for stock trading, such as actor-critic models (A2C and A3C), twin delayed DDPG (TD3), soft actor-critic (SAC), and DQN and its variations (dueling DQN and double DQN). As these models are considerably different, the patterns they recognize on the data and their decision-making may improve the models used in this work. Another critical aspect is the consideration of changes on the MDP framework used, considering: (i) different rewards functions, such as Sharpe ratio, log returns, or long-term rewards; (ii) different actions, such as shorting and using leverage; and (iii) considering more inputs in the state space, such as fundamental analysis indicators. It is also possible to incorporate rules for applying stop-gain and stop-loss actions, limiting the system's risk exposure.

One aspect that could considerably improve the trading system's results is considering an ensemble of different DRL agents. As the models recognize different patterns, their outputs may be aggregated to form a final, better decision (as was implemented and observed on the M1 module). Two main types of ensembles could be explored: (i) ensembles of agents with different trading frequencies for a specific asset; and (ii) ensembles of agents with different trading strategies for the same asset and trading frequency. Both types of ensembles have the potential to reduce the risks of the final trading by exploring different strategies. Lastly, using multiple trading systems to simulate different relevant agents in the market could provide critical information for decision-making instead of considering only one agent in action (as in this work and most of the literature on DRL for trading).

The two scenarios evaluated in this research were chosen because they represent typical and critical scenarios. The last group of recommended future works is related to evaluating new scenarios which were not considered in this research, such as: (i) a steep increase in price; (ii) a more volatile period (which may happen in the case of some specific sectors, such as retail or technology); (iii) longer scenarios (for example, a considerable amount of years); and (iv) a scenario that considers the whole period of an economic bubble, from burst to full recovery. All of those scenarios are more difficult to evaluate as they demand more data for model training. However, using the proposed system, it is possible to implement and evaluate them. Furthermore, the system could also simulate the impact of different optimistic and pessimistic future scenarios, such as the potential trading impacts of events that have a high asset impact and a low occurrence probability (such as an economic bubble on a specific sector). The following section concludes this chapter, summarizing the main points that were addressed.

5.5 Chapter summary

This chapter contained some of the most relevant discussions related to the proposed system, its components, and its potential applications. It also considered this work's main contributions, its main limitations, and recommendations for future works on different areas. Its main contribution is proposing a system that uses time series and sentiment analysis inputs, extracted with state-of-the-art DL models, for the DRL agent. Its four minor contributions are: (i) considering multiple features to predict Brazilian stock indices prices; (ii) considering market sentiment extracted from financial news titles in Portuguese; (iii) Using DRL for automatic stock trading of Brazilian stock indices, considering two different DRL agents; and (iv) using six financial domain metrics to evaluate models for stock trading.

Its main limitations were related to characteristics of the stock market, components of the proposed system, and lack of open datasets and code. Several recommendations for future works were proposed, considering: model inputs, traded assets, the stock market itself, and the main aspects of each module.

6 CONCLUSIONS

This chapter contains the main conclusions of this work. Section 6.1 reviews the main objective of this work and its research questions. Section 6.2 summarizes the proposed trading system, which is composed of a price prediction module (M1), a market sentiment prediction module (M2), and a trading module (MT). Section 6.3 summarizes the main contributions of this work. Lastly, section 6.4 contains the final remarks.

6.1 Main objective and research questions

Improving stock market trading results is the main objective of all trading models and strategies. Nevertheless, few of the proposed trading models and strategies that are currently available consider all of these three critical components: (i) a price prediction module that considers multiple features and that can be easily extended for more features; (ii) an automated market sentiment prediction module for data sources in Portuguese that does not need extensive labeling or expert knowledge (which is rarely available for individual investors); and (iii) a trading model that can adapt to current market changes, learn, and continuously develop new policies or strategies.

Therefore, this research's main objective was to develop a trading system that considers those three components (denominated modules in this work) and evaluate the main state-of-the-art models that could be used on each of them. An in-depth hyperparameters analysis was conducted for all models in the three modules, and several essential aspects were evaluated, such as: the use of different dictionaries for sentiment prediction, the use of technical indicators (TI) for predicting prices and for executing trades, and the use of several different features as inputs for the trading module. The trading module basis was a deep reinforcement learning (DRL) agent, which is starting to be explored as a method that could provide significantly better results than the buy and hold (BH) strategy, which is the main baseline used for evaluating trading models and strategies on different scenarios. It is essential to observe that only a few of the different models and

strategies currently used can provide better results (especially in terms of returns) than the BH in the long run.

Econometrics, traditional machine learning (ML), and state-of-the-art deep learning (DL) models were evaluated on the price prediction module (M1). On the sentiment prediction module (M2), the models evaluated were the multi-layer perceptron or MLP (a common baseline for deep neural networks models) and the convolutional neural network or CNN (considered in many works as the state-of-the-art model for sentiment analysis). Two DRL agents were evaluated on the trading module (MT): deep deterministic policy gradient (DDPG) and proximal policy optimization (PPO). Both are considered state-of-the-art on different domains, but few works apply them for financial trading.

Several research questions were proposed to guide and evaluate the different aspects of this research, such as the architecture of the proposed solution and the design and evaluation of experiments. The main research question was: "Does the proposed system present better trading results than the BH strategy, considering the evaluated risk and returns metrics and two different trading scenarios?". This is a fundamental question because it considers the investors' main objective (maximizing profit over time) in relation to a traditional and well-accepted baseline. It also considers the impacts in terms of returns and risks, which are essential for risk-averse stakeholders and trading on highly volatile scenarios.

In summary, the proposed system presented better results than the BH strategy in both scenarios. The configuration that presented the best results was: (i) using the long short-term memory network (LSTM) multivariate model with OHLCV and TI features for price prediction on the M1 module; (ii) not using the M2 module, unless the investor is highly risk-averse (because this module resulted in considerably lower returns on daily trading); and (iii) using the DDPG DRL agent for trading on the MT module, with a maximum order size of 200 for longer-term trading, and maximum order size of 10 for short-term trading on highly volatile scenarios.

Notwithstanding, it is essential to observe that: (i) as market dynamics are highly complex, those results do not guarantee that the trading system will provide the same results in all scenarios; and (ii) the experiments were conducted with the BOVA11 exchange traded fund (ETF), which is an easier way to trade on the Ibovespa index results so that the results may differ for other assets and markets.

There were also eight secondary questions related to the different aspects of the trading system's modules. These were:

- **SRQ1: "Does the use of DRL with sentiment analysis improve stock trading in terms of profits in relation to the use of the BH strategy?"**. It was observed that the use of the market sentiment prediction as a feature for the trading module did not improve the results in terms of returns-related metrics on Trade 1. Nevertheless, it improved the models' volatility-related and losses-related metrics. Although it may be an option for highly risk-averse investors, the best model configuration (MT1C) is still recommended, as it may provide higher returns in the long term.

- **SRQ2: "Which model results in the best forecast of market indices prices? Econometrics, ML, or DL models?"**. The models that presented the best results in terms of stock price prediction (M1) were: the model ensembles (SARIMAX and Support Vector Regressor or SVR, SARIMAX and LSTM, and SVR and LSTM), the SARIMAX, and the SVR multivariate models. However, the predictions' quality is not necessarily connected with the trading results, as was extensively observed in the literature (and also in this work).

- **SRQ3: "Does the use of TIs as features improve the forecasts of market indices prices?"**. The results for this question considering only the price prediction error were mixed, as some models observed improvement (SARIMAX and LSTM multivariate), while others did not (AdaBoost, LSTM univariate, SVR univariate, and SVR multivariate). Nevertheless, when the whole trading system is considered, the configuration that presented the best results considered TI as features for the LSTM multivariate for price prediction. Therefore, we conclude that the TI improves the trading system results for this specific asset and the scenarios evaluated.

- **SRQ4: "Which model (considering MLP and CNN) best predicts market sentiment?"**. It was observed that the model that best predicted market sentiment was the CNN (with a mean squared error or MSE that was around 37% lower than the MLP models, on average), which is in line with what is observed in the literature.

- **SRQ5: "Does the use of a dictionary (considering Sentilex, Oplexico, and WordnetAffectBR) improve the prediction of market sentiment for financial news headlines in Portuguese?"**. For both the MLP and the CNN models, it was observed that the use of dictionaries improves the quality of the sentiment prediction. The WordNetAffectBR dictionary presented the best results for both models. Therefore, it was concluded that dictionaries are essential for

improving the sentiment results on the M2 module. Notwithstanding, it is vital to observe that dictionaries are the aspect that least impact the evaluated models. The model's architecture (such as number of hidden layers, batch size, dropout rate, number of neurons, among others) has a considerably higher impact than the use of dictionaries. This is an important result, as a considerable amount of the literature on this domain focuses on finding the best dictionaries and lexicons, while the model's choice, architecture, and hyperparameters may impact the final results.

- **SRQ6: "Does the use of the price prediction module improve stock trading results?"**. Based on the experiments conducted on the two trading scenarios, it was concluded that the use of the M1 module improved the daily trading results considerably (especially the MT1C module) for the BOVA11 index in relation to the BH strategy. However, it is crucial to observe that this does not guarantee that the system will always present those results due to the market's complex dynamics.
- **SRQ7: "Does the use of sentiment analysis of news headlines improve stock trading results compared to the BH strategy?"**. Based on the experiments on the two trading scenarios, it was concluded that, although the use of the M2 module improved losses and volatility-related metrics, it did not improve the returns consistently on daily trading. This is especially true for scenarios similar to Trade 1. Therefore, although it can lead to more risk-averse models, it may also suffer from very low returns in low-volatility scenarios.
- **SRQ8: "Does the use of TIs improve the stock trading results of DRL models?"**. As explored on SRQ3, the use of TI improved the results for price prediction for some models. Nevertheless, using TI directly as a feature for the DRL models did not improve the stock trading results for both evaluated scenarios. Therefore, it is recommended to use them as features for the M1 module but not for the MT module.

The following section contains a brief description of the proposed trading system and its main modules. It also contains the models that were chosen based on the several experiments that were conducted.

6.2 Proposed trading system

The trading system proposed in this work contains three modules: (i) M1, which is a stock price prediction module; (ii) M2, which is a stock market sentiment prediction

module; and (iii) MT, which is a stock trading module using a DRL agent. The main inputs for M1 are open, high, low, and close prices, and volume (OHCLV) data and TI. The inputs for M2 are the relevant news titles in Portuguese during each trading day. The main inputs for the MT module depend on the several models evaluated.

The last configuration of the system contained: (i) for M1: a multivariate LSTM with OHCLV and TI inputs; (ii) for M2: a CNN with the GloVe word embedding in Portuguese and the WordNetAffectBR dictionary, and news titles as inputs; and (iii) for MT: the DDPG DRL agent with a maximum order size of 200 (for long-term trading) or of 10 (for short-term trading on highly volatile scenarios, especially on steep downward trends).

After all the components of the trading system are trained (considering the datasets that were developed and the hyperparameter values that provided the best results on the experiments), its working is relatively simple: (i) each day, new relevant data must be gathered (OHCLV data for M1 and news titles in Portuguese for M2); (ii) this data must be inserted on the models for generating the predictions; and (iii) the MT module will output the trading signal (how much to buy or sell on that specific day). The information from item iii can then be considered for trading on that same day or for sending a trading order for the next day, depending on the investor's strategy.

In summary, the Markov Decision Process (MDP) of the DRL agent implemented considers: (i) in the state space, information about the portfolio and the input features for the day; (ii) in the action space, the actions of buying, selling, or not acting in that day (taking into account the maximum order size as a restraint on the buying and selling actions); and (iii) the change of portfolio value on each trading day as the reward function. The library used for implementation, FinRL, also allows for changes in all those aspects (for example, creating new actions such as shorting or reward functions such as the Sharpe ratio or log returns).

The most common form of operation envisioned is to send an order for the next trading day. It is also important to note that it is possible to add functionalities to make the whole trading system autonomous. In that case, the MT module's output will be sent to a brokerage service, implementing the order in the stock market. It is also essential to observe that the modules' configurations can be changed, to explore different predictions or scenarios. Nevertheless, in this case, all modules must be trained again. Lastly, it is vital to observe that the modules must be trained periodically, to incorporate possible new information or changes in the market dynamics.

6.3 Main contributions

This research had one main contribution (which is applicable to different domains) and four minor contributions (which are applicable to the stock trading domain).

The main contribution of this work is related to the proposal of a system that uses time series and sentiment analysis features, extracted with state-of-the-art DL models, for the DRL agent. This is an important contribution for applications of DRL agents on complex scenarios with unknown and changing dynamics and low signal-to-noise, which is characteristic of stock market trading. However, the proposed system can be used in different domains. Two additional inputs were proposed: (i) time series prediction features; and (ii) sentiment prediction features. The experiments conducted on two scenarios on the Brazilian stock market showed that the proposed system provides better results than the baseline, considering six financial metrics. The use of the additional inputs led to better information extracted from the data provided by the environment.

The first minor contribution was related to the M1 module by considering multiple features to predict Brazilian stock indices prices and different models, hyperparameters, cross-validation methods, and TI use. Econometrics, ML, and DL models were evaluated on univariate and multivariate forms. It was observed that ensemble models provided the best results and that the best individual model was the SARIMAX with block time split and OHCLV and TI as inputs. This is an important contribution for exploring different models (especially comparing econometrics, DL, and ensemble models), hyperparameters, cross-validation methods, and TI use for trading market indices.

The second minor contribution was related to the M2 module by considering market sentiment extracted from financial news titles in Portuguese for daily trading purposes. The majority of the works in the literature are in English and focus on developed markets. Therefore, it is vital to evaluate the impact of predicting and considering the sentiment in scenarios that are similar to real-life trading. Two important models were evaluated: MLP and CNN, along with their main hyperparameters and several important sentiment dictionaries in Portuguese. The CNN with the WordNetAffectBR dictionary provided the best results in terms of sentiment prediction. In terms of trading, the market sentiment feature's use proved to be important to improve metrics related to volatility and losses. However, the trading system's best configuration did not consider the market sentiment for daily trading, as the returns obtained by using them as features were too low compared to the other configurations of the trading system.

The third minor contribution was related to using DRL for automatic stock trading of Brazilian stock indices. The use of DRL for stock trading is considerably new, and there is much interest both by researchers and practitioners because those models have been proven to provide good results in complex scenarios in other domains. Nevertheless, this is one of the few works that evaluate in-depth two DRL agents (DDPG and PPO) for trading in the Brazilian stock market. Most of the works in the literature focus on one RL or DRL agent and developed markets. Also, few works in the literature consider additional features for the OHLCV traditional data. This is one of the first works to consider market sentiment, TI, and price prediction inputs for trading with a DRL agent. The impact of the maximum order size was also considered. This is very relevant to highly volatile scenarios and for trading during a steep downward market (as was the case during the Covid-19 pandemics, as illustrated in the Trade 2 scenario).

The fourth minor contribution was related to the use of financial domain metrics to evaluate models for stock trading. This is an important point that is stressed in several important works in the literature, as the majority of the works focus on traditional ML metrics for classification or regression. Also, there are very few papers that consider metrics from several important categories. In this work, six financial metrics were considered for evaluating the different models' results on the two trading scenarios and also to compare them with the BH strategy. These metrics were aggregated in four categories to reflect better their objectives: (i) returns-related metrics: annual returns and cumulative returns; (ii) volatility-related metrics: annual volatility and stability; (iii) risk-related metrics: Sharpe ratio; and (iv) loss-related metrics: maximum drawdown. It is also possible to use the proposed system with different metrics, which may be necessary for different scenarios or sectors. The use of those metrics improved the comprehension of the different trading models considered.

The following section contains the final remarks related to this research.

6.4 Final remarks

This section concludes the research. As it was described, the problem studied is of great interest to investors and companies of all sizes. Better predictions and better trading results could improve the investors' results regarding returns and risk exposure, volatility, and potential losses. The results of the proposed system on the two trading scenarios are exciting, in the sense that the best configuration of the whole system resulted in better results for all evaluated metrics for the highly volatile scenario (Trade 2) and

better returns for the longer-term scenario (Trade 1).

Although more exploration is needed, it is already possible to implement the proposed trading system as a pilot project for a commercial automated trading platform. This would be a direct competitor to the so-called trading robots, which are already becoming a reality in the Brazilian market. Nevertheless, more work is needed to explore further the system and its components' behavior in different scenarios and also in relation to other trading robots. Additionally, several points should be addressed to implement the proposed system in real-life scenarios, such as production implementation aspects and brokerage services' connection for automatically adding trading orders.

This work's main contribution may improve the literature related to DRL agents on different domains. Its four minor contributions may improve price prediction and algorithmic trading, especially on trading with RL and DRL agents. Also, it can provide an important contribution to the Brazilian market, which is rarely explored in the trading and price prediction literature. Also, the methodology used can be further explored to develop new modules, implement new models in each module, implement new features (especially on the M2 module, considering several data sources such as social media messages, experts' analysis, press releases, among others), and implement new reward functions and action and state spaces for the DRL agent. One critical aspect that could be further explored is the use and behavior of different DRL agents and if they could be used as an ensemble to improve the overall trading results.

REFERENCES

ARULKUMARAN, K. et al. Deep reinforcement learning: a brief survey. **IEEE Signal Processing Magazine**, v. 34, n. 6, p. 26–38, 2017.

B3. **Bovespa - Market data e índices**. 2021. Disponível em: <http://www.b3.com.br/pt_br/market-data-e-indices/servicos-de-dados/market-data/>. Acesso em: 2021-01-06.

B3. **Índice Ibovespa**. 2021. Disponível em: <http://www.b3.com.br/pt_br/market-data-e-indices/indices/indices-amplos/ibovespa.htm>. Acesso em: 2021-02-06.

BALLINGS, M. et al. Evaluating multiple classifiers for stock price direction prediction. **Expert Systems with Applications**, v. 42, n. 20, p. 7046–7056, 2015.

BOLLEN, J.; MAO, H.; ZENG, X.-j. Twitter mood predicts the stock market. **Journal of Computational Science**, v. 2, n. 1, p. 1–8, 2011.

BONDT, W. F. M.; THALER, R. Does the stock market overreact? **The Journal of Finance**, v. 40, n. 3, p. 793–805, 1985.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: **Proceedings of the Fifth Annual Workshop on Computational Learning Theory**. Pittsburgh: ACM, p. 144–152, 1992.

BOX, G. E. P. et al. **Time series analysis: forecasting and control**. 5th edition. ed. [S.l.]: John Wiley & Sons, 2015. 712 p. ISBN 1118674928.

CARVALHO, P.; SILVA, M. J. SentiLex-PT: Principais características e potencialidades. **Oslo Studies in Language**, v. 7, n. 1, 2015.

CHANG, C.-C.; LIN, C.-J. LIBSVM: a library for support vector machines. **ACM Transactions on Intelligent Systems and Technology**, v. 2, n. 3, p. 1–27, 2011.

CHIMMULA, V. K. R.; ZHANG, L. Time series forecasting of COVID-19 transmission in Canada using LSTM networks. **Chaos, Solitons Fractals**, v. 135, p. 109864, 2020.

CHONG, E.; HAN, C.; PARK, F. C. Deep learning networks for stock market analysis and prediction : Methodology, data representations, and case studies. **Expert Systems With Applications**, v. 83, p. 187–205, 2017.

CONEGUNDES, L.; PEREIRA, A. Beating the stock market with a deep reinforcement learning day trading system. In: **Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN)**. Glasgow: IEEE, p. 1–8, 2020.

DANTAS, S. G.; SILVA, D. G. Equity Trading at the Brazilian Stock Market Using a Q-Learning Based System. In: **Proceedings of the 2018 7th Brazilian Conference on Intelligent Systems (BRACIS)**. São Paulo: IEEE, p. 133–138, 2018.

DEAN, N. E. et al. Ensemble forecast modeling for the design of COVID-19 vaccine efficacy trials. **Vaccine**, v. 38, n. 46, p. 7213–7216, 2020.

DRIDI, A.; ATZENI, M.; RECUPERO, D. R. FineNews : fine - grained semantic sentiment analysis on financial microblogs and news. **International Journal of Machine Learning and Cybernetics**, v. 10, n. 8, p. 2199–2207, 2019.

DRUCKER, H. Improving regressors using boosting techniques. In: **Proceedings of the Fourteenth International Conference on Machine Learning (ICML)**. [S.l.]: ACM, p. 107–115, 1997.

DRUCKER, H. et al. Support vector regression machines. **Advances in Neural Information Processing Systems**, v. 9, p. 155–161, 1997.

EAPEN, J.; VERMA, A.; BEIN, D. Novel deep learning model with CNN and bi-directional LSTM for improved stock market index prediction. In: **Proceedings of the 2019 IEEE 9th annual computing and communication workshop and conference (CCWC)**. Las Vegas: IEEE, p. 264–270, 2019.

FAMA, E. F. Random walks in stock market prices. **Financial Analysts Journal**, v. 51, n. 1, p. 75–80, 1965.

FERREIRA, T. M. et al. Assessing regression-based sentiment analysis techniques in financial texts. In: **Anais do XVI Encontro Nacional de Inteligência Computacional e Artificial (ENIAC)**. Porto Alegre: SBC, p. 729–740, 2019.

FISCHER, T. Reinforcement learning in financial markets - a survey. **FAU Discussion Papers in Economics**, v. 12/2018, p. 1–48, 2018.

FISCHER, T.; KRAUSS, C. Deep learning with long short-term memory networks for financial market predictions. **European Journal of Operational Research**, v. 270, n. 2, p. 654–669, 2018.

FRANÇOIS-LAVET, V. et al. An introduction to deep reinforcement learning. **Foundations and Trends in Machine Learning**, v. 11, n. 3-4, p. 1–140, 2018.

FREITAS, F. D.; SOUZA, A. F. D.; ALMEIDA, A. R. D. Prediction-based portfolio optimization model using neural networks. **Neurocomputing**, v. 72, p. 2155–2170, 2009.

FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. **Journal of Computer and System Sciences**, v. 55, n. 1, p. 119–139, 1997.

GHOSAL, D. et al. IITP at SemEval-2017 Task 5 : an ensemble of deep learning and feature based models for financial sentiment analysis. In: **Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)**. [S.l.: s.n.], p. 899–903, 2017. v. 1.

GILDO, J.; JÚNIOR, D. A.; MARINHO, L. B. Using online economic news to predict trends in Brazilian stock market sectors. In: **Proceedings of the 24th Brazilian Symposium on Multimedia and the Web**. Salvador: ACM, p. 37–44, 2018.

GUO, S. et al. An autonomous path planning model for unmanned ships based on deep reinforcement learning. **Sensors**, v. 20, n. 2, p. 426, 2020.

GURESEN, E.; KAYAKUTLU, G.; DAIM, T. U. Using artificial neural network models in stock market index prediction. **Expert Systems With Applications**, v. 38, n. 8, p. 10389–10397, 2011.

HARTMANN, N. et al. Portuguese word embeddings: evaluating on word analogies and natural language tasks. **arXiv preprint arXiv:1708.06025**, 2017.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, v. 9, n. 8, p. 1735–1780, 1997.

HU, Z. et al. Listening to chaotic whispers : a deep learning framework for news-oriented stock trend prediction. In: **Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining**. Los Angeles: ACM, p. 261–269, 2018.

HUTTO, C. J.; GILBERT, E. VADER : a parsimonious rule-based model for sentiment analysis of social media text. In: **Proceedings of the Eighth International AAAI Conference on Weblogs and Social Media**. Palo Alto: AAAI, p. 216–225, 2014.

JIN, Z.; YANG, Y.; LIU, Y. Stock closing price prediction based on sentiment analysis and LSTM. **Neural Computing and Applications**, v. 32, n. 13, p. 9713–9729, 2020.

JOHNMAN, M.; VANSTONE, B. J.; GEPP, A. Predicting FTSE 100 returns and volatility using sentiment analysis. **Accounting Finance**, v. 58, p. 253–274, 2018.

JORDAN, M. I.; MITCHELL, T. M. Machine learning: trends, perspectives, and prospects. **Science**, v. 349, n. 6245, p. 255–260, 2015.

JUNIOR, P. R.; SALOMON, F. L. R.; PAMPLONA, E. D. O. ARIMA : an applied time series forecasting model for the Bovespa stock index. **Applied Mathematics**, v. 5, p. 3383–3391, 2014.

KARA, Y.; BOYACIOGLU, M. A.; BAYKAN, O. K. Predicting direction of stock price index movement using artificial neural networks and support vector machines : the sample of the Istanbul Stock Exchange. **Expert Systems With Applications**, v. 38, n. 5, p. 5311–5319, 2011.

KARPATHY, A.; JOHNSON, J.; FEI-FEI, L. Visualizing and understanding recurrent networks. **arXiv preprint arXiv:1506.02078**, p. 1–12, 2015.

KOUTSOUKAS, A. et al. Deep-learning: investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data. **Journal of Cheminformatics**, v. 9, n. 1, p. 1–13, 2017.

KRISTJANPOLLER, W.; FADIC, A.; MINUTOLO, M. C. Volatility forecast using hybrid Neural Network models. **Expert Systems With Applications**, v. 41, n. 5, p. 2437–2442, 2014.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, v. 521, n. 7553, p. 436–444, 2015.

LECUN, Y. et al. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, IEEE, v. 86, n. 11, p. 2278–2324, 1998.

LEI, K. et al. Time-driven feature-aware jointly deep reinforcement learning for financial signal representation and algorithmic trading. **Expert Systems with Applications**, v. 140, p. 1–14, 2020.

LI, J.; RAO, R.; SHI, J. Learning to trade with deep actor critic methods. In: **Proceedings of the 2018 11th International Symposium on Computational Intelligence and Design (ISCID)**. Hangzhou: IEEE, p. 66–71, 2018. v. 2.

LI, Y. Deep reinforcement learning. **arXiv preprint arXiv:1810.06339**, p. 1–150, 2018.

LIESSNER, R. et al. Deep Reinforcement Learning for Advanced Energy Management of Hybrid Electric Vehicles. In: **Proceedings of the International Conference on Agents and Artificial Intelligence (ICAART)**. [S.l.: s.n.], p. 61–72, 2018.

LILLICRAP, T. P. et al. Continuous control with deep reinforcement learning. **arXiv preprint arXiv:1509.02971**, p. 1–14, 2016.

LIM, H.-K. et al. Federated reinforcement learning for training control policies on multiple IoT devices. **Sensors**, v. 20, n. 5, p. 1–15, 2020.

LIU, X.-y. et al. FinRL : a deep reinforcement learning library for automated stock trading in Quantitative Finance. In: **Proceedings of the 34th NeurIPS 2020 Deep RL Workshop**. [S.l.]: NeurIPS, p. 1–11, 2020.

LOUGHRAN, T. I. M.; MCDONALD, B. When is a liability not a liability ? Textual analysis, dictionaries , and 10-Ks. **Journal of Finance**, v. 66, n. 1, p. 35–65, 2011.

LOUKAS, G. et al. Cloud-based cyber-physical intrusion detection for vehicles using deep learning. **IEEE Access**, v. 6, p. 3491–3508, 2017.

MACAL, C.; NORTH, M. Introductory tutorial: Agent-based modeling and simulation. In: IEEE. **Proceedings of the Winter Simulation Conference 2014**. [S.l.], p. 6–20, 2014.

MALKIEL, B. G.; FAMA, E. F. Efficient capital markets: a review of theory and empirical work. **The Journal of Finance**, v. 25, n. 2, p. 383–417, 1970.

MANSAR, Y. et al. Fortia-FBK at SemEval-2017 task 5: bullish or bearish? Inferring sentiment towards brands from financial news headlines. In: **Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)**. Vancouver: ACL, p. 1–6, 2017.

MARTINEZ, L. C. et al. From an artificial neural network to a stock market day-trading system : a case study on the BMF BOVESPA. In: **Proceedings of the 2009 International Joint Conference on Neural Networks (IJCNN)**. Atlanta: IEEE, p. 2006–2013, 2009.

MARTO, T. et al. Development of a labelled dataset of Brazilian agricultural market news titles for sentiment analysis. In: **Anais do XII Congresso Brasileiro de Agroinformática (SBIAGRO)**. Indaiatuba: SBIAgro, p. 1–10, 2019.

MEDEIROS, M. C.; BORGES, V. R. P. Tweet sentiment analysis regarding the Brazilian stock market. In: **Anais do VIII Brazilian Workshop on Social Network Analysis and Mining**. Belém: SBC, p. 71–82, 2019.

MEDHAT, W.; HASSAN, A.; KORASHY, H. Sentiment analysis algorithms and applications: a survey. **Ain Shams Engineering Journal**, v. 5, n. 4, p. 1093–1113, 2014.

MEHTAB, S.; SEN, J.; DASGUPTA, S. Robust Analysis of Stock Price Time Series Using CNN and LSTM-Based Deep Learning Models. In: **Proceedings of the Fourth International Conference on Electronics, Communication and Aerospace Technology (ICECA-2020)**. Coimbatore: IEEE, p. 1481–1486, 2020.

MENG, T. L.; KHUSHI, M. Reinforcement Learning in Financial Markets. **Data**, v. 4, n. 110, p. 1–17, 2019.

MNIH, V. et al. Human-level control through deep reinforcement learning. **Nature**, v. 518, n. 7540, p. 529–533, 2015.

NASSIRTOUSSI, A. K. et al. Text mining for market prediction : a systematic review. **Expert Systems with Applications**, v. 41, n. 16, p. 7653–7670, 2014.

NEGAHBAN, A.; YILMAZ, L. Agent-based simulation applications in marketing research: an integrated review. **Journal of Simulation**, v. 8, n. 2, p. 129–142, 2014.

NELSON, D. M. Q.; PEREIRA, A. C. M.; OLIVEIRA, R. A. D. Stock market's price movement prediction with LSTM neural networks. In: **Proceedings of the 2017 International joint conference on neural networks (IJCNN)**. Anchorage: IEEE, p. 1419–1426, 2017.

NEUENSCHWANDER, B. et al. Sentiment analysis for streams of web data : a case study of Brazilian financial markets. In: **Proceedings of the 20th Brazilian Symposium on Multimedia and the Web**. João Pessoa: ACM, p. 167–170, 2014.

OLIVEIRA, F. A. D.; NOBRE, C. N.; ZÁRATE, L. E. Applying artificial neural networks to prediction of stock price and improvement of the directional prediction index – case study of PETR4, Petrobras, Brazil. **Expert Systems With Applications**, v. 40, n. 18, p. 7596–7606, 2013.

PASQUALOTTI, P. R.; VIEIRA, R. WordnetAffectBR: uma base lexical de palavras de emoções para a língua portuguesa. **RENOTE-Revista Novas Tecnologias na Educação**, v. 6, n. 1, 2008.

PAULI, S.; KLEINA, M.; BONAT, W. Comparing artificial neural network architectures for Brazilian stock market prediction. **Annals of Data Science**, v. 7, n. 4, p. 613–628, 2020.

PENDHARKAR, P. C.; CUSATIS, P. Trading financial indices with reinforcement learning agents. **Expert Systems with Applications**, v. 103, p. 1–13, 2018.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. GloVe : global vectors for word representation. In: **Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. Doha: ACL, p. 1532–1543, 2014.

PERSIO, L.; HONCHAR, O. Artificial Neural Networks architectures for stock price prediction : comparisons and applications. **International Journal of Circuits, Systems and Signal Processing**, v. 10, p. 403–413, 2016.

PLATT, J. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. **Advances in large margin classifiers**, v. 10, n. 3, p. 61–74, 1999.

RIBEIRO, M. H. D. M. et al. Short-term forecasting COVID-19 cumulative confirmed cases: perspectives for Brazil. **Chaos, Solitons Fractals**, v. 135, p. 109853, 2020.

RUDER, S.; GHAFFARI, P.; BRESLIN, J. G. A hierarchical model of reviews for aspect-based sentiment analysis. In: **Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. Austin: ACL, p. 999–1005, 2016.

RUNDO, F. et al. Machine learning for quantitative finance applications : a survey. **Applied Sciences**, v. 9, n. 5574, p. 1–20, 2019.

RYLL, L.; SEIDENS, S. Evaluating the performance of machine learning algorithms in financial market forecasting : a comprehensive survey. **arXiv preprint arXiv:1906.07786**, p. 1–22, 2019.

SAJAD, S.; SCHUKAT, M.; HOWLEY, E. Deep reinforcement learning : an overview. In: **Proceedings of SAI Intelligent Systems Conference (IntelliSys)**. London: SAI, p. 426–440, 2016.

SCHULMAN, J. et al. Proximal policy optimization algorithms. **arXiv preprint arXiv:1707.06347**, p. 1–12, 2017.

SEZER, O. B.; GUDELEK, M. U.; OZBAYOGLU, A. M. Financial time series forecasting with deep learning : a systematic literature review : 2005 – 2019. **Applied Soft Computing Journal**, v. 90, p. 106181, 2020.

SIAMI-NAMINI, S.; TAVAKOLI, N.; NAMIN, A. S. A comparative analysis of forecasting financial time series using ARIMA, LSTM, and BiLSTM. **arXiv preprint arXiv:1911.09512**, p. 1–8, 2019.

SILVA, M. J.; CARVALHO, P.; SARMENTO, L. Building a sentiment lexicon for social judgement mining. In: **Proceedings of the International Conference on Computational Processing of the Portuguese Language**. [S.l.]: Springer, p. 218–228, 2012.

SILVER, D. et al. Mastering the game of Go with deep neural networks and tree search. **Nature**, v. 529, n. 7585, p. 484–489, 2016.

SILVER, D. et al. Deterministic policy gradient algorithms. In: **Proceedings of the International Conference on Machine Learning (ICML)**. [S.l.]: PMLR, p. 387–395, 2014.

SOHANGIR, S. et al. Big data : deep Learning for financial sentiment analysis. **Journal of Big Data**, v. 5, n. 3, p. 1–25, 2018.

SOUMA, W.; VODENSKA, I.; AOYAMA, H. Enhanced news sentiment analysis using deep learning methods. **Journal of Computational Social Science**, v. 2, n. 1, p. 33–46, 2019.

SOUZA, M.; VIEIRA, R. Sentiment analysis on twitter data for portuguese language. In: **Proceedings of the International Conference on Computational Processing of the Portuguese Language**. [S.l.]: Springer, p. 241–247, 2012.

SUTTON, R. S.; BARTO, A. G. **Reinforcement learning: an introduction**. 2nd edition. ed. [S.l.]: MIT press, 2018. 552 p. ISBN 0262352702.

TRELEAVEN, B. P.; GALAS, M.; LALCHAND, V. Algorithmic Trading Review. **Communications of the ACM**, v. 56, n. 11, p. 76–85, 2013.

VÁZQUEZ-CANTELI, J. R.; NAGY, Z. Reinforcement learning for demand response : A review of algorithms and modeling techniques. **Applied Energy**, v. 235, p. 1072–1089, 2019.

WANG, J. et al. AlphaStock : a buying-winners-and-selling-losers investment strategy using interpretable deep reinforcement attention networks. In: **Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining**. Anchorage: ACM, p. 1900–1908, 2019.

WANG, Y. et al. **Deep Q-trading**. [S.l.], 2017. v. 20160036, 1–9 p.

WANG, Z.; ZHANG, J.; VERMA, N. Realizing low-energy classification systems by implementing matrix multiplication directly within an ADC. **IEEE Transactions on Biomedical Circuits and Systems**, v. 9, n. 6, p. 825–837, 2015.

WENG, B. et al. Predicting short-term stock prices using ensemble methods and online data sources. **Expert Systems With Applications**, v. 112, p. 258–273, 2018.

WORLD BANK. **GDP per country (current US$)**. 2021. Disponível em: <https://data.worldbank.org/indicator/NY.GDP.MKTP.CD>. Acesso em: 2021-01-05.

WU, J. et al. Quantitative Trading on Stock Market Based on Deep Reinforcement Learning. In: **Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN)**. Budapest: IEEE, p. 1–8, 2019.

YADAV, A.; JHA, C. K.; SHARAN, A. Optimizing LSTM for time series prediction in Indian stock market. **Procedia Computer Science**, v. 167, p. 2091–2100, 2020.

YANG, H. et al. Deep reinforcement learning for automated stock trading : an ensemble strategy. In: **Proceedings of ICAIF2020: ACM International Conference on AI in Finance**. New York: ACM, p. 1–9, 2020.

ZEROUAL, A. et al. Deep learning methods for forecasting COVID-19 time-Series data: a comparative study. **Chaos, Solitons Fractals**, v. 140, p. 110121, 2020.

ZHANG, L.; WANG, S.; LIU, B. Deep learning for sentiment analysis : a survey. **Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery**, v. 8, n. 4, p. 1–25, 2018.

ZHANG, Y.; WALLACE, B. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. **arXiv preprint arXiv:1510.03820**, p. 1–18, 2015.